Annotation comparison Vetea Jacot

Generated by Doxygen 1.9.1

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Namespace Documentation

3.1 main Namespace Reference

This script is used to compute the distance between two structural annotations of a same genome, one reference and one alternative annotation.

Classes

- class Clusters

  This class represents clusters of overlapping loci computed by the function construct_clusters.
- class Locus

  This class represents an annotation's locus identified from reading the associated GFF file with the get_gff_borders function.

Functions

- def get_structure_id (line, debug=False, verbose=False)

  This function retrieves and returns the id of the structure described from a line read from a GFF file.
- def get_parent_id (line, debug=False, verbose=False)

  This function retrieves and returns the parent id of the structure described from a line read from a GFF file.
- def get_gff_borders (path, debug=False, verbose=False)

  This function expects a string corresponding to the file path of the GFF file to read, and returns a dictionary of instances of the class 'Locus', detailing all the relevant information for the gene and its mRNAs.
- def annotation_sort (dict_ref, dict_alt, debug=False, verbose=False)

  This function creates a list of all the locus coordinates for both annotations and sorts it in ascending order by their lower bound position.
- def construct_clusters (dict_ref, dict_alt, locus_order, debug=False, verbose=False)

  This function groups every locus in the tuple list 'locus_order' in instances of the 'Clusters' class depending on the overlap of the reference and alternative loci.
- def get_area_bounds (ref, alt, debug=False, verbose=False)

  This function retrieves all the CDS coordinates from the given lists of coordinates of the reference and alternative annotations and includes them in a unique list of coordinates.
- def is_in_cds (cds_bounds, area_bounds, debug=False, verbose=False)

  This function indicates for each couple of bounds in the given list of area bounds (.
- def compare_loci (ref_locus, alt_locus, debug=False, verbose=False)

This function compares two annotations' loci returned by the function get_gff_borders and creates for each pair of reference-alternative mRNAs a comparison list detailing the identities and differences between the two annotations's codon position structure.

- def create_vectors (borders, debug=False, verbose=False)

  This function expects a list of all CDS coordinates (start and end) of a locus.
- def old_compare_loci (ref_locus, alt_locus, debug=False, verbose=False)

  This function expects two loci corresponding to two annotations of the same genome, creates and compares their structure strings (create_vectors function)
- def annotation_match (cluster_ref, cluster_alt, create_strings, debug=False, verbose=False)

  This function compares the loci of the reference and alternative clusters returned by the construct_clusters function by assigning each locus to the overlapping locus of the other annotation cluster which gives the highest computed identity.
- def write_results (results, debug=False, verbose=False)

  This function writes to a new 'results.csv' file the results of the annotation comparison retrieved from the identities dictionary returned by the annotation_match function.
- def annotation_comparison (ref_path, alt_path, debug=False, verbose=False, create_strings=False)

  Main function of this program.
- def **usage** ()
- def **main** ()

3.1.1 Detailed Description

This script is used to compute the distance between two structural annotations of a same genome, one reference and one alternative annotation.

It expects as input the paths to the annotation files (in GFF format), displays the computed distances between all annotation pairs, and returns a dictionary of lists of lists detailing the matchs/mismatchs between the two annotations' structure string

3.1.2 Function Documentation

3.1.2.1 annotation_comparison()

```
def main.annotation_comparison (
              ref_path,
              alt_path,
              debug = False,
              verbose = False,
              create_strings = False )
```

Main function of this program.

Given a reference and alternative path, gets the corresponding GFF files and compares the two annotations to return their structure's identity level

Parameters

ref_path	Path of the GFF file describing the reference annotation
alt_path	Path of the GFF file describing the aternative annotation
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Returns

Returns a dictionary of dictionaries of floats corresponding to the structure string identity between each locus of each annotation compared to those of the reference

Remarks

Loci found in one annotation but not the other are ignored

3.1.2.2 annotation_match()

```
def main.annotation_match (
              cluster_ref,
              cluster_alt,
              create_strings,
              debug = False,
              verbose = False )
```

This function compares the loci of the reference and alternative clusters returned by the construct_clusters function by assigning each locus to the overlapping locus of the other annotation cluster which gives the highest computed identity.

Each comparison results are displayed in the terminal and written to a 'results' dictionary detailing locus information.

See also

construct_clusters()

Parameters

cluster_ref	Dictionary of Locus class instances describing the loci of the reference annotation cluster
cluster_alt	Dictionary of Locus class instances describing the loci of the alternative annotation cluster
create_strings	Boolean indicating wether to use the new compare_loci fucntion ('False') or the old_compare_loci function ('True')
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

See also

compare_loci()

old_compare_loci()

Returns

Returns a dictionary detailing the locus information for each locus comparison

3.1.2.3 annotation_sort()

```
def main.annotation_sort (
              dict_ref,
              dict_alt,
              debug = False,
              verbose = False )
```

This function creates a list of all the locus coordinates for both annotations and sorts it in ascending order by their lower bound position.

Parameters

dict_ref	Dictionary containing all loci of the reference annotation, as returned by the 'get_gff_borders' function
dict_alt	Dictionary containing all loci of the alternative annotation, as returned by the 'get_gff_borders' function
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

See also

get_gff_borders()

Returns

Returns a list of tuples containing the lower and upper bounds of each locus, its locus ID, and a boolean indicating if the locus was retrieved from the reference (True) or the alternative (False)

3.1.2.4 compare_loci()

```
def main.compare_loci (
              ref_locus,
              alt_locus,
              debug = False,
              verbose = False )
```

This function compares two annotations' loci returned by the function get_gff_borders and creates for each pair of reference-alternative mRNAs a comparison list detailing the identities and differences between the two annotations's codon position structure.

It returns a tuple containing the comparison list giving the highest identity, the computed identity level, and the list of mismatch areas indentified by the comparison.

See also

get_gff_borders()

Parameters

ref_locus	The reference annotation's locus (Locus class instance)
alt_locus	The alternative annotation's locus (Locus class instance)
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Returns

Returns a tuple containing a list indicating the number of codon position mismatches (first position) and matches (second position) between the two border lists giving the maximum indentity between all mRNAs, the identity level computed from this list, and the comaprison areas producing mismatches

Remarks

This function doesn't expect any annotation to be a 'reference'

3.1.2.5 construct_clusters()

```
def main.construct_clusters (
              dict_ref,
              dict_alt,
              locus_order,
              debug = False,
              verbose = False )
```

This function groups every locus in the tuple list 'locus_order' in instances of the 'Clusters' class depending on the overlap of the reference and alternative loci.

Each cluster contains a group of mutually-overlapping loci which don't overlap with others.

See also

Clusters

Parameters

dict_ref	Dictionary of reference loci, as returned by get_gff_borders
dict_alt	Dictionary of alternative loci, as returned by get_gff_borders
locus_order	list of tuples containing the information for each locus of each annotation in ascending order of their lower border, as returned by annotation_sort
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

See also

get_gff_borders()
annotation_sort()

Remarks

The clusters names will not necessarily follow each other. 'Cluster6' might follow 'Cluster2'

Returns

an instance of the Clusters class describing the cluster structure of the two annotations loci

3.1.2.6 create_vectors()

```
def main.create_vectors (
              borders,
              debug = False,
              verbose = False )
```

This function expects a list of all CDS coordinates (start and end) of a locus.

It returns a list indicating the start of the locus as first value and a string describing the codon position of each nucleotide (1,2,3, or 0 in the case of a non-CDS nucleotide) of the locus/gene as second value

Parameters

borders	The list containing all start-end coordinates of the annotation's locus' CDS
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

See also

get_gff_borders()

Returns

Returns a list describing the start of the locus and the annotation structure of the locus

3.1.2.7 get_area_bounds()

```
def main.get_area_bounds (
              ref,
              alt,
              debug = False,
              verbose = False )
```

This function retrieves all the CDS coordinates from the given lists of coordinates of the reference and alternative annotations and includes them in a unique list of coordinates.

The coordinates are sorted in ascending order.

Parameters

ref	List of CDS coordinates of the reference annotation
alt	List of CDS coordinates of the alternative annotation
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Returns

Returns a list of coordinates compiling all coordinates from both initial lists in ascending order

3.1.2.8 get_gff_borders()

```
def main.get_gff_borders (
              path,
              debug = False,
              verbose = False )
```

This function expects a string corresponding to the file path of the GFF file to read, and returns a dictionary of instances of the class 'Locus', detailing all the relevant information for the gene and its mRNAs.

Parameters

path	Path of the file to read
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Remarks

If the parent ID of a CDS does not match the ID of the previous mRNA (indicating an incorrect file structure), an entry is added to a 'log' file but the function is not interrupted

Returns

Returns a dictionary of instances of the class 'Locus', containing the information of the CDS borders of each mRNA of the gene, the start and end coordinates, the DNA strand on which the gene is predicted, and the locus ID

See also

Locus

3.1.2.9 get_parent_id()

```
def main.get_parent_id (
              line,
              debug = False,
              verbose = False )
```

This function retrieves and returns the parent id of the structure described from a line read from a GFF file.

Parameters

line	The line read from the file (string)
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Remarks

This function expects the file to be in GFF format, and the parent id field to come after the id field

Returns

Returns the id of the parent of the structure described by the line

3.1.2.10 get_structure_id()

```
def main.get_structure_id (
              line,
              debug = False,
              verbose = False )
```

This function retrieves and returns the id of the structure described from a line read from a GFF file.

Parameters

line	The line read from the file (string)
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Remarks

This function expects the file to be in GFF format

Returns

Returns the id of the structure described by the line

3.1.2.11 is_in_cds()

```
def main.is_in_cds (
              cds_bounds,
              area_bounds,
              debug = False,
              verbose = False )
```

This function indicates for each couple of bounds in the given list of area bounds (.

Parameters

area_bounds),if	they delimit an area which includes a CDS from the given CDS coordinates list (
cds_bounds).	It is used during the comparison of areas in compare_loci() to know if the reference or alternative have a CDS in the area

See also

compare_loci()

Parameters

cds_bounds	List of CDS coordinates for an annotation
area_bounds	List of area bounds
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Returns

Returns a list indicating for each couple of bounds if they include a CDS ('True') or not ('False')

3.1.2.12 old_compare_loci()

```
def main.old_compare_loci (
              ref_locus,
              alt_locus,
              debug = False,
              verbose = False )
```

This function expects two loci corresponding to two annotations of the same genome, creates and compares their structure strings (create_vectors function)

See also

create_vectors()

Parameters

borders_vector_ref	List of start position and vector of the locus of the reference annotation
borders_vector_alt	List of start position and vector of the locus of the alternative annotation
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Returns

Returns a tuple containing the list of mismatches (first value) and matches (second value) and the computed string identity

Remarks

This function doesn't expect any annotation to be a 'reference'

3.1.2.13 write_results()

```
def main.write_results (
              results,
              debug = False,
              verbose = False )
```

This function writes to a new 'results.csv' file the results of the annotation comparison retrieved from the identities dictionary returned by the annotation_match function.

See also

annotation_match()

Parameters

results	A list of list of dictionaries containing results of the annotation comparison, as returned by annotation_match
debug	If True, triggers display of many messages intended for debugging the program. Default is 'False'
verbose	If True, triggers display of more information messages. Default is 'False'

Remarks

Results are written in CSV ('Comma-Separated Values') format

Chapter 4

Class Documentation

4.1 main.Clusters Class Reference

This class represents clusters of overlapping loci computed by the function construct_clusters.

Public Member Functions

- def __init__ (self)

  This method initialises the class with an empty dictionary.
- def clusters (self)

  This method is used to retrieve the 'clusters' attribute of an instance.
- def get_mRNAs (self, loc_id, ref)

  This method is used to retrieve the complete list of mRNAs of the locus of the cluster specified with its locus ID and a boolean indicating if the locus is to be searched for in the reference or alternative list.

Public Attributes

- **clusters**

4.1.1 Detailed Description

This class represents clusters of overlapping loci computed by the function construct_clusters.

It posesses only one attribute, 'clusters', which is a dictionary of dictionaries of instances of the class 'Locus' indicating the loci attributed to each cluster.

See also

construct_clusters()

Locus

4.1.2 Member Function Documentation

4.1.2.1 clusters()

```
def main.Clusters.clusters (
              self )
```

This method is used to retrieve the 'clusters' attribute of an instance.

Returns

Returns the 'clusters' attribute of the class instance

4.1.2.2 get_mRNAs()

```
def main.Clusters.get_mRNAs (
              self,
              loc_id,
              ref )
```

This method is used to retrieve the complete list of mRNAs of the locus of the cluster specified with its locus ID and a boolean indicating if the locus is to be searched for in the reference or alternative list.

Parameters

loc_id	Identifier of the locus from which to retrieve mRNAs
ref	Boolean indicating from which of the reference (True) or alternative (False) locus to retrieve the mRNAs

Returns

Returns the list of all mRNAs of the specified locus

Remarks

This method is intended to be used in unit tests to verify the clusters contains the intended loci

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

- /home/runner/work/Annotation_Comparison/Annotation_Comparison/script/main.py

4.2 main.Locus Class Reference

This class represents an annotation's locus identified from reading the associated GFF file with the get_gff_borders function.

Public Member Functions

- def __init__ (self, name="", mRNAs=None, start=-1, end=-1, direction="")

  This method initialises the class with default values for each attribute.
- def mRNAs (self)

  This method is used to retrieve the 'mRNAs' attribute of an instance.
- def contain_mrnas (self, ∗∗mrnas)

  This method is used to verify if all mRNAs of a dictionary are present in the class instance's 'mRNAs' attribute.
- def show_init (self)

  This function was added for convenience as a way to easily retrieve the attribute values of the class instance as a formatted string.

Public Attributes

- **name**
- **mRNAs**
- **start**
- **end**
- **direction**

4.2.1 Detailed Description

This class represents an annotation's locus identified from reading the associated GFF file with the get_gff_borders function.

It posesses multiple attributes describing the locus

See also

get_gff_borders()

4.2.2 Constructor & Destructor Documentation

4.2.2.1 __init__()

```
def main.Locus.__init__ (
              self,
              name = "",
              mRNAs = None,
              start = -1,
              end = -1,
              direction = "" )
```

This method initialises the class with default values for each attribute.

Parameters

name	ID of the locus
mRNAs	Dictionary listing all mRNAs of the locus. Each key corresponds to the ID of the mRNA and eahc value to the list of all CDS coordinates as retrieved by the get_gff_borders function
start	Start coordinate of the locus
end	End coordinate of the locus
dicrection	String indicating if the locus is on the 'direct' or 'reverse' strand

See also

get_gff_borders()

4.2.3 Member Function Documentation

4.2.3.1 contain_mrnas()

```
def main.Locus.contain_mrnas (
              self,
             ** mrnas )
```

This method is used to verify if all mRNAs of a dictionary are present in the class instance's 'mRNAs' attribute.

Parameters

***mRNAs*	Dictionary with mRNA names as keys and CDS coordinates list as values

Returns

Returns a boolean indicating if all given mRNAs were found in the class instance's 'mRNAs' attribute

Remarks

This method is intended to be used in unit tests to verify the locus contains the intended mRNAs

4.2.3.2 mRNAs()

```
def main.Locus.mRNAs (
              self )
```

This method is used to retrieve the 'mRNAs' attribute of an instance.

Returns

Returns the 'mRNAs' attribute of the class instance

4.2.3.3 show_init()

```
def main.Locus.show_init (
              self )
```

This function was added for convenience as a way to easily retrieve the attribute values of the class instance as a formatted string.

Returns

Returns a string describing the values of each attribute of the class instance

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

- /home/runner/work/Annotation_Comparison/Annotation_Comparison/script/main.py

Index