

SOLiD™ System Application Documentation: AB Resequencing Analysis Pipeline (“Corona Lite”)

1	PROGRAM	3
2	INTRODUCTION	3
3	SYSTEM REQUIREMENTS	3
4	INSTALLATION	4
5	RUNNING CORONA LITE	5
6	JOB SCHEDULING OVERVIEW	5
7	RUNNING TIME	6
8	MATCHING PIPELINE DOCUMENTATION	6
8.1	Matching Pipeline Preparation	6
8.2	Matching Pipeline Synopsis.....	8
8.3	Matching Pipeline Output Files	9
8.4	Matching Pipeline Script Usage	11
9	PAIRING PIPELINE DOCUMENTATION	12
9.1	Pairing Pipeline Preparation.....	13
9.2	Pairing Pipeline Synopsis.....	13
9.3	Pairing Pipeline Output Files.....	15
9.4	Pairing Pipeline Script Usage.....	17
9.5	Pairing Pipeline Small Indel Option.....	19
9.5.1	Small Indel Option Output files	19
9.5.2	Small Indel Option Post-processing	20
9.5.3	Other Considerations	21
9.5.4	Indel Alignment Information	21
10	SNP PIPELINE DOCUMENTATION	22

10.1	SNP Pipeline Preparation	22
10.2	SNP Pipeline Synopsis.....	22
10.3	SNP Pipeline Output Files	25
10.4	SNP Pipeline Script Usage.....	27
11	SAMPLE DATA	29
12	SOFTWARE CHANGE HISTORY	30

1.0	1/29/2009	First draft.
-----	-----------	--------------

1 Program

Program Name: AB Resequencing Analysis Pipeline (“Corona Lite”)

Program Version: 4.0

Development Languages: Perl, Python, Java

PBS Is Required: No

2 Introduction

Corona Lite is an off-instrument SOLiD data analysis software package that can be used to analyze large and complex genomes and align SOLiD reads against reference sequences. The current capabilities include:

- Mapping SOLiD reads to reference genomes, including genomes consisting of multiple chromosomes.
- Mapping SOLiD data against sequence databases containing thousands of sequences.
- Pairing of matched mate-pair run reads.
- Reporting small indels on mate-pair data.
- Calling single nucleotide polymorphisms (SNPs) and generating consensus sequences.

Applied Biosystems has tested Corona Lite with both the PBS and the LSF job scheduler systems. It has also been successfully used with Sun Grid Engine (SGE), and it should work on any computer farm system with minor modifications to some scripts. See **Job Scheduling Overview** for more details.

3 System Requirements

CPU: 64 bit Intel or AMD CPU

RAM: 2 GB to 4 GB per CPU

Note: Read mapping requires about six times the reference length plus 1 GB of RAM per job. For example, the Matching pipeline requires about 2.7 GB RAM per job when running against human chromosome 1 (247M base pairs). Pairing requires approximately 3 GB per job. If you have 2 GB per CPU, you will have to restrict the jobs to 2 CPUs per job or 3 GB RAM per job.

Disk available to the head node:

Disk requirements will vary depending on the protocol used and the size of the sample. Each pipeline generates many files; and some of the files contain intermediate results that may be deleted to recover disk space. (See the individual “Output Files” sections for more details.) The table below describes the intermediate and final disk requirements of each pipeline for a typical case.

Pipeline	Intermediate	Final
Matching (1)	100G	40-80G
Mate-pairing (2)	200-300G	30-50G
SNP & consensus	240G (3)	10-20G
Total:		80-150G

Notes:

1. Mapping of 120 Million 25 base pair reads to the human genome allowing 2 mismatches.
2. Pairing of 2 runs of 120 Million reads each.
3. Approximately 8 times the size of the reference genome.

Disk locally available to each compute node: 50GB per job recommended

Operating System: 64-bit Linux

Python Version: 2.3.4 and later

Perl Version: 5.8.5 and above

Job Scheduler Compatibility: Portable Batch System (PBS), **(Optional)** Load Sharing Facility (LSF), Sun Grid Engine (SGE)

4 Installation

1. Copy the Corona Lite tgz file to a directory (<INSTALL_DIR>) where the software is to be installed. Note that this directory has to be accessible from all the nodes in the cluster. Enter the following:

```
% cd <INSTALL_DIR>
% tar xvfz corona_lite_vx.xrx.x.tgz
```

The archive will be expanded into the directory
<INSTALL_DIR>/corona_lite_version.

2. Update the script paths for Perl and Python. By default, Corona Lite sets these paths to /usr/bin/perl and /usr/bin/python respectively.

For example to change the python path to /usr/local/bin/python, enter the following:

```
% cd <INSTALL_DIR>/corona_lite/bin
% perl -i -pe 's[/usr/bin/python][usr/local/bin/python]' *.py
```

3. Configure your environment, and update your shell's init script (.cshrc, .bashrc, and so on) for future sessions with Corona Lite.

For csh/tcsh:

```
% setenv CORONAROOT <INSTALL_DIR>/corona_lite
% source $CORONAROOT/etc/profile.d/corona.csh
```

For sh/ksh/bash:

```
%export CORONAROOT=<INSTALL_DIR>/corona_lite  
%source $CORONAROOT/etc/profile.d/corona.sh
```

4. **(Optional)** Rebuild mapreads package.

This package comes with pre-built binaries for the Linux 64-bit platform. To rebuild the binary components of mapreads, enter the following:

```
%cd $CORONAROOT/src/  
%make
```

The package is now ready for use.

5 Running Corona Lite

Corona Lite consists of three pipelines: **Matching**, **Pairing** and **SNP/Consensus Calling**. The Pairing pipeline also has an option to find small-indels using mate-paired data. Each pipeline is applied by executing several scripts from the command line.

The Matching pipeline takes a set of SOLiD reads and aligns them to a reference genome. When analyzing mate-pair reads, the reads from the first and second primers (named F3 & R3) are matched to the reference genome with two independent runs of the Matching pipeline.

The Pairing pipeline takes the two mate-pair results from the Matching pipeline and uses the mate-pair information to improve the accuracy of the read matching, find small indels, reports sites of structural variation, and recover unmapped reads.

The SNP pipeline takes the results of either the Matching pipeline or Pairing pipeline and produces SNP calls. This pipeline can also call SNPs from multiple runs to produce a single consensus.

6 Job Scheduling Overview

Applying each of Corona Lite's pipelines involves executing a script that divides the analysis job into many sub-jobs for distribution to the compute nodes by a job scheduler. The sub-jobs are defined as a number of shell scripts saved to a subdirectory named `scripts`. Paths to these scripts are listed in a file named `JOBS_LIST.txt`, which is a tab delimited text file with the following columns:

Column 1:

Job ID: Sequential job identifier number starting with 1. Jobs are sent to the scheduler in order of Job ID.

Column 2:

Script Name: Full path to the scripts defining each sub-job.

Column 3:

Wait for jobs list: Comma separated list of Job IDs that must complete before any successive jobs are scheduled for execution.

These jobs are then sent to the PBS or LSF by entering one of the following:

```
% submit_scripts_to_PBS.pl -j JOB_LIST.txt &
% submit_scripts_to_LSF.pl -j JOB_LIST.txt &
```

Similar scripts can be written to schedule these jobs using Sun Grid Engine (SGE) or other job scheduling systems.

To execute the jobs **without** a scheduling system, locally execute each of the scripts in JOB_LIST.txt as follows:

```
% cut -f2 JOB_LIST.txt > cmd
% sh cmd
```

7 Running Time

Each pipeline requires about a day to finish if given enough CPUs.

The exact run time may vary. The longer the reads, and/or the more mismatch allowed, the more time is required to run the matching job.

8 Matching Pipeline Documentation

Corona Lite's Matching pipeline is the process of aligning SOLiD Color-space reads (a .csfasta file) to a reference genome sequence.

8.1 Matching Pipeline Preparation

To perform matching you need the following:

1. SOLiD Reads

A single .csfasta file containing the reads to be aligned with the reference. The file can be compressed using gzip to save disk space and reduce network I/O.

The pipeline can analyze the compressed reads file without decompression, and supports Reads files compressed by the gzip utility. To create a compressed Reads file, enter the following:

```
% gzip reads.csfasta
```

This command will produce the compressed reads file `reads.csfasta.gz`. To keep the uncompressed version, use the following command instead:

```
% gzip < reads.csfasta > reads.csfasta.gz
```

The reads within the reads file must be in the same original order as when generated from the instrument, ordered by Panel_id, X and Y coordinates. If the reads are out of order, use the following command to sort the file:

```
% radsort unsorted.csfasta 6 > sorted.csfasta
```

2. Reference Sequences

A set of fasta files defining the reference sequence.

- For a well-assembled genome, each file should contain one sequence entry representing a single chromosome.
- For a draft genome that may not have been assembled into chromosomes but is in contig format, or a sequence database consisting of many tens or hundreds of thousands of sequences, each file needs to contain between 100-250 megabases of sequence.

Requirements:

- Each file should contain one sequence.
- The sequence portion of the fasta file should contain only A, C, G, N, T, and newline characters.
- Only uppercase base characters are permitted.
- Paths to these files will be referenced in the Chromosome Map (cmap) file, described in item 4 below.

A reference sequence should be validated and formatted using the following commands:

```
% reference_validation.pl -r raw_sequence  
-o validated_sequence -s 999999999
```

3. Double Encoded Reference Genome Sequences (Required only for SNP pipeline)

A set of fasta files containing the double encoded sequences of the reference genome. In Corona Lite, a **double encoded** sequence is simply the color space translation of a nucleotide sequence where the four colors are represented with the symbols A,C,G and T. This special encoding is required by the SNP/Consensus calling pipeline. The double encoded genome is **not** required for Matching and Pairing.

Corona Lite provides a script named `encodeFasta.py` which can convert a fasta file to a double encoded fasta file. To double encode a sequence, enter the following:

```
% encodeFasta.py -l -n -a fasta.file > de.fasta.file
```

where `fasta.file` is the fasta file you want to encode and `de.fasta.file` will be the double encoded fasta file.

4. Chromosome Map (cmap) File

The `cmap` file identifies the set of files comprising the reference sequences. This file is also used in the Pairing and Matching pipelines. It is a single tab delimited text file where, in the case of a finished genome, each line of the file represents a single sequence entry. The columns are:

1. Sequence ID. This must be a sequential integer, starting from 1.
2. Sequence Name (Strings are OK).
3. Full path to the corresponding fasta file representing this sequence entry. See Step 2 above.
4. Full path to the corresponding double encoded sequence file representing this sequence entry. This file is required only by the SNP pipeline so it can be left blank if you do not intend to apply the SNP pipeline. See Step 3 above.

5. (Optional) Concatenated Genome Sequence File

Some Corona Lite steps will require a single fasta file containing all the sequence entries in the genome. Creating a GFF file and running the Pairing pipeline will require this file. This file can be created from the cmap file using the following command:

```
% concatenate_chr_seqs_cmap.pl -c cmap.file -o newfile
```

where `newfile` is the concatenated Genome sequence file.

6. (Optional) SOLiD Quality Values

A single `.qual` file containing the quality scores corresponding to the reads in the `.csfasta` file in item 1. This file can be used for SOLiD v2 GFF file generation.

8.2 Matching Pipeline Synopsis

Assume the following:

- `reads.csfasta` contains color space SOLiD reads.
- `cmap.file` is present and in the correct format, as described in section 8.1.
- `all.references.fasta` is a fasta file containing all the reference sequences.

1. The first step is to prepare jobs for execution by the scheduler and create a job list for submission. Here we prepare to match `reads.csfasta` (`-csfasta`) to the reference sequences defined in `cmap.file` (`-cmap`) with 25 base pair reads (`-t`), allowing up to 2 base mismatches (`-e`) and reporting up to 10 alignments in the set of reference sequences present in each of the files listed in the `cmap` file (`-z`). The results are written to a directory named `./matching` (`-d`).

```
% matching_large_genomes_cmap_save_script.pl -csfasta  
reads.csfasta -dir ./matching -cmap cmap.file -t 25 -e 2 -z 10
```

2. The next step is to submit the jobs to the scheduler and wait for the jobs to complete:

```
% cd ./matching  
% submit_scripts_to_PBS.pl -j JOB_LIST.txt*
```

* Substitute `submit_scripts_to_LSF.pl` or custom job submission script at this step if LSF or other job scheduling system is used.

3. Examine the results files in the current working directory. The file names will be prefixed based on the `csfasta` containing the SOLiD reads. The whole genome

'match file' will be suffixed with `.csfasta.ma.##.#` (where `##` = readlength and `#` = allowed mismatches)*. In this example this file will be named:
`reads.csfasta.ma.25.2`.

* Note that all the results file names contain the read length and allowed mismatches.

This file contains all the reads and their matching location, if found in the reference genome. When the reference is an assembled genome, each fasta header description line contains the read identifier followed by a comma separated list of genome locations. Genome locations follow the following format:

```
<sequence entry>_<match position>.<number of mismatches>
```

where:

- `sequence entry`: Index of sequence entry containing the match. This corresponds to the `sequence ID` field in the `cmap` file.
- `match position`: Integer representing the position in the reference sequence to which the first color call of the read is matched. A negative number means that the match was made to the reverse complement of the reference sequence. The absolute value of sequence position is the distance of the first base from the 5' end of the reference sequence.
- `number of mismatches`: Number of mismatching bases in match result.

Example genome locations:

- `1_781751.2` indicates a match in sequence entry 1 beginning at position 781751 and proceeding towards the 3' end of the reference with 2 mismatches.
- `1_-320443.0` indicates a match in sequence entry 1 beginning at position 320443 and proceeding towards the 5' end of the reference with 0 mismatches.

4. (Optional) Convert results to GFF2 format. MaToGff is a free tool from the SOLiD Software Development Community for converting matching or pairing results to GFF version 2 format. If you have installed the MaToGff package, you can convert the mapping results to the GFF2 format. Assume the quality file for the run is named `reads.qual`:

```
% MaToGff.sh reads.csfasta.ma.25.2 --convert=unique --clear=1  
--sort --qvs=reads.qual > _gff_temp_
```

```
% AnnotateChanges.sh _gff_temp_ all.references.fasta -b >  
myGFF file
```

See the MaToGff documentation for details on using MaToGff. Pairing results can also be used as input, but use a different shell script: `MatesToGff.sh`.

8.3 Matching Pipeline Output Files

The description below assumes that the reference sequence was an assembled genome, but is applicable to other types of references as well.

1. **.csfasta.ma.tag_length.mismatch_number file:**
The whole genome match file containing all the match results of the SOLiD reads to the reference sequence.
2. **.csfasta.ma.tag_length.miss_match_number.idx file:**
Index file for the match file, to be used in pairing and small indel pipeline.
3. **.unique.csfasta.ma.tag_length.mismatch_number file:**
Same as #1 above, but reports only the subset of unique matches. Unique matches are defined as reads with only a single match within the reference sequence.
4. **.unique.tag_length.mismatch_number.bc.txt file:**
Concatenated base change file, which is the parsed, tab-delimited unique matching result. This file does **not** contain chromosome information.
5. **.unique.tag_length.mismatch_number.coverage.unique.csv file:**
Concatenated unique coverage number, one position per line. This file does **not** contain chromosome information.
6. **.unique.tag_length.mismatch_number.errors.csv file:**
Error profile for the tag generated from unique matching result.
7. **.unique.tag_length.mismatch_number.stats.txt file:**
Whole matching statistics with both unique and non-unique matching information.
8. **AllChromosomes/ directory:**
Redundant match files and unique match files, these files are created for convenience and can be deleted.
9. **chr*/ directories:** There are 7 files in each chr directory:
 - **.csfasta.ma.tag_length.mismatch_number files:**
These are the original individual match files. With the default `-compact` option for the Matching pipeline, there are no chromosome number or no sequence, and reads without hits to the sequence entry are omitted from the corresponding file. This compact file format might be useful for tag counting applications; they are just intermediate matching results and can be deleted.
 - **.unique.csfasta.ma.tag_length.mismatch_number file:**
This is the unique subset of matches from the match file for each chromosome, redundant with item 2 above. It can be deleted.
 - **.unique.tag_length.mismatch_number.bc.txt file:**
Redundant with item 3 above, but this file has sequence entry information because it resides in a chr directory.
 - **.unique.tag_length.mismatch_number.coverage.unique.csv file:**
Redundant with item 4 above, but contains sequence entry information due to its location.
 - **.unique.random.csfasta.ma.tag_length.mismatch_number file:**
Used to calculate unique + random starting points. Random matches are defined as a single random pick of the multiple match positions for a given read.
 - **Two .start_points files:**
Starting point number for the current chr dir.
 - **Split read files:**
If reads were split during matching, there are additional files ending with `.startNum_endNum` which are per set intermediate files. All should be deleted after completion.

10. **JOB_LIST.txt:**

Contains the list of scripts to be executed on the cluster. Found under the scripts directory and contains dependency information. The file is useful for sub-job submission and can be used to rerun part of the analysis job.

11. **The scripts/ directory:**

Contains all the scripts to run the jobs for each step. If you use the LSF submission script provided, this directory also contains the job logs.

After deleting the intermediate files, the matching results can be reduced to around 40-80G for a 25_2 run against human genomes with 150-200M reads.

Items 9 and 10 are common among the Matching, Pairing and SNP pipelines.

8.4 Matching Pipeline Script Usage

```
matching_large_genomes_cmap_save_script.pl [options] --help -csfasta
```

```
-dir -cmap -t -e -p -a -z -reads -incremental -[no]compact -copy_reads  
-tempdir -schema
```

[Options:]

-help Display the help message you are looking at.

[Required]

- -csfasta csfasta file
- -dir Output Directory
- -cmap Chromosome map file
- -t Tag Length
- -e Number of Errors

[Optional]

- -p Pattern for masking positions in a read. Defaults is all 1's.
- -a Count adjacent errors as 1: 0 = no; 1 = valid adjacent errors. Default is 0.
- -z Maximum number of hits reported for each read. Default is 10.
- -reads Number of reads per subset. Default is no subsets.
- -incremental Flag to perform incremental matching by removing reads which are already mapped.
- -compact Flag to output intermediate matching results in compact form. Default is on; use nocompact to turn it off.

- `-copy_reads`
Flag to copy reads file to local scratch directory defined by `-tempdir`. Default is Off.
- `-tempdir`
Space used for scratch files. Default is `/scratch`.
- `-schema`
Schema file for the mapping. If not set, the standard schema is used.

More information:

- `-csfasta`
The reads file, either F3 or R3.
- `-dir`
The output directory.
- `-cmap`
The cmap file explained at the first step.
- `-t, -e`
For repetitive genome, such as human, we recommend 25_2 if the tag length is 25 base pairs.
- `-p`
Default set to all 1's at the length of the tag, but you can mask a position by changing the number to 0 at that position.
- `-z`
To reduce match file size, we recommend setting z to 10 for matching against repetitive genomes.
- `-reads`
Used for splitting input reads file.
- `-compact`
The raw matching results in each chr dir saved without sequence lines. Lines without hits are omitted as well.
- `-tempdir`
Directory used by the mapping software to write scratch files. A drive local to each compute node is preferred. If no such drive is available, choose "." to use the network drives where each matching job is taking place.
- `-schema`
For use with custom schema, not present in the standard installation.

The scripts are saved in the scripts directory under `-dir`. There is a file named `JOB_LIST.txt` in `-dir` which contains all the locations and names for all the scripts. Submit the jobs to the scheduler using `submit_jobs_to_PBS.pl` or `submit_jobs_to_LSF.pl`.

9 Pairing Pipeline Documentation

Corona Lite's Pairing pipeline combines paired reads from a SOLiD mate-pair run. It uses the results of the Matching pipeline as the initial set of candidate positions on the

reference genome. Pairing uses knowledge about the range of insert sizes in the sample to improve the accuracy of read matching and can be used to detect structural variation between the sample and the reference.

9.1 Pairing Pipeline Preparation

To perform pairing the following are required:

1. Reference Genome

A single fasta file containing all the sequences in the reference genome. You may have created this during the Matching pipeline. By default, multi-fasta format is used, such that multiple chromosomes can be specified. To make this from individual chromosome files, enter the following:

```
% concatenate_chr_seqs_cmap.pl -c cmap_file -o outfile
```

Important Note: Using `cat` for this may reorder the chromosomes unexpectedly. Also, each fasta entry in the multi-fasta reference file must have the same index as the corresponding sequence entry in the cmap file.

If you are **not** using the current version of Corona Lite (v4.0r1 or later) you need the following:

2. A pair of Match files

Two match files (file ending in `.csfasta.ma.##.#`) from the Matching pipeline: one from each of the Matching pipeline results of the F3 and R3 tags to the reference genome. These files must be properly sorted to use with the mate-pair pipeline.

By default, the Matching pipeline produces match files that are already properly sorted. If the file has been reordered after creation by the Matching pipeline, you can sort it correctly by entering the following:

```
% #Not needed if files are from current pipeline  
% radsort match.file 6 > sorted.match.file
```

If you perform the matching pipeline using Corona Lite v4.0r1.0 and above, each match file will also have a small index file made (match file name plus the `.idx` extension). These index files are used in the pairing and small-indel finding steps.

9.2 Pairing Pipeline Synopsis

Assume the following:

- `f3.match.file` is the match file containing the F3 reads.
- `r3.match.file` is the match file containing the R3 reads.
- `all.references.fasta` is a fasta file containing all the reference sequences.
- `cmap.file` is the cmap file.

1. If using match files made from Corona Lite **prior to version 4.0r1.0**, you must create an index file for both F3 and R3 tags by entering the following:

```
% #Not needed if files from Corona Lite version 4.0r1.0 or later.
```

```
% panelGrouping f3.match.file,r3.match.file 1 &
```

Note: The number of groups can be any value between 1 and the number of shared panels, typically around 2000.

2. Estimate the range of insert sizes in the library.
Reads are paired while allowing a large range of insert sizes. Reads with strand and correct orientation that match the reference are identified. The output is a distribution of insert sizes which is used to estimate the range of insert sizes in the library.

```
% pairing_by_group.pl -f3 f3.match.file -r3 r3.match.file
--error total 4
--output_dir pairingDir
--reference all.references.fasta
--find_pairing_dist
--comments "Pair first 100 panels"
```

Note: `pairing_by_group.pl` is the new name for the script `pairing_by_panel.pl`. For convenience, both scripts are the same, so the names are interchangeable.

```
% cd pairingDir_0_20000/scripts
% submit_scripts_to_PBS.pl -j JOB_LIST.txt
```

Then, plot `pairingDist.freq.binned` to determine insert size ranges to use.

Note: You can use `--min/max_panel_id` to select panels other than the first 100.

Here, the two separate tags (F3 and R3) are matched to the reference genome spaced between 0 and 20 Kb apart (`-insert_start` and `-insert_end`). The `ref` and `-e` options are not required for this step.

3. The next step is to run pairing on all panels using the insert size distribution boundaries from Step 2:

```
% pairing_by_group.pl -f3 f3.match.file -r3 r3.match.file
--min_insert_size [min]
--max_insert_size [max]
--error total 4
--output_dir pairingDir
--reference all.references.fasta
--comments "Your own comments"
```

Scripts are saved to `./pairingDir_[min]_[max]`. Enter the following to submit sub-jobs to the cluster:

```
% cd pairingDir_[min]_[max]/scripts
% submit_scripts_to_PBS.pl -j JOB_LIST.txt
```

* Substitute `submit_scripts_to_LSF.pl` or custom job submission script at this step if LSF or other job scheduling system is used.

Note: Mate-pair rescue

The `-reference` and `-error total` options are used to rescue paired reads that don't match within the distance allowed by the insert size or within the allowed number of mismatches. These potential reads for rescue will be re-matched to the reference (`-reference`) to search for matches missed by the Matching pipeline, allowing a maximum total of 4 mismatches (`-error total`) in both F3 and R3 tags.

9.3 Pairing Pipeline Output Files

1. F3_R3.mates/F3_R3.mates.non-redundant format:

`F3_R3.mates` contains the pairing information and is the same as in previous versions. `F3_R3.mates.non-redundant` has changed from the previous `F3_R3.unique` in that it is now sorted by the reference sequence entry position of the F3 tag. Previously, the negative strand hits were sorted separately from the positive strand hits.

Both files are tab delimited text files showing the paired reads that were unambiguously matched to the reference genome. Each row represents a pair of reads from a single bead from both the F3 and R3 tags. The columns are:

1. Bead ID.
2. F3 colorspace read.
3. R3 colorspace read.
4. Number of mismatches in the F3 read.
5. Number of mismatches in the R3 read.
6. Total number of mismatches in both reads.
7. Reference sequence ID (sequence entry ID) to which the F3 sequence was matched.
8. Reference sequence ID (sequence entry ID) to which the R3 sequence was matched.
9. 0-based position on the reference where the first base of the F3 read was matched.*
10. 0-based position on the reference where the first base of the R3 read was matched.*
11. Mate-pair category **

* Negative position means that the match was made to the reverse complement of the reference sequence. The absolute value of sequence position is the distance of the first base of the read from the 5' end of the reference sequence.

**The mate-pair category is a three letter code that describes the strand, orientation and distance of reads in the pair. The first letter of the code identifies whether both reads are on the same strand. The second letter of the code refers to the orientation of the reads. The third letter of the code refers to the spacing of the reads. The code AAA identifies mate-pairs with both F3 and R3 reads that are consistent with the reference sequence based on orientation, strand location, and insert size. All other codes identify mate-pairs that do not meet these specifications and may indicate structural differences to the reference.

Table 1. Categories of mate-paired reads

Category	A Reference	B Insertion	C Deletion
AA = Same strand and orientation	AAA	AAB	AAC
AB = Same strand, reverse orientation	ABA	ABB	ABC
BA = Different strands, reads oriented away from each other	BAA	BAB	BAC
BB = Different strands, reads oriented towards each other	BBA	BBB	BBC

C** = reads map to different reference sequence entry

2. **counts.dat:** Very basic statistics for the pairing run.
3. **sortedFragments.gff, sortedMates.gff (Optional)** Convert mate-pair results to GFF2 format.
 MaToGff is a free tool from the SOLiD Software Development Community for converting matching results to GFF version 2 format. If you installed the MaToGff package, you can convert these results to the GFF2 format. See the documentation for MaToGff for more details on usage. You must use MatesToGFF.sh to convert mate-pair output to GFF2 format.


```
% MatesToGff.sh F3_R3.mates --f3qv=f3.qual --r3qv=r3.qual
--ref=all.references.fasta --b --fout=sortedFragments.gff
--mout=sortedMates.gff
```
4. **indel-evidence-list-list.pas:**
Individual indel evidences.
5. **indel-candidates-list.gff and indel-candidates-list.txt:**
Download the new Small Indel Tool from the SOLiD Software tool site for documentation and updates in making these files.
6. **chromosomes directory:**
The Pairing pipeline now automatically runs the multi_chr_pairing_parser.pl script to generate this directory. It can be used as the input for the SNP pipeline.
7. **intermediates directory:**
Contains intermediate results for the pairing and/or small indel runs.
8. **pairingDist.freq:**
The frequencies of the pairing distances.

9. **pairingDist.freq.binned:**
Binned frequency file used to determine insert size.
10. **panelGroupingInfo.txt:**
Shows panel grouping for the pairing run.

9.4 Pairing Pipeline Script Usage

Pairing-by-group.pl

```
pairing_by_group.pl --f3_match_file <F3File> --r3_match_file <R3File>  
--min_insert_size [min] --max_insert_size [max]  
--error_total 4 --output_dir pairingDir --reference human.fa
```

Usage:

[Required]

- `--f3_match_file, -f3` F3 match file
- `--r3_match_file, -r3` R3 match file
- `--output_dir, -dir` Results directory
- `--error_total, -e` Total number of errors to allow in both tags.
- `--reference, -ref` Reference sequence. Leave out to turn off mate-pair rescue.

and either

- `--min_insert_size, -lsz` Minimum insert size for a good mate
- `--max_insert_size, -hsz` Maximum insert size for a good mate

Or

```
--find_pairing_dist
```

Does the first 100 panels using 0 to 20 Kb pairing distance to make a histogram file.

[Info]

- `--comments` Comments that will be stored in the LOG file.
- `--help, -?` Show options.
- `--verbose` Show more verbose help.
- `--man` Show man pages.

[Optional]

- `--ref_format, -rf`
Multiple-entry or single-entry fasta format for reference. Default is multiple.
- `--f3_length, -f3l`
Match length of F3 tag. Use this if the match length is different from the tag length.
- `--r3_length, -r3l`
Match length of R3 tag. Use this if the match length is different from the tag length.

- `--mask_f3, -f3m`
F3 mask. Positions are masked with a 0 in matching.
- `--mask_r3, -r3m`
R3 mask. Positions are masked with a 0 in matching.
- `--min_panel_id, -lp`
Minimum panel id to pair.
- `--max_panel_id, -hp`
Maximum panel id to pair.
- `--max_num_panels, -mnp`
Total number of panels to pair.
- `--unique_only, -uo`
Normal mate-pairs are reported only if they are placed uniquely. Default is Yes.
- `--mismatch_threshold, -mm_th`
Mismatch Threshold. See `--help` for full information.
- `--rand_redundant`
Flag to have truly random selection for redundant F3/R3 mates. Default: Pseudo-random selection is used.

[Optional, Indel Finding]

- `--nofind_indels, -noindel`
Turn off indel finding. Default: On.
- `--indel_parameters, -idp`
Indel parameters [D=11,l=4,i=3,d=3]
 - D is the maximum deletion size to find
 - l is one higher than the maximum deletion size. For example l=4 means that the largest insertion size will be 3.
 - I and d is the number of base pairs from the edge required before an indel can be found for insertion and deletion, respectively.
- `--repeat_limit_indel, -rli`
Repeat limit for indel finding. Default: 10.
- `--error_indel, -ei`
Error total for indel finding. Default: Half of `error_total`.
- `--min_num_evid`
Minimum number of pieces of evidence required to make an indel call. Default: 2.
- `--max_num_evid`
Maximum number of evidences required to make an indel call. Default is -1 (no limit)
- `--skip_mates, -sm`
Skips the production of the mates file to speed indel finding.
- `--indels_in_good_mates, -igm`
Find indels in reads that normally would be excluded for indel finding because the read pair could form a non-gapped alignment ('good mate').

[Optional, Run Environment]

- `--queue_sys`
Queue system used: PBS or `save_script`. Default: `save_script`.
- `--num_jobs`
Number of simultaneous jobs. Default: 10.
- `--scratch/tempdir`
Scratch path, i.e. `/scratch`, `/tmp`, ``pwd`` Default: `/scratch`

Usage:

```
pairing_counts.pl [-dir <dir>] [-outputfile <out>]
```

where `dir` is the directory containing the pairing results, and `out` is the file to print the results to.

9.5 Pairing Pipeline Small Indel Option

Please refer to the Small Indel Tool from the SOLiD™ Software tools site (available in early January, 2009) for updated documentation.

9.5.1 Small Indel Option Output files

- **indel-evidence-list.pas**
Read by read evidences for indels. Note: This file will be unchanged until the next release of Corona Lite. (Updates to the AB Small Indel Tool are downstream of this file.)
- **indel-candidate-list.txt**
The two coordinates represent the range of the start position of the indel event. (The end is not displayed). For both insertions and deletions, this coordinate is immediately before the indel event.

The range comes from ambiguity in determining the precise indel location (for example, in a polynucleotide tract, AAA-, AA-A, -AAA are all possible indel locations) and from the combination of evidences.

There are two ranges listed in this file: **tight** and **loose**. A **tight** range is where all the pieces of evidence contain this range. This could be the **null** set which would be displayed as a blank column. A **loose** range is all possible ranges from all the available evidence.

- **indel-candidate-list.gff**
Contains the information in `.gff v3` format. Note that this does not contain as much information as the `.txt` file above.

For insertions, the coordinates are at the base position immediately before the insertion event. For deletions, the coordinates are reported as a range of positions in the reference missing from the sample.

Note: Users of Corona Lite v4.0r2.0 or earlier require the installation of the Small Indel Tool on the SOLiD™ Software tools site to patch issues involved with the making of these candidate files.

9.5.2 Small Indel Option Post-processing

These commands are automatically generated when using `pairing_by_group.pl`, (see `scripts/PAIR_POST_INDEL_1.sh`) but are useful if you wish to perform some analysis without rerunning the whole pipeline.

Removal of highest insertion size

To remove artifacts due to the cut-off constraint in the pairing algorithm, the highest insertion size is removed as part of the post processing step. The script `indel-remove-sizes.pl` performs this task:

```
for num in 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
do
  indel-remove-sizes.pl 0 3 < intermediates/indel.dat.$num.pas >>
  indel-sorted-list.pas.tmp
done
```

Note, if you used a different number of jobs, then change the 9 value above to `<# jobs - 1>`.

Sorting and adding run identifiers

To sort the results by reference sequence entry position, and then add identifiers for the run, and read length, enter the following:

```
sort -n intermediates/indel-sorted-list.pas.tmp | sed
"s/_R3,/_25.2_Lib1_1_25.2_CLARA_20071113_1B,/g"
> indel-evidence-list.pas
```

Here, 25.2 is the read length and number of mismatches in matching. Lib1_1 is the library indicator, and CLARA_20071113_1B is the run name. The algorithm parses using the underscore character (`_`), so be sure to reproduce the format found above.

Combining evidence to form candidates

Note: The following candidate files are more easily created using the AB Small Indel Tool.

Multiple strands of evidence are typically required when calling candidates. Parameters used here may change if you have low (i.e. 2x) or high coverages (i.e. 50x).

```
mpindel_sum.pl indel-evidence-list.pas 2 -1 200 1 MP >
intermediates/indel-sorted-list.sum
```

The 2 and -1 settings are the default for the min and max number of non-redundant evidence to consider for a candidate. Set these in the main `pairing_by_group.pl` script using `--min_num_evidence` and `--max_num_evidence`.

The 200 setting is how much information is stored for each candidate. If you have high coverage, such as 200x, you may want to edit this setting to something higher in `pairing_by_group.pl`.

The next parameter, 1, is a flag setting for conservative grouping. Using 1 is for conservative grouping which has 5 base pairs maximum between consecutive pieces of evidence and 0 is for the higher of 15 or (7*indel size) maximum between pieces of evidence. For both, if there are already 5 evidences, then additional ones will be grouped only if it is 2 or fewer base pairs from the last one. This is to handle possible Short Tandem Repeats (STR's).

Filtering and Creating Candidate gff file

See the AB Small Indel Tool documentation for details in creating these.

9.5.3 Other Considerations

Skipping recreation of mates file

If you would like to skip recreating the mates file, use the `--skip_mates` option. This will significantly reduce computational time.

Low/High Coverage

The parameters have been optimized for moderate coverages (6-15X). For high coverage situations, such as for enriched or microbe, consider increasing `--min_num_evidence` to a higher value than the default of 2.

For low coverage situations, consider setting `--max_num_evid` to some value around 2-3 times the average coverage, to avoid enriching for repeat regions (and thus possible false positives.)

Finding indels in “good” mates

Only missing or bad pairs are considered for indel finding because the likelihood is low that it's an indel if a good pair could instead be made.

Use `--indels_in_good_mates` to consider indels even if it a non-indel pairing can be found for a particular bead.

9.5.4 Indel Alignment Information

The indel evidence files contain alignment information that is represented in the following format:

```
>78_158_886_R3,1_555384.21.2(18:18_20)[T32312...2]|1_556514.0
```

>78_158_886 is the bead ID. (This is a pairing, so R3 in this case is irrelevant). The rescued pairing (R3 matching, F3) is separated by |. The indel can be found in either the R3 or F3 tag. In this case, it is found in the R3 tag, specifically,

1_555384.21.2 translates to a chromosome 1 hit at position 555384, and 21 is the match length, thus the insertion size is

```
read_length - match_length - 1
```

Negative values are deletions, and positive, insertions. Here, the read length was 25, so the indel is an insertion of size 3. The read position was found was position 18, zero

based, while 18-20 is the range of other possible positions. Finally, [T32312...2] is the read sequence.

Note the positions above are all 0-based. This contrasts with the positions reported in the candidate files have been adjusted to be 1-based.

10 SNP Pipeline Documentation

Corona Lite's SNP pipeline is the process of identifying single nucleotide variation between the reference and the samples.

This pipeline uses the results of the Matching pipeline for single read experiments or the results of the Pairing pipeline for mate-pair experiments.

10.1 SNP Pipeline Preparation

To apply the SNP pipeline, you need the following:

1. Matching results from the Matching pipeline or mate-pair results from the Pairing pipeline.
 - If using results from a single tag fragment SOLiD run, the unique match file from the Matching pipeline (the file ending in `unique.csfasta.ma.##.#`) is used for the SNP pipeline.
 - If using results from a mate-pair SOLiD run, the file `F3_R3.mates.non_redundant` from the Pairing pipeline is used.

Note: Starting with this version of Corona Lite, the file `F3_R3.mates.unique` was renamed `F3_R3.mates.non_redundant`. The Pairing pipeline automatically processes this file into a directory named `chromosomes` under pairing results that can be used for the SNP pipeline.

2. A Chromosome Map (cmap) file.

This should be the same cmap file used in the preceding run(s) of the Matching pipeline. Note that for this pipeline, the cmap file must reference double encoded references sequences.

10.2 SNP Pipeline Synopsis

Assume the following:

- `unique.match.file` is the unique match file from a run of the Matching pipeline.
- `F3_R3.mates.non_redundant` is the non redundant mates (previously named `F3_R3.mates.unique`) file from a run of the Pairing pipeline.

1. Prepare files for analysis.

The preparation step depends on the type of data being used for the SNP pipeline.

Single tag results:

Here `unique.match.file (-f)` is split into separate files by chromosome in the directory `chr_unique (-c)`:

```
% split_by_chromosome.pl -f unique.match.file -c chr_unique
```

Mate-pair results:

Here `F3_R3.mates.non_redundant (-mates)` is split into separate files by reference sequence entry and `mate_pair_category` (the three letter code from the Pairing pipeline that describes the position and orientation of the aligned paired reads) in the directory `chromosomes (-o_dir)`. `cmap.file` is the `cmap.file`.

```
% multi_chr_pairing_parser.pl -mates  
F3_R3.mates.non_redundant -o_dir chromosomes
```

Note: The current version of the Corona Lite Pairing pipeline automatically processes `F3_R3.mates.non_redundant` into the `chromosomes` directory, so users may skip this step and just use the `chromosomes` directory under pairing results for the SNP pipeline.

2. Prepare consensus prep for execution by the scheduler. Syntax for this step depends on whether fragment tags or mate-pair results are being used.

Fragment tag runs:

The SNP analysis is conducted on files prepared in Step 1 (`-match_dir`) using the reference sequences in `cmap.file (-cmap)` with results written to a directory named `runs (-o_dir)`.

Parameters used in the Matching pipeline are passed to the script: Match length of 25 (`-ml_f3`) and 2 mismatches (`-e_f3`). The script is also informed about the types of probes used in the SOLiD run (`-12only`).* **

```
% consensus_prep_and_wrapper_cmap_save_script.pl  
-match_dir chr_unique -cmap cmap.file -o_dir runs  
-ml_f3 25 -e_f3 2 -12only -preonly
```

```
% submit_scripts_to_PBS.pl -j JOBS.txt ***
```

* The `-12only` switch informs the script that the query positions of the SOLiD probes are at positions 1 and 2. With SOLiD 2.0, all runs use these '12 probes'. Hence, the `-12only` switch should always be set when using SOLiD 2.0 data.

** If multiple SOLiD sequencing runs are to be analyzed, use the `-preonly` option for `consensus_prep_and_wrapper_cmap_save_script.pl`. This will shorten the processing time, but SNPs will not be called for individual runs. We now recommend using this option even for single run analysis, because the `consensus_wrapper` script for multiple runs has new features not implemented in the single step wrapper script.

*** Substitute `submit_scripts_to_LSF.pl` or custom job submission script at this step if LSF or other job scheduling system is used.

Mate-pair runs:

Here the SNP analysis is conducted on files prepared in Step 1 (`-mates_dir`) using the reference sequences in `cmap.file` (`-cmap`) with results written to a directory named `runs`.

Parameters used in the Matching pipeline are passed to the script: Match lengths of 25 (`-ml_f3` and `-ml_r3`) and 2 mismatches (`-e_f3` `-e_r3`). The range of insert sizes used in the Pairing pipeline are 1500 (`-insert_start`) to 2600 (`-insert_end`). The script is also informed about the types of probes used in the SOLiD run (`-12only`).* **

```
% consensus_prep_and_wrapper_cmap_save_script.pl
-mates_dir chromosomes -cmap cmap.file -o_dir runs
-ml_f3 25 -ml_r3 25 -e_f3 2 -e_r3 2
-insert_start 1500 -insert_end 2600 -12only -preponly
% submit_scripts_to_PBS.pl -j JOBS.txt ***
```

* The `-12only` switch informs the script that the query positions of the SOLiD probes are at positions 1 and 2. With SOLiD 2.0, all runs use these '12 probes'. Hence, the `-12only` switch should always be set when using SOLiD 2.0 data.

** If multiple SOLiD sequencing runs are to be analyzed, use the `-preponly` option for `consensus_prep_and_wrapper_cmap_save_script.pl`. This will shorten the processing time, but SNPs will not be called for individual runs. We now recommend using this option even for single run analysis, because the `consensus_wrapper` script for multiple runs has new features not implemented in the single step wrapper script.

*** Substitute `submit_scripts_to_LSF.pl` or custom job submission script at this step if LSF or other job scheduling system is used.

3. Examine the results.

The results of Step 2 are written to the directory named with the `-o_dir` option. This directory will contain one subdirectory per chromosome..

Continue with Step 4 to analyze with results from multiple samples if you used the `-preponly` option.

4. Prepare files for analysis with a single or multiple samples.

Repeat Steps 1 through 3 on results from each SOLiD run, placing each result in a separate directory. To reduce processing time, run `consensus_prep_and_wrapper_cmap_save_script.pl` with the `-preponly` switch. Running with this switch enabled will not call SNPs with individual samples.

5. Prepare the Consensus Prep file (CP file).

Create a directory to contain results and make it the current working directory.

The Consensus Prep file (CP file is a tab delimited text file with the following columns:

- Full file path to the results from `consensus_prep_and_wrapper_cmap_save_script.pl`. This is the directory specified with `-o_dir` option.
- Result Type: F3, R3 or `paired` for F3, R3 or Mate-paired results respectively.
- Read Length.

6. Perform consensus wrapper analysis.

```
% cw_multi_wrap_general.pl -cp cp.file
-output_dir consensus_wrapper
-cmap cmap.file
```

where `cp.file` is the consensus prep file and `consensus_wrapper` is a directory to contain results.

```
% cd consensus_wrapper
% submit_scripts_to_PBS.pl -j JOB_LIST.txt
```

Note: Substitute `submit_scripts_to_LSF.pl` or custom job submission script at this step if LSF or other job scheduling system is used.

7. Examine results from multiple sample analysis.

The results of Step 6 are written to the directory set with the `-output_dir` option. Subdirectories will be created for each chromosome. The principal result file in these subdirectories is named `snps.txt`.

10.3 SNP Pipeline Output Files

SNP call jobs are divided by chromosome numbers, with each chromosome being a job. Within each `chr` directory, there are a lot of files, but most are intermediate files and should be deleted after successful job run. Files worth saving are:

- **snps.txt:**
SNP result sorted by chromosome coordinates.
- **snps_sorted.txt:**
SNP results sorted by confidence.
- **bp_consensus_confirmed_sequence_with_Ns.fasta:**
The consensus sequence in base space.
- **F3_R3_good_mates_annotated.gff:**
Keep this file if you want to display the tags in the SOLiD Alignment Browser.

If you run the multi-sample/multi-library SNP pipeline, the `consensus_prep` directories can be removed after all jobs are finished. However, if you plan to add more sample/library data to the SNP callings later, save these directories.

Here is the description of the `snps.txt` and `snps_sorted.txt`:

Column 1

The number of reads covering the position with 2 color space calls. Note that reads only partially covering the base with the last color space call of the read are **not** counted.

Column 2

The reference base at that position.

Column 3

The consensus base at that position. **Note:** IUPAC codes are used for heterozygotes.

Columns 4 – 7

Refers to the reference base and columns.

Columns 8 – 11

Refers to the consensus base score = the weighted score assigned to the reference or consensus base based on all the reads that cover it with 2 color space calls. These are the weighted scores for each of the 16 types of dibase combinations sum to 1.

- `conf` = The average confidence of each read that was used to generate the weighted score of the reference or consensus base.
- `Single` or `F3*` = Number of single tag or F3 reads that agree with the reference or consensus/number of single tag or F3 unique molecules (start points) that agree with the reference or consensus.
- `Paired` or `R3*` = Number of paired or R3 reads that agree with the reference or consensus/number of paired or R3 unique molecules (start points) that agree with the reference or consensus.

Column 12

Base space coordinate of the position (1-based). `snps.txt` is sorted by this column.

`snps_sorted.txt` attempts to sort `snps.txt` in order of confidence of the call:

1. The first sort is on the total number of unique molecules supporting the consensus base, the sum of the second numbers in columns 10 and 11.
2. The second sort is on the coverage, column 1.
3. The third sort is on the confidence of the consensus base, column 9.
4. The fourth sort is on the coordinate, column 12.

* Depending on which version of the variation pipeline is run, columns 6-7 and 10-11 refer to either single and paired tags or F3 and R3 tags as indicated by the column headers.

10.4 SNP Pipeline Script Usage

```
split_by_chromosome.pl [options] -f *match file w/multiple sequences*  
-c *chromosome directory* -cmap *chromosome map file- OPTIONAL*
```

```
multi_chr_pairing_parser.pl [options] -mates *multi-chr .mates file* -  
o_dir *output directory*
```

```
consensus_prep_and_wrapper_cmap_save_script.pl [options] [-match_dir  
OR -mates_dir] -ml_f3 *match length, F3* -ml_r3 *match length, R3* -  
e_f3 *errors_f3* -e_r3 *errors_r3* -insert_start *insert start: leave  
undefined if frag run* -insert_end *-insert end: leave undefined if  
frag run* -cmap *chromosome map file* -o_dir *output directory*
```

[Optional]

- -primer_f3 Defaults to T.
- -primer_r3 Defaults to G.
- -pbs_input Defaults to PBS_General.input.
- -12only Only use this if run is 1,2-only.
- -preponly Only run consensus_prep.pl.

You call `consensus_prep_and_wrapper_cmap_save_script.pl` either after matching or pairing is complete. If you want, you can call it on just the F3 tag and just the R3 tag alone. The options are:

- -match_dir or -mates_dir
Specify the location of either the output of pairing (the chromosomes directory in the first step) or the output of matching (the chr_unique folder).

Next, there are a series of arguments for F3 and R3. If you're running it on F3 only or R3 only, just leave off all the F3 or R3 arguments.

- -m_f3, -m_r3
The length for which the reads were matched.
- -t_f3, -t_r3
This is the actual length of the reads in the file, these arguments are optional. If the matching was performed on a shorter length than the tags, provide these values here or else the program will fail.
- -e_f3 and -e_r3
The number of mismatches allowed in the F3 and R3 tags.
- -insert_start, -insert_end
For pairing only. Leave these blank if the run is fragments.
- -cmap
This is the same cmap file using in matching and mate-pairing. The 4th column of this file is a doubleEncoded reference sequence file, which is not required for matching and mate-pairing but is required for the SNP calling pipeline.

All output will go into this base directory with "chr" subdirectories.

- `-primer_f3` and `-primer_r3`
You probably don't want to use these arguments, but you can change the primer bases if you want to.
- `-pbs_input`
This original program from `corona_large_genome` uses the utility `PBS_general.pl` to submit jobs to the PBS cluster using the `PBSJob` module. It will create a tab-delimited file of working directories, commands to execute, and how many processors per node to use for running each of the programs. A modified version of the script, `PBS_general_save_script.pl`, is used here to create shell scripts instead of submitting jobs. You can change the name of the file (`PBS_General.input`) with this argument flag.
- `-12only`
For the 1,2-only probes. Without this option, the default is for mixed 1,2 and 4,5 probe.
- `-preponly`
This is now recommended even for single sample runs. For one or more samples, run the consensus prep using this option, then do the separate `consensus_wrapper` step. For example:

Fragment run:

```
% consensus_prep_and_wrapper_cmap_save_script.pl
-match_dir path/chr_unique -ml_f3 25 -e_f3 2 -cmap
/share/reference/genomes/Hs_B36.cmap -o_dir runs -12only -
preponly
```

or

```
consensus_prep_and_wrapper_cmap_save_script.pl -match_dir
path/chr_unique -ml_r3 25 -e_r3 2 -cmap
/share/reference/genomes/Hs_B36.cmap -o_dir runs -12only -
preponly
```

Mate Pair run:

```
% consensus_prep_and_wrapper_cmap_save_script.pl
-mates_dir chromosomes -ml_f3 25 -ml_r3 25 -e_f3 2 -e_r3 2
-insert_start 400 -insert_end 1450 -cmap
/share/reference/genomes/Hs_B36.cmap -o_dir runs -12only -
preponly
```

Notes:

This step will generate a `JOB_LIST.txt` file. You need to run the jobs specified by this file before proceeding to the consensus wrapper step.

You need to run `consensus_prep` for each SOLiD run/library, and the results will be created in separate directories. The next step (consensus wrapper) will combine each `consensus_prep` result to call SNPs on the combined data.

```
cw_multi_wrap_general.pl -cp *CP Path File* -output_dir *output
dir* -cmap *cmap file* -pt *Use Error by Position and Type
Matrix*
```

This is the script to generate commands to run `consensus_wrapper_multiple_samples.pl` on each chromosome.

- `-cp`
A tab-delimited file describing the location, run type, read length of each `consensus_prep` results directory. For example:

```
/foo/bar/consensus_prep/library1/runs/      F3    50
/foo/bar/consensus_prep/library2/runs/      paired 25
/foo/bar/consensus_prep/library3/runs/      F3    35
/foo/bar/consensus_prep/library4/runs/      paired 25
```

- `-cmap`
The same cmap file used in matching and pairing steps
- `-output_dir`
Directory which contains SNP results subdirectories, one for each sequence entry described in the cmap file. Also contains the scripts directory and `JOB_LIST.txt` file for submitting scripts to job scheduling systems.
- `-pt`
Use error by position and type matrix for SNP calls, getting slightly different SNP calls.

11 Sample Data

A small data set used for testing Corona Lite pipelines can be downloaded from <http://solidsoftwaretools.com/qf/project/corona/>. It contains:

- 10 panels of a DH10B mate-pair run data.
- A set of 4 fake chromosomes produced by splitting the DH10B reference genome.
- A command file that contains step by step commands to run all the pipelines in Corona Lite.
- Expected results that were done previously.

More information can be found in the README file that comes with the sample download.

12 Software Change History

v4.0r2.0

- Additional processing of indel results. Added filtering and gff.
- Included the mapreads source code in order to allow re-compilation for those who require it; It can be found in the \$CORONAROOT/src directory. See installation instructions for details.

v4.0r1.1:

- Minor changes to the Make file for mapreads source recompile.
- Updated documentation files.

v4.0r1.0:

Matching changes:

- -iub support in matching, post matching statistics needs fixing, but raw matching file is OK.
- Support for gzipped reads file, the -csfasta option needs to point to the gzipped reads file.
- -copy_reads to local scratch when specified, significant speed increase when matching large reads files with many lines in the schema file by reducing network I/O.

Pairing changes:

- Complete change to the wrapper, no pre-splitting of match file is required. It also performs small indel finding.

SNP pipeline changes:

- Added the -pt option.

v4.0:

Multiple changes to the Matching pipeline:

- Valid_Adjacent=1 schemas can be automatically selected.
- Added many new schemas to support the VA=1 option.
- Masking for VA=1 is disabled, to be supported in the future.
- Fixed a small bug in valid adjacent matching stats.
- Changes to ignore counting Ns in basechange files.

Removal of Apache License information from Perl code, as Corona Lite is now open source.

v0.32:

Preparation for SOLiD software community website release:

- Removing v2 GFF tool, which will be released separately. Adding instructions to the Corona Lite directory structure.
- Remove corona_large genome information from documentation.
- Providing better explanation for job submission/running process.

v0.31R2:

- Added a script `submit_scripts_to_PBS.pl` which uses `JOB_LISTS.txt` to submit jobs to PBS. With this function added, `corona_large_genome` is no longer needed.
- Updated schemas in `corona_lite/etc/schemas` for use in matchings.
- Included v2 gff conversion tool version 2.03. The command wrapper scripts have changed, see `ReadMe_gffv2.03.txt` under `corona_lite__v0.31R2` for more details.

v0.31R1:

- Fixed a bug in mapreads when using `-compact` and masking positions. Also included the `compress` binary in the bin directory.

v0.31:

- Major change to the core mapper to reduce the scratch file and intermediate raw matching file sizes. For a 25_2 run with `z=10`, the size of scratch files are about 1/10 of previous sizes. For example, matching of 230M reads against human chr1 uses < 1.7G scratch versus 10-20G previously. There is now an `-tempdir` option to set where scratch files should go. The sum of the raw matching file sizes reduced from the size of the reads file (5-10G) x number_of_chromosomes x 1.2 to roughly 5-6X of the reads file size, which is similar to the merged match file size. Due to the reduced I/O load, plus some internal improvement, the mapper runs roughly 25% faster.
- The process to merge/concatenate individual chromosome raw matching file now uses much faster C-based code, which takes roughly 1/4 of the time.
- Starting point number calculation moved to the `post_matching_by_chr` jobs. This cuts run time for large reads files on the `post_matching_final` step to about half.
- Added `-schema` option for Matching pipeline to use custom search schema. Fixed several bugs in the count adjacent error as 1 option (`-a`).
- Added `-tempdir` option so users can conveniently set the location of scratch files used in matching and post-matching jobs.
- Added a `CORONA_LITE.RESULTS` file to describe files generated by each of the three main pipelines.

Additional files in the bin dir:

- `post_matching_by_chr_map_fast.pl` replaces
- `post_matching_by_chr_map.pl` `matching_stats_human_fast.pl`
- replaces `matching_stats_human_memory_fix.pl`
- `matching_unique_and_random.pl` replaces `matching_unique.pl`
- `start_point_from_match_file_fast.pl` replaces
- `start_point_from_match_file.pl` `merge_map` `compress`

v0.30:

- Updated SNP pipeline capable of using multiple samples as input, also added the new GFF v2 converter that can export corrected base sequences of reads.
- `README.SNPS` is updated to describe the new SNP wrapper.

Additional scripts:

- bin dir: `snp_list_sort_multiple_samples.pl`
- `snp_list_multiple_samples.pl` `snp_counts.pl`
- `snp_confirmation_multiple_samples_yoruban.pl`
- `snp_confirmation_multiple_samples.pl`

- concatenate_snp_probs.pl
- concatenate_consensus_statistics.pl
- cw_multi_wrap_general.pl
- consensus_wrapper_multiple_samples.pl gffv2_module.sh

v0.23:

- Remove -run_limit from README.SNPS, this flag is not needed for Corona Lite because commands are saved as scripts.
- matching_stats_human_memory_fix.pl bug, will fail after getting unique start points from the sort -u -T . | wc -l cmd due to failure to remove preceding spaces from the result of the shell command (only happens if the result is less than 7 digits).
- Change in pairing_by_panel_save_script.pl to implement when -ref is missing, it will simply ignore pairing rescue instead of failing.
- Change in pairing_by_panel_save_script.pl to implement the new -mismatch_threshold flag, see README.PAIRING for more information.
- Add information in README.PAIRING about concatenated genome used in the pairing rescue process.
- Set JOB_LIST.txt file to be parallel to scripts dir, whose default location now goes to the parent dir of output_dir. Added encodeFasta.py to the collection for generating double-encoded genome.

v0.22:

Changes to make a unified RELEASE_NOTE for both packages.

v0.21:

Minor corrections to the README.SNPS file.

v0.2:

March 2008. All three pipelines (matching, pairing and SNP-calling) are included. Many bug fixes, all tested on internal LSF farm.

v0.1:

January 2008. The first adaptation of the corona_large_genome to run in non-PBS environment. Only the genome matching included. Installed at two non-AB sites with LSF environment.

Applied Biosystems, and AB (Design) are registered trademarks and SOLiD is a trademark of Applied Biosystems or its affiliates in the US and/or certain other countries.

© 2009 Applied Biosystems. All rights reserved.