SeqDPT

Generated by Doxygen 1.8.3.1

Thu May 23 2013 12:09:08

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 seqan::AdapterScoringMatrix Struct Reference

Struct used to define a new custom scoring matrix.

`#include <adapterTrimming.h>`

4.1.1 Detailed Description

Struct used to define a new custom scoring matrix.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

4.2 AdapterTrimmingParams Struct Reference

Struct holding all adapter trimming parameters.

Public Attributes

- bool paired
- bool noAdapter
- bool run
- Dna5QString adapter1
- Dna5QString adapter2
- Mode mode
- MatchMode mmode
- AdapterTrimmingStats stats

4.2.1 Detailed Description

Struct holding all adapter trimming parameters.

Struct holding the parameters needed for adapter trimming.

4.2.2 Member Data Documentation

4.2.2.1 Dna5QString AdapterTrimmingParams::adapter1

Holds the first adapter.

4.2.2.2 Dna5QString AdapterTrimmingParams::adapter2

Holds the second adapter

4.2.2.3 MatchMode AdapterTrimmingParams::mmode

Enum, indicating which trimming mode is used. Necessary for casting mode.

4.2.2.4 Mode AdapterTrimmingParams::mode

Hold the Mode-object which determines which trimming mode shall be used.

4.2.2.5 bool AdapterTrimmingParams::noAdapter

TRUE if no adapter file is provided.

4.2.2.6 bool AdapterTrimmingParams::paired

TRUE if paired-end data is used.

4.2.2.7 bool AdapterTrimmingParams::run

TRUE if the adapter trimming shall be run.

4.2.2.8 AdapterTrimmingStats AdapterTrimmingParams::stats

Holds interesting numbers about trimmed adapters.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/SeqDPT.cpp

4.3 AdapterTrimmingStats Struct Reference

Struct to hold information about certain adapter trimming statistics.

`#include <adapterTrimming.h>`

Public Member Functions

- void **clear** ()

Public Attributes

- unsigned **a1count**
- unsigned a2count
- unsigned overlapSum
- unsigned **minOverlap**
- unsigned maxOverlap

4.3.1 Detailed Description

Struct to hold information about certain adapter trimming statistics.

4.3.2 Member Data Documentation

4.3.2.1 unsigned AdapterTrimmingStats::a2count

The number of adapters present in forward and backward reads.

4.3.2.2 unsigned AdapterTrimmingStats::maxOverlap

Minimum and maximum overlap between adapter and reads.

4.3.2.3 unsigned AdapterTrimmingStats::overlapSum

The sum of the removed adapter bases. Later used to calculate mean adapter size.

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

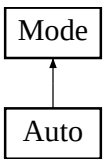
- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

4.4 Auto Struct Reference

Tagging struct for the automatic match algorithm.

`#include <adapterTrimming.h>`

Inheritance diagram for Auto:



4.4.1 Detailed Description

Tagging struct for the automatic match algorithm.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

4.5 BWA Struct Reference

The tagging structure for the BWA trimming algorithm.

`#include <readTrimming.h>`

4.5.1 Detailed Description

The tagging structure for the BWA trimming algorithm.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/readTrimming.h

4.6 DemultiplexingParams Struct Reference

Struct holding all demultiplexing parameters.

Public Attributes

- seqan::String< char > barcodeFile
- seqan::StringSet < seqan::String< seqan::Dna > > barcodes
- seqan::StringSet < seqan::String< char > > barcodeIds
- seqan::String< char > multiplexFile
- seqan::StringSet < seqan::String< seqan::Dna5Q > > multiplex
- bool approximate
- bool hardClip
- bool run
- bool runx

4.6.1 Detailed Description

Struct holding all demultiplexing parameters.

4.6.2 Member Data Documentation

4.6.2.1 bool DemultiplexingParams::approximate

TRUE if approximate search shall be used

4.6.2.2 seqan::String<char> DemultiplexingParams::barcodeFile

Holds the path to the barcode-file

4.6.2.3 seqan::StringSet<seqan::String<char> > DemultiplexingParams::barcodeIds

Holds the StringSet of barcode-IDs

4.6.2.4 seqan::StringSet<seqan::String<seqan::Dna> > DemultiplexingParams::barcodes

Holds the StringSet of barcodes

4.6.2.5 bool DemultiplexingParams::hardClip

TRUE if hardClip shall be used

4.6.2.6 seqan::StringSet<seqan::String<seqan::Dna5Q> > DemultiplexingParams::multiplex

Holds the StringSet of multiplex barcodes

4.6.2.7 seqan::String<char> DemultiplexingParams::multiplexFile

Holds the path to the multiplex-file

4.6.2.8 bool DemultiplexingParams::run

TRUE if demultiplexing shall run

4.6.2.9 bool DemultiplexingParams::runx

TRUE if multiplex demultiplexing shall run

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/SeqDPT.cpp

4.7 Dna5QAdapter Struct Reference

This structure wraps a sequence with dedicated Dna5 and quality strings so we can use it with our trimming function.

`#include <readTrimming.h>`

Public Member Functions

- **Dna5QAdapter** (seqan::String< seqan::Dna5 > &s, seqan::CharString &q)

Public Attributes

- seqan::String< seqan::Dna5 > & seq
- seqan::CharString & qual

4.7.1 Detailed Description

This structure wraps a sequence with dedicated Dna5 and quality strings so we can use it with our trimming function.

4.7.2 Member Data Documentation

4.7.2.1 seqan::CharString& Dna5QAdapter::qual

The quality string of the sequence.

4.7.2.2 seqan::String<seqan::Dna5>& Dna5QAdapter::seq

The Dna5 string of the sequence.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/readTrimming.h

4.8 Mean Struct Reference

The tagging structure for the window trimming algorithm.

`#include <readTrimming.h>`

Public Member Functions

- **Mean** (unsigned w)

Public Attributes

- unsigned window

4.8.1 Detailed Description

The tagging structure for the window trimming algorithm.

4.8.2 Member Data Documentation

4.8.2.1 unsigned Mean::window

The window size used for quality calculations.

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

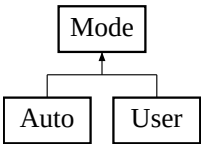
- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/readTrimming.h

4.9 Mode Struct Reference

A struct encapsulating information about the match algorithm.

`#include <adapterTrimming.h>`

Inheritance diagram for Mode:



4.9.1 Detailed Description

A struct encapsulating information about the match algorithm.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

4.10 OutputStreams Class Reference

Class that dynamically manages output streams that write out sets of sequences.

Public Member Functions

- OutputStreams (seqan::CharString base, seqan::SeqIOFileFormat_::Type format, bool compress)

  Constructor for the OutputStreams object. Prepares the file extension which will be used for all streams created by this object and saves a base directory path.

- template<typename TKey , typename TMap >
  bool exists (TKey &key, TMap &map)

  Checks whether a key exists in a std::map.

- void addStream (seqan::CharString fileName, int id)

  Add a new output streams to the collection of streams.

- void addStreams (seqan::CharString fileName1, seqan::CharString fileName2, int id)

  Add a new output streams to the collection of streams.

- template<typename TMap , typename TNames >
  void updateStreams (TMap &map, TNames &names, bool pair)

  This method takes a vector of numbers and checks if these numbers are already associated with a stream. If not, a new stream is added and the opened file is named according to the list of names. One or two files are created.

- template<typename TIds , typename TSeqs , typename TMap , typename TNames >
  void writeSeqs (TIds &ids, TSeqs &seqs, TMap &map, TNames &names)

  Writes the sets of ids and sequences to their corresponding files.

- template<typename TIds , typename TSeqs , typename TMap , typename TNames >
  void writeSeqs (TIds &ids1, TSeqs &seqs1, TIds &ids2, TSeqs &seqs2, TMap &map, TNames &names)

  Writes the sets of ids and sequences to their corresponding files. Overload for writing paired-end sequence sets.

- ~OutputStreams ()

  Destructor of the object holding the output streams. Needed to destroy the streams properly after they have been created with new.

4.10.1 Detailed Description

Class that dynamically manages output streams that write out sets of sequences.

4.10.2 Member Function Documentation

4.10.2.1 void OutputStreams::addStream (seqan::CharString *fileName,* int *id*) [inline]

Add a new output streams to the collection of streams.

Parameters

fileName	The name of the first file that will be created.
id	The associated id that will be used to identify the stream.

4.10.2.2 void OutputStreams::addStreams (seqan::CharString *fileName1,* seqan::CharString *fileName2,* int *id*) [inline]

Add a new output streams to the collection of streams.

Parameters

fileName1	The name of the first file that will be created.
fileName2	The name of the second file that will be created.
id	The associated id that will be used to identify the pair of streams.

4.10.2.3 template<typename TKey , typename TMap > bool OutputStreams::exists (TKey & *key,* TMap & *map*) [inline]

Checks whether a key exists in a std::map.

Returns

True if the key exists, false otherwise.

4.10.2.4 template<typename TMap , typename TNames > void OutputStreams::updateStreams (TMap & *map,* TNames & *names,* bool *pair*) [inline]

This method takes a vector of numbers and checks if these numbers are already associated with a stream. If not, a new stream is added and the opened file is named according to the list of names. One or two files are created.

Parameters

map	The list of IDs for which the existence of a file should be checked.
names	The list of names to be used when creating new files.
pair	Indicates whether one or two (a pair of files) should be created.

4.10.2.5 template<typename TIds , typename TSeqs , typename TMap , typename TNames > void OutputStreams::writeSeqs (TIds & *ids,* TSeqs & *seqs,* TMap & *map,* TNames & *names*) [inline]

Writes the sets of ids and sequences to their corresponding files.

Parameters

ids	A list of sets of sequence IDs. (As returned by readRecord etc.)
seqs	A list of sets of sequences.
map	A mapping of the sets of sequences to their corresponding output streams.
names	Names to be used when creating new streams.

4.10.2.6 template<typename TIds , typename TSeqs , typename TMap , typename TNames > void OutputStreams::writeSeqs (TIds & *ids1*, TSeqs & *seqs1*, TIds & *ids2*, TSeqs & *seqs2*, TMap & *map*, TNames & *names*) [inline]

Writes the sets of ids and sequences to their corresponding files. Overload for writing paired-end sequence sets.

Parameters

ids1	A list of sets of forward sequence IDs. (As returned by readRecord etc.)
seqs1	A list of sets of forward sequences.
ids2	A list of sets of backward sequence IDs. (As returned by readRecord etc.)
seqs2	A list of sets of backward sequences.
map	A mapping of the sets of sequences to their corresponding output streams.
names	Names to be used when creating new streams.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/SeqDPT.cpp

4.11 QualityTrimmingParams Struct Reference

Struct holding all quality trimming parameters.

Public Attributes

- TrimmingMode trim_mode
- int cutoff
- int min_length
- bool run
- QualityTrimmingStats stats

4.11.1 Detailed Description

Struct holding all quality trimming parameters.

Struct holding the parameters needed for quality trimming.

4.11.2 Member Data Documentation

4.11.2.1 int QualityTrimmingParams::cutoff

Holds the cutoff score.

4.11.2.2 int QualityTrimmingParams::min_length

Holds the minam length of a sequnce after trimming.

4.11.2.3 bool QualityTrimmingParams::run

TRUE if the quality trimming shall run.

4.11.2.4 QualityTrimmingStats QualityTrimmingParams::stats

Holds interesting numbers about quality trimming.

4.11.2.5 TrimmingMode QualityTrimmingParams::trim_mode

Holds the ::TrimmingAlgorithm-object which determines which algrotihm shall be used.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/SeqDPT.cpp

4.12 QualityTrimmingStats Struct Reference

Public Member Functions

- void **clear** ()

Public Attributes

- unsigned **dropped_1**
- unsigned **dropped_2**

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/readTrimming.h

4.13 seqan::ScoringMatrixData_< int, Dna5, AdapterScoringMatrix > Struct Template Reference

Struct containing data for the seqan::AdapterScoringMatrix custom scoring matrix. Matches score 1, mismatches -1 and matches against N with 0.

`#include <adapterTrimming.h>`

Public Types

- enum { **VALUE_SIZE** = ValueSize<Dna5>::VALUE, **TAB_SIZE** = VALUE_SIZE * VALUE_SIZE }

Static Public Member Functions

- static int const * **getData** ()

4.13.1 Detailed Description

template<>struct seqan::ScoringMatrixData_< int, Dna5, AdapterScoringMatrix >

Struct containing data for the seqan::AdapterScoringMatrix custom scoring matrix. Matches score 1, mismatches -1 and matches against N with 0.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

4.14 STRING_REVERSE_COMPLEMENT< TValue > Struct Template Reference

A metafunction which constructs a ModifiedString type for an Alphabet.

`#include <adapterTrimming.h>`

Public Types

- typedef ModifiedString
  < ModifiedString< String
  < TValue >, ModView
  < FunctorComplement< TValue >
  > >, ModReverse > **Type**

4.14.1 Detailed Description

template<class TValue>struct STRING_REVERSE_COMPLEMENT< TValue >

A metafunction which constructs a ModifiedString type for an Alphabet.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

4.15 Tail Struct Reference

The tagging structure for the Tail trimming algorithm.

`#include <readTrimming.h>`

4.15.1 Detailed Description

The tagging structure for the Tail trimming algorithm.

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

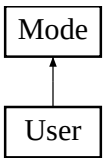
- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/readTrimming.h

4.16 User Struct Reference

Tagging struct representing the the match algorithm working with values supplied by the user. Saves those values as members.

`#include <adapterTrimming.h>`

Inheritance diagram for User:



Public Member Functions

- **User** (int m, int e)

Public Attributes

- int min_length
- int errors

4.16.1 Detailed Description

Tagging struct representing the the match algorithm working with values supplied by the user. Saves those values as members.

4.16.2 Member Data Documentation

4.16.2.1 int User::errors

The maximum number of errors we allow.

4.16.2.2 int User::min_length

The minimum length of the overlap.

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

- D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h

Chapter 5

File Documentation

5.1 D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/adapterTrimming.h File Reference

Contains the functions for adapter trimming.

```
#include <seqan/align.h>
#include <seqan/find.h>
```

Classes

- struct seqan::AdapterScoringMatrix

  Struct used to define a new custom scoring matrix.
- struct seqan::ScoringMatrixData_< int, Dna5, AdapterScoringMatrix >

  Struct containing data for the seqan::AdapterScoringMatrix custom scoring matrix. Matches score 1, mismatches -1 and matches against N with 0.
- struct Mode

  A struct encapsulating information about the match algorithm.
- struct Auto

  Tagging struct for the automatic match algorithm.
- struct User

  Tagging struct representing the the match algorithm working with values supplied by the user. Saves those values as members.
- struct AdapterTrimmingStats

  Struct to hold information about certain adapter trimming statistics.
- struct STRING_REVERSE_COMPLEMENT< TValue >

  A metafunction which constructs a ModifiedString type for an Alphabet.

Typedefs

- typedef Score< int,
  ScoreMatrix< Dna5,
  AdapterScoringMatrix > > **TScore**

Functions

- template<typename TSeq1 , typename TSeq2 , bool TTop, bool TLeft, bool TRight, bool TBottom>
  seqan::Pair< unsigned,
  seqan::Align< TSeq1 > > alignPair (TSeq1 &seq1, TSeq2 &seq2, const seqan::AlignConfig< TTop, TLeft, TRight, TBottom > &config, bool band=false)

  Align two sequences, returning the score and align object of the alignment.
- template<typename TRow >
  unsigned countTotalGaps (TRow &row)

  Returns the total number of gaps in a row object.
- template<typename TAlign >
  unsigned getOverlap (TAlign &align)

  Determines the overlap between two (free-shift) aligned gapless sequences.
- template<typename TAlign >
  unsigned getInsertSize (TAlign &align)

  Given an alignment with two overlapping (forward and reverse) sequences, this function determines the actual size of the insert they cover.
- template<typename TSeq >
  unsigned stripPair (TSeq &seq1, TSeq &seq2)

  Removes adapter contamination from paired-end reads.
- template<typename TSeq >
  unsigned stripPair (TSeq &seq1, TSeq &adapter1, TSeq &seq2, TSeq &adapter2)

  Removes adapter contamination from paired-end reads with adapter information.
- template<typename TSeq >
  unsigned stripPairBatch (seqan::StringSet< TSeq > &set1, seqan::StringSet< TSeq > &set2, AdapterTrimmingStats &stats)

  Removes adapter contamination from paired-end reads.
- template<typename TSeq , typename TAdapter >
  seqan::Pair< unsigned,
  seqan::Align< TSeq > > alignAdapter (TSeq &seq, TAdapter &adapter)

  Aligns a sequence to an adapter.
- bool isMatch (int overlap, int mismatches, const Auto &)

  Checks if a overlap of an alignment is accepted, based on mismatches and the length of the overlap.
- bool isMatch (int overlap, int mismatches, const User &userOptions)

  Checks if a overlap of an alignment is accepted, based on mismatches and the length of the overlap.
- template<typename TSeq , typename TAdapter , typename TSpec >
  unsigned stripAdapter (TSeq &seq, TAdapter &adapter, TSpec &spec)

  Remove adapter sequence from a sequence.
- template<typename TSeq , typename TAdapter , typename TSpec >
  unsigned stripAdapterBatch (seqan::StringSet< TSeq > &set, TAdapter &adapter, TSpec const &spec, AdapterTrimmingStats &stats, bool reverse=false)

  Remove adapter sequence from a set of sequences.
- template<typename TSeq , typename TSpec >
  unsigned stripReverseAdapterBatch (seqan::StringSet< TSeq > &set, TSeq &adapter, TSpec const &spec, AdapterTrimmingStats &stats)

  Simple interface to align the reverse complement of an adapter to a batch of sequences. to a set of reads and remove significant matches.

5.1.1 Detailed Description

Contains the functions for adapter trimming.

5.1.2 Function Documentation

5.1.2.1 template<typename TSeq , typename TAdapter > seqan::Pair<unsigned, seqan::Align<TSeq> > alignAdapter (TSeq & *seq,* TAdapter & *adapter*)

Aligns a sequence to an adapter.

Parameters

seq	The sequence which adapter contamination shall be removed.
adapter	The adapter that might contaminate the sequence.

Returns

A pair of unsigned ints containing the score of the alignment and the alignment object.

5.1.2.2 template<typename TSeq1 , typename TSeq2 , bool TTop, bool TLeft, bool TRight, bool TBottom> seqan::Pair<unsigned, seqan::Align<TSeq1> > alignPair (TSeq1 & *seq1,* TSeq2 & *seq2,* const seqan::AlignConfig< TTop, TLeft, TRight, TBottom > & *config,* bool *band* = `false`)

Align two sequences, returning the score and align object of the alignment.

Parameters

seq1	The first sequence of the alignment.
seq2	The second sequence of the alignment.
config	The alignment configuration used by the Seqan's globalAlignment method.
band	Whether the alignment's lower diagonal should be banded at -2. (Two diagonals below the main diagonal.) Useful to speed up computation for 5'-3' overlap alignments.

Remarks

The main intended uses are AlignConfig<true, false, true, false> with band = true and AlignConfig<true, true, true, true> with band = false.
We use a custom scoring scheme which scores 1 for match, -1 for mismatch, 0 for match with N.

Returns

A pair containing the score of the alignment and the aling object.

5.1.2.3 template<typename TRow > unsigned countTotalGaps (TRow & *row*)

Returns the total number of gaps in a row object.

Parameters

row	The gapped row object used by seqan alignments.

5.1.2.4 template<typename TAlign > unsigned getInsertSize (TAlign & *align*)

Given an alignment with two overlapping (forward and reverse) sequences, this function determines the actual size of the insert they cover.

Parameters

align	An alignment object with two aligned overlapping sequences.

Precondition

The aligned sequences must overlap and not contain any internal gaps.

Remarks

This method can reconstruct the insert perfectly, provided the alignment object represents the real alignment. There can be small discrepancies if indels occurred.

Returns

The insert size that was determined from the overlap alignment.

5.1.2.5 template<typename TAlign > unsigned getOverlap (TAlign & *align*)

Determines the overlap between two (free-shift) aligned gapless sequences.

Parameters

align	An alignment object with two aligned overlapping sequences.

Precondition

The aligned sequences must not contain any internal gaps.

Returns

The number of overlapping positions.

5.1.2.6 bool isMatch (int *overlap,* int *mismatches,* const Auto &)

Checks if a overlap of an alignment is accepted, based on mismatches and the length of the overlap.

Parameters

overlap	The number of overlapping positions in the overlap alignment.
mismatches	The number of allowed mismatches in the overlapping region.

Remarks

This method automatically uses a very simple heuristic to determine matches.

Returns

Bool indicating if the alignment is significant.

5.1.2.7 bool isMatch (int *overlap,* int *mismatches,* const User & *userOptions*)

Checks if a overlap of an alignment is accepted, based on mismatches and the length of the overlap.

Parameters

overlap	The number of overlapping positions in the overlap alignment.
mismatches	The number of allowed mismatches in the overlapping region.
userOptions	Parameters specifying requirements for the overlap.

Remarks

This overload is used to let the user specify match requirements.

Returns

Bool indicating if the alignment is significant.

5.1.2.8 template<typename TSeq , typename TAdapter , typename TSpec > unsigned stripAdapter (TSeq & *seq,* TAdapter & *adapter,* TSpec & *spec*)

Remove adapter sequence from a sequence.

Parameters

seq	The sequence whose adapter contamination should be removed.
adapter	The adapter sequence that might contaminate the sequence.
spec	The match algorithm that decides whether a match was significant.

Returns

The overlap of the sequence with the adapter.

5.1.2.9 template<typename TSeq , typename TAdapter , typename TSpec > unsigned stripAdapterBatch (seqan::StringSet< TSeq > & *set,* TAdapter & *adapter,* TSpec const & *spec,* **AdapterTrimmingStats** & *stats,* bool *reverse* = `false`)

Remove adapter sequence from a set of sequences.

Parameters

set	A StringSet of sequences whose adapter contaminations should be removed.
adapter	The adapter sequence that might contaminate the sequences.
spec	The match algorithm that decides whether a match was significant.
reverse	Indicates whether we align forward or reverse sequences against an adapter. Exists mainly to enable correct writing of the appropriate fields when gathering statistics.

Remarks

This method simply applies stripAdapter to all sequences in **set**.
This method can operate concurrently with OpenMP.

Returns

The number of sequences that had some bases removed.

5.1.2.10 template<typename TSeq > unsigned stripPair (TSeq & *seq1,* TSeq & *seq2*)

Removes adapter contamination from paired-end reads.

Parameters

seq1	The forward read.
seq2	The backward read.

Remarks

This method is very accurate and does not need any knowledge of the specific adapter sequences contaminating the reads.

Returns

The determined actual insert size or 0 if the sequences don't overlap.

Remarks

The number of trimmed bases is len(seq) - insert, if the sequence was greater than the insert size.

5.1.2.11 template<typename TSeq > unsigned stripPair (TSeq & *seq1,* TSeq & *adapter1,* TSeq & *seq2,* TSeq & *adapter2*)

Removes adapter contamination from paired-end reads with adapter information.

Parameters

seq1	The forward read.
adapter1	The adapter that contaminates the forward read.
seq2	The backward read.
adapter2	The adapter whose reverse complement contaminates the backward read.

Remarks

This method can be less accurate than the overload which doesn't use any adapter sequence information, since it is more constrained in how it tries to overlap the sequences.

Returns

The determined actual insert size or 0 if the sequences don't overlap.

Remarks

The number of trimmed bases is len(seq) - insert, if the sequence was greater than the insert size.

5.1.2.12 template<typename TSeq > unsigned stripPairBatch (seqan::StringSet< TSeq > & *set1,* seqan::StringSet< TSeq > & *set2,* AdapterTrimmingStats & *stats*)

Removes adapter contamination from paired-end reads.

Parameters

seq1	The set of forward reads.
seq2	The set of backward reads.

Remarks

This method is very accurate and does not need any knowledge of the specific adapter sequences contaminating the reads.
This method can operate concurrently with OpenMP.

Returns

The number of sequences that had some bases removed.

5.1.2.13 template<typename TSeq , typename TSpec > unsigned stripReverseAdapterBatch (seqan::StringSet< TSeq > & *set,* TSeq & *adapter,* TSpec const & *spec,* AdapterTrimmingStats & *stats*)

Simple interface to align the reverse complement of an adapter to a batch of sequences. to a set of reads and remove significant matches.

Parameters

set	A StringSet of sequences whose adapter contaminations should be removed.
adapter	The adapter sequence whose reverse complement might contaminate the sequences.
spec	The match algorithm that decides whether a match was significant.

Remarks

This interface can be used to detect contamination in reverse reads of paired-end reads. Those reads might read into the reverse complement of an adapter.

Returns

The number of sequences that had some bases removed.

5.2 D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/demultiplex.h File Reference

Contains the functions for barcode demultiplexing.

```
#include <seqan/find.h>
#include <seqan/basic.h>
#include <seqan/sequence.h>
#include <seqan/index.h>
#include <seqan/seq_io.h>
```

Typedefs

- typedef String< Dna5Q > **TAlphabet**

Functions

- template<typename TSeqs , typename TIds , typename TBarcodes >
  bool check (TSeqs &seqs, TSeqs &seqsRev, TIds &ids, TBarcodes &barcodes)

  Function for controlling the sequences and deleting too short ones. Also checks the barcodes.
- template<typename TSeqs , typename TIds , typename TBarcodes >
  bool check (TSeqs &seqs, TIds &ids, TBarcodes &barcodes)

Overload of check(TSeqs& seqs, TSeqs& seqsRev, TIds& ids, TBarcodes& barcodes) for single-end data.

- template<typename TSeqs >
  TSeqs getPrefix (TSeqs &seqs, unsigned len)

  Function for extracting the prefices of all given sequences.
- template<typename TPrefix , typename TFinder >
  int findExactIndex (const TPrefix &prefix, TFinder &finder)

  Function for exact search for one prefix in the barcode indices.
- template<typename TPrefices , typename TFinder >
  std::vector< int > findAllExactIndex (const TPrefices &prefices, TFinder &finder)

  Function for performing findAllExactIndex on all given prefices and barcodes.
- template<typename TBarcodes >
  std::vector< Pattern< String
  < Dna5Q >, DPSearch
  < SimpleScore > > > makePatterns (TBarcodes &barcodes)

  Function for creating a vector of patterns used by findApprox and findAllApprox.
- template<typename TPrefix , typename TPatterns >
  int findApprox (TPrefix &prefix, TPatterns &patterns)

  Function for approximate search for one prefix in all barcodes.
- template<typename TPrefices , typename TPatterns >
  std::vector< int > findAllApprox (const TPrefices &prefices, TPatterns &patterns)

  Function for performing findApprox on all given prefices and barcodes.
- template<typename TSeqs >
  void clipBarcodes (TSeqs &seqs, const std::vector< int > &matches, unsigned len)

  Function for clipping the barcodes from the sequences if they could be matched beforehand.
- template<typename TSeqs >
  void clipBarcodes (TSeqs &seqs, int len)

  Overload of clipBarcodes<seqs, matches, len> if the first len bases of every sequence shall be clipped regardless of the matching(or not matching) barcodes.
- void resizeGroups (std::vector< std::vector< int > > &groups)

  Function for resizing the groups of the vector produced by group and used by it.
- template<typename TMatches , typename TBarcodes >
  std::vector< std::vector< int > > group (const TMatches &matches, const TBarcodes &barcodes)

  Function for sorting the matched and clipped sequences into one vector.
- template<typename TBarcodes , typename TMultiplex , typename TFinder >
  std::vector< std::vector< int > > DoAll (TMultiplex &multiplex, TBarcodes &barcodes, TFinder &esaFinder)

  Function for performing all demultiplexing operations. Using exact search and multiplex barcodes.
- template<typename TBarcodes , typename TMultiplex >
  std::vector< std::vector< int > > DoAll (TMultiplex &multiplex, TBarcodes &barcodes)

  Overload of ::DoAll(TBarcodes& multiplex, TBarcodes& barcodes, TFinder& esaFinder). Using approximate search and multiplex barcodes.
- template<typename TSeqs , typename TBarcodes , typename TFinder >
  std::vector< std::vector< int > > DoAll (TSeqs &seqs, TBarcodes &barcodes, TFinder &esaFinder, bool hardClip)

  Overload of ::DoAll(TBarcodes& multiplex, TBarcodes& barcodes, TFinder& esaFinder). Using exact search and inline barcodes.
- template<typename TSeqs , typename TBarcodes >
  std::vector< std::vector< int > > DoAll (TSeqs &seqs, TBarcodes &barcodes, bool hardClip)

  Overload of ::DoAll(TBarcodes& multiplex, TBarcodes& barcodes, TFinder& esaFinder). Using approximate search and inline barcodes.
- template<typename TSeqs , typename TIds >
  void buildSets (TSeqs &seqs, TSeqs &seqsRev, TIds &ids, TIds &idsRev, const std::vector< std::vector< int > > &groups, std::vector< TSeqs > &gSeqs, std::vector< TSeqs > &gSeqsRev, std::vector< TIds > &gIds, std::vector< TIds > &gIdsRev)

  Function for creating Vectors of StringSets from the vector produced by DoAll, or group.

- template<typename TSeqs , typename TIds >
  void buildSets (TSeqs &seqs, TIds &ids, const std::vector< std::vector< int > > &groups, std::vector< TSeqs > &gSeqs, std::vector< TIds > &gIds)

  Overload of buildSets(TSeqs& seqs, TSeqs& seqsRev, TIds& ids, TIds& idsRev, const std::vector<std::vector<int> >& groups, std::vector<TSeqs>& gSeqs, std::vector<TSeqs>& gSeqsRev, std::vector<TIds>& gIds, std::vector<TIds>& gIdsRev) for single-end data.
- template<typename TgSeqs , typename TgIds , typename TBarcodeIds >
  int writeGroups (TgSeqs &gSeqs, TgSeqs &gSeqsRev, TgIds &gIds, TBarcodeIds &barcodeIds, std::vector< std::vector< int > > &groups, String< char > &path)

  Function for Writing out the demultiplexed sequences.
- template<typename TgSeqs , typename TgIds , typename TBarcodeIds , typename Tgroups >
  int writeGroups (TgSeqs &gSeqs, TgIds &gIds, TBarcodeIds &barcodeIds, Tgroups &groups, String< char > &path)

  Function for Writing out the demultiplexed sequences.

5.2.1 Detailed Description

Contains the functions for barcode demultiplexing.

5.2.2 Function Documentation

5.2.2.1 template<typename TSeqs , typename TIds > void buildSets (TSeqs & *seqs,* TSeqs & *seqsRev,* TIds & *ids,* TIds & *idsRev,* const std::vector< std::vector< int > > & *groups,* std::vector< TSeqs > & *gSeqs,* std::vector< TSeqs > & *gSeqsRev,* std::vector< TIds > & *gIds,* std::vector< TIds > & *gIdsRev*)

Function for creating Vectors of StringSets from the vector produced by DoAll, or group.

Parameters

seqs	StringSet of forward reads.
seqsRev	StringSet backward reads.
ids	StringSet of IDs of forward reads.
idsRev	StringSet of IDs of backward reads.
groups	Two-dimensional vector of integers representing the indices of the sequences. Produced by DoAll, or group.
gSeq	Vector of StringSets to be filled with the forward reads.
gSeqRev	Vector of StringSets to be filled with the backward reads.
gIds	Vector of StringSets to be filled with the IDs of forwad reads.
gIds	Vector of StringSets to be filled with the IDs of backward reads.

Remarks

the given StringSets are deleted.

5.2.2.2 template<typename TSeqs , typename TIds > void buildSets (TSeqs & *seqs,* TIds & *ids,* const std::vector< std::vector< int > > & *groups,* std::vector< TSeqs > & *gSeqs,* std::vector< TIds > & *gIds*)

Overload of buildSets(TSeqs& seqs, TSeqs& seqsRev, TIds& ids, TIds& idsRev, const std::vector<std::vector<int> >& groups, std::vector<TSeqs>& gSeqs, std::vector<TSeqs>& gSeqsRev, std::vector<TIds>& gIds, std::vector<TIds>& gIdsRev) for single-end data.

Parameters

seqs	StringSet of forward reads.
ids	StringSet of IDs.
groups	Two-dimensional vector of integers representing the indices of the sequences. Produced by DoAll, or group.
gSeq	Vector of StringSets to be filled with the forward reads.
gIds	Vector of StringSets to be filled with the IDs.

Remarks

the given StringSets are deleted.

5.2.2.3 template<typename TSeqs , typename TIds , typename TBarcodes > bool check (TSeqs & *seqs,* TSeqs & *seqsRev,* TIds & *ids,* TBarcodes & *barcodes*)

Function for controlling the sequences and deleting too short ones. Also checks the barcodes.

The Function deletes the ID and sequence from the given StringSet if the sequence length is <= barcode length.

Parameters

seqs	StringSet of forward reads.
seqsRev	StringSet of backward reads.
ids	StringSet of IDs.
barcodes	StringSet of barcodes

Returns

A bool: **false** on different lengths of the barcodes, **true** otherwise.

Remarks

The n-th entries of all given StringSets must be associated.

Warning

If the programm is not terminated uppon return of **false**, crashes or wrong results are to be exspected.

5.2.2.4 template<typename TSeqs , typename TIds , typename TBarcodes > bool check (TSeqs & *seqs,* TIds & *ids,* TBarcodes & *barcodes*)

Overload of check(TSeqs& seqs, TSeqs& seqsRev, TIds& ids, TBarcodes& barcodes) for single-end data.

Parameters

seqs	StringSet of reads.
ids	StringSet of IDs.
barcodes	StringSet of barcodes

5.2.2.5 template<typename TSeqs > void clipBarcodes (TSeqs & *seqs,* const std::vector< int > & *matches,* unsigned *len*)

Function for clipping the barcodes from the sequences if they could be matched beforehand.

Parameters

seqs	StringSet of sequences.
matches	Vector produced by findAllExactIndex or findAllApprox, holding information about the matched barcodes.
len	Unsigned integer representing the length of the barcodes.

Precondition

The n-th element of **seq** must be associated with the n-th element of **matches**.

Warning

If the length of the sequences in seq has not been controlled beforehand by check the program might crash.

5.2.2.6 template<typename TSeqs > void clipBarcodes (TSeqs & *seqs,* int *len*)

Overload of clipBarcodes<seqs, matches, len> if the first len bases of every sequence shall be clipped regardless of the matching(or not matching) barcodes.

Parameters

seqs	StringSet of sequences.
len	unsigned integer representing the length of the barcodes.

Precondition

The n-th element of **seq** must be associated with the n-th element of **matches**.

Warning

Only to be used if the first len bases of every sequence are sure to hold a barcode.
If the length of the sequences in seq has not been controlled beforehand by check the program might crash.

5.2.2.7 template<typename TBarcodes , typename TMultiplex , typename TFinder > std::vector<std::vector<int> > DoAll (TMultiplex & *multiplex,* TBarcodes & *barcodes,* TFinder & *esaFinder*)

Function for performing all demultiplexing operations. Using exact search and multiplex barcodes.

Parameters

multiplex	StringSet of multiplex barcodes.
barcodes	StringSet of barcodes.
esaFinder	esaFinder-object of the barcode-index.

Returns

A two-dimensional vector of integers representing the indices of the sequences.

Remarks

The 0-th group (i.e. 0-th collum of the vector) contains the unidentified sequences.
All following groups hold the sequences associated with the i-1-th barcode.

5.2.2.8 template<typename TBarcodes , typename TMultiplex > std::vector<std::vector<int> > DoAll (TMultiplex & *multiplex,* TBarcodes & *barcodes*)

Overload of ::DoAll(TBarcodes& multiplex, TBarcodes& barcodes, TFinder& esaFinder). Using approximate search and multiplex barcodes.

Parameters

multiplex	StringSet of multiplex barcodes.
barcodes	StringSet of barcodes.

Returns

A two-dimensional vector of integers representing the indices of the sequences.

Remarks

The 0-th group (i.e. 0-th collum of the vector) contains the unidentified sequences.
All following groups hold the sequences associated with the i-1-th barcode.

5.2.2.9 template<typename TSeqs , typename TBarcodes , typename TFinder > std::vector<std::vector<int> > DoAll (TSeqs & *seqs,* TBarcodes & *barcodes,* TFinder & *esaFinder,* bool *hardClip*)

Overload of ::DoAll(TBarcodes& multiplex, TBarcodes& barcodes, TFinder& esaFinder). Using exact search and inline barcodes.

Parameters

seqs	StringSet of sequences.
barcodes	StringSet of barcodes.
esaFinder	esaFinder-object of the barcode-index.
hardClip	Boolean indicating wheter the first Bases of each sequence shall be clipped without considering the barcodes (true).

Returns

A two-dimensional vector of integers representing the indices of the sequences.

Remarks

The 0-th group (i.e. 0-th collum of the vector) contains the unidentified sequences.
All following groups hold the sequences associated with the i-1-th barcode.

5.2.2.10 template<typename TSeqs , typename TBarcodes > std::vector<std::vector<int> > DoAll (TSeqs & *seqs,* TBarcodes & *barcodes,* bool *hardClip*)

Overload of ::DoAll(TBarcodes& multiplex, TBarcodes& barcodes, TFinder& esaFinder). Using approximate search and inline barcodes.

Parameters

seqs	StringSet of sequences.
barcodes	StringSet of barcodes.
hardClip	Boolean indicating wheter the first Bases of each sequence shall be clipped without considering the barcodes (true).

Returns

A two-dimensional vector of integers representing the indices of the sequences.

Remarks

The 0-th group (i.e. 0-th collum of the vector) contains the unidentified sequences.
All following groups hold the sequences associated with the i-1-th barcode.

5.2.2.11 template<typename TPrefices , typename TPatterns > std::vector<int> findAllApprox (const TPrefices & *prefices*, TPatterns & *patterns*)

Function for performing findApprox on all given prefices and barcodes.

Parameters

prefices	StringSet of prefices.
patterns	Vector of pattern-objects generated by makePatterns.

Returns

A vector of integers representing the positions of the matched barcodes.

Remarks

The n-th element of the vector is associated with the n-th sequence.

Warning

Only the first hit of each is reported.

5.2.2.12 template<typename TPrefices , typename TFinder > std::vector<int> findAllExactIndex (const TPrefices & *prefices*, TFinder & *finder*)

Function for performing findAllExactIndex on all given prefices and barcodes.

Parameters

prefices	StringSet of prefices.
finder	Finder-object holding the preprocessed barcodes.

Returns

A Vector of integers representing the positions of the matched barcodes. If a sequence could not be matched, the position is replaced by **-1**.

Remarks

The n-th element of the resulting vector is associated with the n-th prefix/sequence.

Warning

Only the first hit of each prefix is repoted.

5.2.2.13 template<typename TPrefix , typename TPatterns > int findApprox (TPrefix & *prefix,* TPatterns & *patterns*)

Function for approximate search for one prefix in all barcodes.

Parameters

prefix	Prefix of a sequence.
patterns	Vector of pattern-objects generated by makePatterns.

Returns

An integer representing the position of the matched barcode. If no hit occured **-1** is returned.

Warning

Only the first hit is reported.

5.2.2.14 template<typename TPrefix , typename TFinder > int findExactIndex (const TPrefix & *prefix,* TFinder & *finder*)

Function for exact search for one prefix in the barcode indices.

Parameters

prefix	Prefix of a sequence.
finder	Finder-object holding the preprocessed barcodes.

Returns

An integer representing the position of the matching barcode. If no hit occured **-1** is returned.

Warning

Only the first hit is reported.

5.2.2.15 template<typename TSeqs > TSeqs getPrefix (TSeqs & *seqs,* unsigned *len*)

Function for extracting the prefices of all given sequences.

Parameters

seqs	StringSet of sequences.
len	Unsigned integer representing the desired prefix length.

Returns

A StringSet of prefices of length **len**

5.2.2.16 template<typename TMatches , typename TBarcodes > std::vector<std::vector<int> > group (const TMatches & *matches,* const TBarcodes & *barcodes*)

Function for sorting the matched and clipped sequences into one vector.

The function takes the results of findAllExactIndex or findAllApprox and the sequnce indices into a vector

Parameters

matches	Vector holding information about the matched barcodes. Produced by findAllExactIndex or findAllApprox functions.
barcodes	StringSet of barcodes.

Returns

A two-dimensional vector of integers representing the indices of the sequences.

Remarks

The 0-th group (i.e. 0-th collum of the vector) contains the unidentified sequences.
All following groups hold the sequences associated with the i-1-th barcode.

5.2.2.17 template<typename TBarcodes > std::vector<Pattern<String<Dna5Q>, DPSearch<SimpleScore> > > makePatterns (TBarcodes & *barcodes*)

Function for creating a vector of patterns used by findApprox and findAllApprox.

The pattern-objects generated use DPSearch and a simple score of 0,-1,-2 (match, mismatch, gap)

Parameters

barcodes	StringSet of barcodes.

Returns

A vector of pattern-Objects.

5.2.2.18 void resizeGroups (std::vector< std::vector< int > > & *groups*)

Function for resizing the groups of the vector produced by group and used by it.

Parameters

groups	Two-dimensional vector containing integers representing the indices of the sequences. Produced by group.

5.2.2.19 template<typename TgSeqs , typename TgIds , typename TBarcodeIds > int writeGroups (TgSeqs & *gSeqs,* TgSeqs & *gSeqsRev,* TgIds & *gIds,* TBarcodeIds & *barcodeIds,* std::vector< std::vector< int > > & *groups,* String< char > & *path*)

Function for Writing out the demultiplexed sequences.

The output consists of one or more FastQ files, one for each used barcode. The filenames are derived from the barcode IDs.

Parameters

gSeqs	Vector of StringSets of sequences, generated by buildSets.
gIds	Vector of SringSets of IDs, generated by buildSets.
barcodeIds	StringSet of barcode IDs.
groups	Two-dimensional vector of integers representing the indices of the sequences. Produced by DoAll, or group.
path	String<char> indicating the output folder.

Returns

An integer. 1 on errors, 0 otherwise.

Remarks

When giving the path to the output-folder, the last slash must be written. Otherwise everything after the last slash will be added to the filenames.
Not used in final code, but used in a test of the demultiplexing functions.

5.2.2.20 template<typename TgSeqs , typename TgIds , typename TBarcodeIds , typename Tgroups > int writeGroups (TgSeqs & *gSeqs,* TgIds & *gIds,* TBarcodeIds & *barcodeIds,* Tgroups & *groups,* String< char > & *path*)

Function for Writing out the demultiplexed sequences.

The output consists of one or more FastQ files, two for each used barcode. The filenames are derived from the barcode IDs. The files containing the backward-reads end with "_REV.fq".

Parameters

gSeqs	Vector of StringSets of forward-reads, generated by buildSets.
gSeqsRev	Vector of StringSets of backward-reads, generated by buildSets.
gIds	Vector of SringSets of IDs, generated by buildSets.
barcodeIds	StringSet of barcode IDs.
groups	Two-dimensional vector of integers representing the indices of the sequences. Produced by DoAll, odr group.
path	String<char> indicating the output folder.

Returns

An integer. 1 on errors, 0 otherwise.

Remarks

When giving the path to the output-folder, the last slash must be written. Otherwise everything after the last slash will be added to the filenames.
Not used in final code, but used in a test of the demultiplexing functions.

5.3 D:/SeqAn/Development/seqan-trunk/sandbox/my_sandbox/apps/SeqDPT/readTrimming.h File Reference

Contains the functions for read trimming.

Classes

- struct Tail

  The tagging structure for the Tail trimming algorithm.
- struct BWA

  The tagging structure for the BWA trimming algorithm.
- struct Mean

  The tagging structure for the window trimming algorithm.
- struct Dna5QAdapter

  This structure wraps a sequence with dedicated Dna5 and quality strings so we can use it with our trimming function.
- struct QualityTrimmingStats

Functions

- unsigned getQuality (const seqan::String< seqan::Dna5Q > &seq, unsigned i)

  Determines the quality at position i of the Dna5QString.
- unsigned getQuality (const Dna5QAdapter &seq, unsigned i)

  Determines the quality at position i in the quality string of the Dna5QAdapter.
- unsigned length (const Dna5QAdapter &seq)

  A specialization of the length function for use with the wrapper Dna5QAdapter. Returns the length of the sequence contained in the wrapper.
- void erase (const Dna5QAdapter &seq, unsigned begin, unsigned end)

  A specialization of the erase function for use with the wrapper Dna5QAdapter. Simply trimmes both sequence and quality strings separately.
- template<typename TSeq >
  unsigned _trimRead (const TSeq &seq, unsigned const cutoff, Tail const &)

  A very simple trimming mechanism that removes low quality bases from the end.
- template<typename TSeq >
  unsigned _trimRead (const TSeq &seq, unsigned const cutoff, BWA const &)

  A trimming algorithm using the method implemented in BWA.
- template<typename TSeq >
  unsigned _trimRead (const TSeq &seq, unsigned const _cutoff, Mean const &spec)

  A trimming algorithm using a sliding window method to find the trimming position.
- template<typename TSeq , typename TQual , typename TSpec >
  unsigned trimRead (TSeq &seq, TQual &qual, unsigned const cutoff, TSpec const &spec)

  Interface that trims a sequence.
- template<typename TSeq , typename TSpec >
  unsigned trimRead (TSeq &seq, unsigned const cutoff, TSpec const &spec)

  Interface that trims a sequence.
- template<typename TSet , typename TSpec >
  unsigned _trimReads (TSet &seqSet, unsigned const cutoff, TSpec const &spec)

  Trims a set of reads and marks reads that are too short for removal.
- template<typename TSet1 , typename TSet2 , typename TId >
  void removeValues (TSet1 &set1, TSet2 &set2, TId id)

  *Removes two sequences at the position **id** from two StringSets.*
- template<typename TId , typename TSeq >
  unsigned dropReads (seqan::StringSet< TId > &idSet, seqan::StringSet< TSeq > &seqSet, unsigned const min_length, QualityTrimmingStats &stats)

  Drop reads which are too short. This is done in a way such that the pair structure is conserved.
- template<typename TId , typename TSeq >
  unsigned dropReads (seqan::StringSet< TId > &idSet1, seqan::StringSet< TSeq > &seqSet1, seqan::StringSet< TId > &idSet2, seqan::StringSet< TSeq > &seqSet2, unsigned const min_length, QualityTrimmingStats &stats)

  Drop reads which are too short. This is done in a way such that the pair structure is conserved.
- template<typename TSeq , typename TSpec >
  unsigned trimBatch (seqan::StringSet< TSeq > &seqSet, unsigned const cutoff, TSpec const &spec)

  Trims bad quality bases from a set of sequences.
- template<typename TSeq , typename TSpec >
  seqan::Pair< unsigned, unsigned > trimPairBatch (seqan::StringSet< TSeq > &seqSet1, seqan::StringSet< TSeq > &seqSet2, unsigned const cutoff, TSpec const &spec)

  Trims bad quality bases from two sets of sequences.

5.3.1 Detailed Description

Contains the functions for read trimming.

5.3.2 Function Documentation

5.3.2.1 template<typename TSeq > unsigned _trimRead (const TSeq & *seq,* unsigned const *cutoff,* Tail const &)

A very simple trimming mechanism that removes low quality bases from the end.

Parameters

seq	The sequence that shall be trimmed.
cutoff	The minimum quality required.

Returns

The trimming position, i.e. the first base to be removed.

Remarks

Simply cut off as many low quality bases from the end as possible before finding a good one.

5.3.2.2 template<typename TSeq > unsigned _trimRead (const TSeq & *seq,* unsigned const *cutoff,* BWA const &)

A trimming algorithm using the method implemented in BWA.

Parameters

seq	The sequence that shall be trimmed.
cutoff	The minimum quality required.

Returns

The trimming position, i.e. the first base to be removed.

Remarks

Trimming mechanism used in BWA. Trim to argmax_x sum_{i=x+1}^l {cutoff - q_i}

5.3.2.3 template<typename TSeq > unsigned _trimRead (const TSeq & *seq,* unsigned const *_cutoff,* Mean const & *spec*)

A trimming algorithm using a sliding window method to find the trimming position.

Parameters

seq	The sequence that shall be trimmed.
cutQual	The minimum quality required.
spec	The algorithm tag containing the window size used for trimming.

Returns

An unsigned int representing the trimming position, i.e. the position of the first base to be removed.

5.3.2.4 template<typename TSet , typename TSpec > unsigned _trimReads (TSet & *seqSet,* unsigned const *cutoff,* TSpec const & *spec*)

Trims a set of reads and marks reads that are too short for removal.

Parameters

removedID	A collection that stores the IDs of sequences marked for removal.
seqSet	A collection of sequences that shall be trimmed.
cutoff	The minimum quality required from a base.
min_length	The minimum length required after trimming. Shorter sequences are marked for removal in **removedID**.
spec	The trimming algorithm used for trimming.

Returns

The number of reads where bases were removed.

5.3.2.5 template<typename TId , typename TSeq > unsigned dropReads (seqan::StringSet< TId > & *idSet,* seqan::StringSet< TSeq > & *seqSet,* unsigned const *min_length,* QualityTrimmingStats & *stats*)

Drop reads which are too short. This is done in a way such that the pair structure is conserved.

Parameters

idSet	StringSet of FastA-IDs for the set of sequences.
seqSet	StringSet containing the reads.
min_length	The minimum length required after trimming. Shorter sequences will be deleted.
stats	Quality statistic struct to store how many reads were dropped.

5.3.2.6 template<typename TId , typename TSeq > unsigned dropReads (seqan::StringSet< TId > & *idSet1,* seqan::StringSet< TSeq > & *seqSet1,* seqan::StringSet< TId > & *idSet2,* seqan::StringSet< TSeq > & *seqSet2,* unsigned const *min_length,* QualityTrimmingStats & *stats*)

Drop reads which are too short. This is done in a way such that the pair structure is conserved.

Parameters

idSet1	StringSet of FastA-IDs for the first set of sequences.
seqSet1	StringSet containing the forward reads of the paired sequences.
idSet2	StringSet of FastA-IDs for the second set of sequences.
seqSet2	StringSet containing the backward reads of the paired sequences.
min_length	The minimum length required after trimming. Shorter sequences will be deleted. (See remark).
stats	Quality statistic struct to store how many reads were dropped.

Remarks

If one read is marked for removal, while its sibling is not, the read will be replaced by a single "N". If both reads are marked for removal, they are removed. This way the relation between paired reads stays intact.

5.3.2.7 unsigned getQuality (const seqan::String< seqan::Dna5Q > & *seq,* unsigned *i*) [inline]

Determines the quality at position i of the Dna5QString.

Parameters

seq	The Dna5QString containing bases and qualities.
i	The index of the base whose quality shall be returned.

Returns

Phred quality of the base at position **i**.

5.3.2.8 unsigned getQuality (const Dna5QAdapter & *seq*, unsigned *i*) [inline]

Determines the quality at position i in the quality string of the Dna5QAdapter.

Parameters

seq	The Dna5QAdapter structure containing the quality string.
i	The index of the base whose quality shall be returned.

Returns

Phred quality of the base at position **i**.

5.3.2.9 template<typename TSet1 , typename TSet2 , typename TId > void removeValues (TSet1 & *set1*, TSet2 & *set2*, TId *id*)

Removes two sequences at the position **id** from two StringSets.

Parameters

set1	The first set where position **id** is removed.
set2	The second set where position **id** is removed.
id	The id that identifies the read to be removed.

5.3.2.10 template<typename TSeq , typename TSpec > unsigned trimBatch (seqan::StringSet< TSeq > & *seqSet*, unsigned const *cutoff*, TSpec const & *spec*)

Trims bad quality bases from a set of sequences.

Parameters

idSet	StringSet of FastA-IDs for the first set of sequences.
seqSet	StringSet containing the forward reads of the paired sequences.
cutoff	The minimum quality required of a base.
spec	The trimming algorithm used to trim the sequences.

Returns

The number of sequences which had bases removed from.

5.3.2.11 template<typename TSeq , typename TSpec > seqan::Pair<unsigned, unsigned> trimPairBatch (seqan::StringSet< TSeq > & *seqSet1*, seqan::StringSet< TSeq > & *seqSet2*, unsigned const *cutoff*, TSpec const & *spec*)

Trims bad quality bases from two sets of sequences.

Parameters

seqSet1	StringSet containing the forward reads of the paired sequences.
seqSet2	StringSet containing the backward reads of the paired sequences.
cutoff	The minimum quality required from a base.
spec	The trimming algorithm used to trim the sequences.

Returns

A pair of unsigned ints containing the number of reads which had bases removed from.

5.3.2.12 template<typename TSeq , typename TQual , typename TSpec > unsigned trimRead (TSeq & *seq*, TQual & *qual*, unsigned const *cutoff*, TSpec const & *spec*)

Interface that trims a sequence.

Parameters

seq	The sequence that shall be trimmed.
cutoff	The minimum quality required from a base.
spec	The trimming algorithm used for trimming.

Returns

The number of bases trimmed from the sequence.

Remarks

This function takes dedicated sequence and quality strings, puts them into a wrapper structure and calls trimRead(TSeq& seq, unsigned const cutoff, TSpec const & spec).

5.3.2.13 template<typename TSeq , typename TSpec > unsigned trimRead (TSeq & *seq*, unsigned const *cutoff*, TSpec const & *spec*)

Interface that trims a sequence.

Parameters

seq	The sequence that shall be trimmed.
cutoff	The minimum quality required from a base.
spec	The trimming algorithm used for trimming.

Returns

The number of bases trimmed from the sequence.

Index