

Paired-End Adapter Finder
v15.4.14

Generated by Doxygen 1.8.9.1

Wed May 6 2015 17:21:48

Contents

1	Paired-End Adapter Finder v15.4.14	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	CS Class Reference	5
3.1.1	Detailed Description	5
3.1.2	Member Function Documentation	6
3.1.2.1	calc_phred	6
3.1.2.2	checkConfidence	6
3.1.2.3	cs	6
3.1.2.4	print_cs	6
3.1.2.5	print_nucCount_phred	6
3.1.3	Member Data Documentation	6
3.1.3.1	adapterLength	6
3.1.3.2	adapterPos	6
3.1.3.3	Confidence	7
3.1.3.4	consensus	7
3.1.3.5	nucleotidecount	7
3.1.3.6	nuclist	7
3.1.3.7	phred	7
3.2	Input Class Reference	7
3.2.1	Detailed Description	7
3.2.2	Member Function Documentation	7
3.2.2.1	complementInput	7
3.3	NW Class Reference	8
3.3.1	Detailed Description	8
3.3.2	Member Function Documentation	9
3.3.2.1	clear	9
3.3.2.2	dpm_init	9
3.3.2.3	max	9

3.3.2.4	nw	9
3.3.2.5	nw_align	9
3.3.2.6	print_al	10
3.3.2.7	print_matrix	10
3.3.2.8	print_traceback	10
3.3.2.9	verifyPercentage	10
3.3.3	Member Data Documentation	10
3.3.3.1	colmax	10
3.3.3.2	count	11
3.3.3.3	F	11
3.3.3.4	Fx	11
3.3.3.5	Fy	11
3.3.3.6	percentage	11
3.3.3.7	rowmax	11
3.3.3.8	traceback	11
3.3.3.9	tracebackscore	11
Index		13

Chapter 1

Paired-End Adapter Finder v15.4.14

The Paired-End Adapter Finder constructs two consensus adapter sequences from Reads 1 and Reads 2 of paired-end sequencing by using the Needleman-Wunsh algorithm to align the two reads and determining which region of the sequence is actually an adapter. The user has to input two fastq format files, corresponding to Read 1 and Read 2, and the result will be the consensus adapter sequences for both the Adapters in Read 1 and Read 2 respectively of the paired-end sequencing.

Author

Rayan Gan

Date

April 2015

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

CS	The CS (p. 5) Class corresponds to the Consensus Sequence which is used obtain the adapter sequences from the two reads	5
Input	The Input (p. 7) Class transforms the sequences to be used in the Adapter Sequencer	7
NW	The NW (p. 8) Class corresponds to the Needleman-Wunsch algorithm which is used to align the two reads	8

Chapter 3

Class Documentation

3.1 CS Class Reference

The **CS** (p. 5) Class corresponds to the Consensus Sequence which is used obtain the adapter sequences from the two reads.

```
#include <CS.h>
```

Public Member Functions

- void **calc_phred** ()
- void **checkConfidence** (double conf, int &confTrue, int adapLenCount)
- void **cs** (string seq_1, int max)
- void **print_cs** (int opt)
- void **print_nucCount_phred** ()

Private Attributes

- int **adapterLength**
- int **adapterPos**
- int **Confidence** [20]
- char **consensus** [20]
- int **nucleotidecount** [4][20]
- double **phred** [4][20]

Static Private Attributes

- static const char **nuclist** [4] = {'A','C','G','T'}

3.1.1 Detailed Description

The **CS** (p. 5) Class corresponds to the Consensus Sequence which is used obtain the adapter sequences from the two reads.

The Consensus Sequence for both Read 1 and Read 2 adapters are determined based on the traceback of the dynamic programming algorithm. The Consensus Sequence is searched based on how well Read 1 and Read 2 align to each other. The ends of Read 1 and Read 2 which do not align form adapters and are added to the nucleotide count. The final Consensus Sequence is based on the nucleotide count.

Author

Rayan Gan

Date

April 2015

3.1.2 Member Function Documentation**3.1.2.1 void CS::calc_phred ()**

Calculates the Phred scores of each nucleotide in the nucleotide count

3.1.2.2 void CS::checkConfidence (double *conf*, int & *confTrue*, int *adapLenCount*)

Checks the confidence level of the consensus sequences

Parameters

<i>conf</i>	Confidence level
<i>confTrue</i>	Does the sequence meet the confidence level?

3.1.2.3 void CS::cs (string *seq_1*, int *max*)Main function in **CS** (p.5) class which performs the algorithm to find the consensus sequences**Parameters**

<i>seq_1</i>	Read 1 in the first fastq file
<i>max</i>	Length at which alignment is the best

3.1.2.4 void CS::print_cs (int *opt* = 0)

Prints out the Consensus Sequence

Parameters

<i>opt</i>	Option to choose if Read 1 or Read 2 (0 - Read 1, 1 - Read 2)
------------	---

3.1.2.5 void CS::print_nucCount_phred ()

Prints out the nucleotide count and Phred scores

3.1.3 Member Data Documentation**3.1.3.1 int CS::adapterLength [private]**

Length of adapter.

3.1.3.2 int CS::adapterPos [private]

Position of adapter.

3.1.3.3 `int CS::Confidence[20] [private]`

Confidence of adapter.

3.1.3.4 `char CS::consensus[20] [private]`

Consensus sequence of adapter.

3.1.3.5 `int CS::nucleotidecount[4][20] [private]`

2D-array of number of nucleotides at each position in the adapter.

3.1.3.6 `const char CS::nuclist = {'A','C','G','T'} [static], [private]`

List of nucleotides.

3.1.3.7 `double CS::phred[4][20] [private]`

2D-array of Phred score of each nucleotide in the adapter.

The documentation for this class was generated from the following files:

- `/export/home/ryan/Documents/PEAdapterFinder/CS.h`
- `/export/home/ryan/Documents/PEAdapterFinder/CS.cpp`

3.2 Input Class Reference

The **Input** (p. 7) Class transforms the sequences to be used in the Adapter Sequencer.

Public Member Functions

- `int complementInput (string &)`

3.2.1 Detailed Description

The **Input** (p. 7) Class transforms the sequences to be used in the Adapter Sequencer.

The **Input** (p. 7) Class is responsible to reverse complement Read 2 before dynamic programming is done.

Author

Rayan Gan

Date

April 2015

3.2.2 Member Function Documentation

3.2.2.1 `int Input::complementInput (string & seq)`

Reverse complement the input for sequences from the second input file.

Parameters

<i>seq</i>	The sequence to be complemented
------------	---------------------------------

The documentation for this class was generated from the following file:

- /export/home/ryan/Documents/PEAdapterFinder/Input.cpp

3.3 NW Class Reference

The **NW** (p. 8) Class corresponds to the Needleman-Wunsch algorithm which is used to align the two reads.

```
#include <NW.h>
```

Public Member Functions

- `int nw (string seq_1, string seq_2, string &seq_1_al, string &seq_2_al, int debug)`

Public Attributes

- `int colmax`
- `int count`
- `int ** F`
- `int Fx`
- `int Fy`
- `double percentage`
- `int rowmax`
- `char ** traceback`
- `char * tracebackscore`

Private Member Functions

- `void clear ()`
- `void dpm_init (int **F, char **traceback, int L1, int L2, int d)`
- `int max (int f1, int f2, int f3, char &ptr)`
- `int nw_align (int **F, char **traceback, string seq_1, string seq_2, string &seq_1_al, string &seq_2_al, int d)`
- `void print_al (string &seq_1_al, string &seq_2_al)`
- `void print_matrix (int **F, string seq_1, string seq_2)`
- `void print_traceback (char **traceback, string seq_1, string seq_2)`
- `void verifyPercentage (string seq_1_al, string seq_2_al)`

3.3.1 Detailed Description

The **NW** (p. 8) Class corresponds to the Needleman-Wunsch algorithm which is used to align the two reads.

The Needleman-Wunsch algorithm is implemented thorough dynamic programming. Read 1 is aligned against the reverse complement of Read 2 to obtain the best alignment. The nucleotides at the right end of Read 1 and upper end of Read 2 form the adapter sequences for Adapter 1 and Adapter 2 respectively.

Author

Ryan Gan

Date

April 2015

3.3.2 Member Function Documentation

3.3.2.1 void NW::clear () [private]

Clears the dynamic programming and traceback matrices

3.3.2.2 void NW::dpm_init (int ** *F*, char ** *traceback*, int *L1*, int *L2*, int *d*) [private]

Initialize the dynamic programming matrix and traceback matrix

Parameters

<i>F</i>	The dynamic programming matrix
<i>traceback</i>	The traceback matrix
<i>L1</i>	The length of Read 1
<i>L2</i>	The length of Read 2
<i>d</i>	Gap penalty

3.3.2.3 int NW::max (int *f1*, int *f2*, int *f3*, char & *ptr*) [private]

Compares the upper, diagonal and left values to determine traceback

Parameters

<i>f1</i>	The value from Upper Cell
<i>f2</i>	The value from Diagonal Cell
<i>f3</i>	The value from Left Cell
<i>ptr</i>	Returns the symbol for the traceback matrix

3.3.2.4 int NW::nw (string *seq_1*, string *seq_2*, string & *seq_1_al*, string & *seq_2_al*, int *debug*)

Main function in the **NW** (p.8) class which calls other functions to perform dynamic programming on Reads 1 and Reads 2

Parameters

<i>seq_1</i>	Read 1 in the first fastq file
<i>seq_2</i>	Read 2 in the second fastq file
<i>seq_1_al</i>	Read 1 aligned to Read 2
<i>seq_2_al</i>	Read 2 aligned to Read 1

3.3.2.5 int NW::nw_align (int ** *F*, char ** *traceback*, string *seq_1*, string *seq_2*, string & *seq_1_al*, string & *seq_2_al*, int *d*) [private]

Runs the Needleman-Wunsch Alignment to get the best alignment

Parameters

<i>F</i>	The dynamic programming matrix
<i>traceback</i>	The traceback matrix
<i>seq_1</i>	Read 1 in the first fastq file
<i>seq_2</i>	Read 2 in the second fastq file
<i>seq_1_al</i>	Read 1 aligned to Read 2
<i>seq_2_al</i>	Read 2 aligned to Read 1
<i>d</i>	Gap penalty

3.3.2.6 void NW::print_al (string & *seq_1_al*, string & *seq_2_al*) [private]

Prints out the aligned sequences

Parameters

<i>seq_1_al</i>	Read 1 aligned to Read 2
<i>seq_2_al</i>	Read 2 aligned to Read 1

3.3.2.7 void NW::print_matrix (int ** *F*, string *seq_1*, string *seq_2*) [private]

Prints out the dynamic programming matrix

Parameters

<i>F</i>	The dynamic programming matrix
<i>seq_1</i>	Read 1 in the first fastq file
<i>seq_2</i>	Read 2 in the second fastq file

3.3.2.8 void NW::print_traceback (char ** *traceback*, string *seq_1*, string *seq_2*) [private]

Prints out the traceback matrix

Parameters

<i>traceback</i>	The traceback matrix
<i>seq_1</i>	Read 1 in the first fastq file
<i>seq_2</i>	Read 2 in the second fastq file

3.3.2.9 void NW::verifyPercentage (string *seq_1_al*, string *seq_2_al*) [private]

Determines the traceback percentage

Parameters

<i>seq_1_al</i>	Read 1 aligned to Read 2
<i>seq_2_al</i>	Read 2 aligned to Read 1

3.3.3 Member Data Documentation

3.3.3.1 int NW::colmax

Traceback value at first column of dynamic programming.

3.3.3.2 int NW::count

Counts the number of runs of **NW** (p. 8).

3.3.3.3 int NW::F**

Stores the dynamic programming values.

3.3.3.4 int NW::Fx

Keeps track of the length of sequence 1.

3.3.3.5 int NW::Fy

Keeps track of the length of sequence 2.

3.3.3.6 double NW::percentage

Percentage of matches in best traceback.

3.3.3.7 int NW::rowmax

Largest value at last row of dynamic programming.

3.3.3.8 char NW::traceback**

Stores the traceback characters.

3.3.3.9 char* NW::tracebackscore

Stores the best traceback characters.

The documentation for this class was generated from the following files:

- /export/home/ryan/Documents/PEAdapterFinder/NW.h
- /export/home/ryan/Documents/PEAdapterFinder/NW.cpp

Index

adapterLength
 CS, 6
adapterPos
 CS, 6
CS, 5
 adapterLength, 6
 adapterPos, 6
 calc_phred, 6
 checkConfidence, 6
 Confidence, 6
 consensus, 7
 cs, 6
 nucleotidecount, 7
 nuclist, 7
 phred, 7
 print_cs, 6
 print_nucCount_phred, 6
calc_phred
 CS, 6
checkConfidence
 CS, 6
clear
 NW, 9
colmax
 NW, 10
complementInput
 Input, 7
Confidence
 CS, 6
consensus
 CS, 7
count
 NW, 10
cs
 CS, 6
dpm_init
 NW, 9
F
 NW, 11
Fx
 NW, 11
Fy
 NW, 11
Input, 7
 complementInput, 7
max
 NW, 9
NW, 8
 clear, 9
 colmax, 10
 count, 10
 dpm_init, 9
 F, 11
 Fx, 11
 Fy, 11
 max, 9
 nw, 9
 nw_align, 9
 percentage, 11
 print_al, 10
 print_matrix, 10
 print_traceback, 10
 rowmax, 11
 traceback, 11
 tracebackscore, 11
 verifyPercentage, 10
nucleotidecount
 CS, 7
nuclist
 CS, 7
nw
 NW, 9
nw_align
 NW, 9
percentage
 NW, 11
phred
 CS, 7
print_al
 NW, 10
print_cs
 CS, 6
print_matrix
 NW, 10
print_nucCount_phred
 CS, 6
print_traceback
 NW, 10
rowmax
 NW, 11
traceback
 NW, 11

tracebackscore
NW, 11

verifyPercentage
NW, 10