Neisseria Ortholog Analysis Tool

Leo Przybylski
przybyls@arizona.edu

February 27, 2010

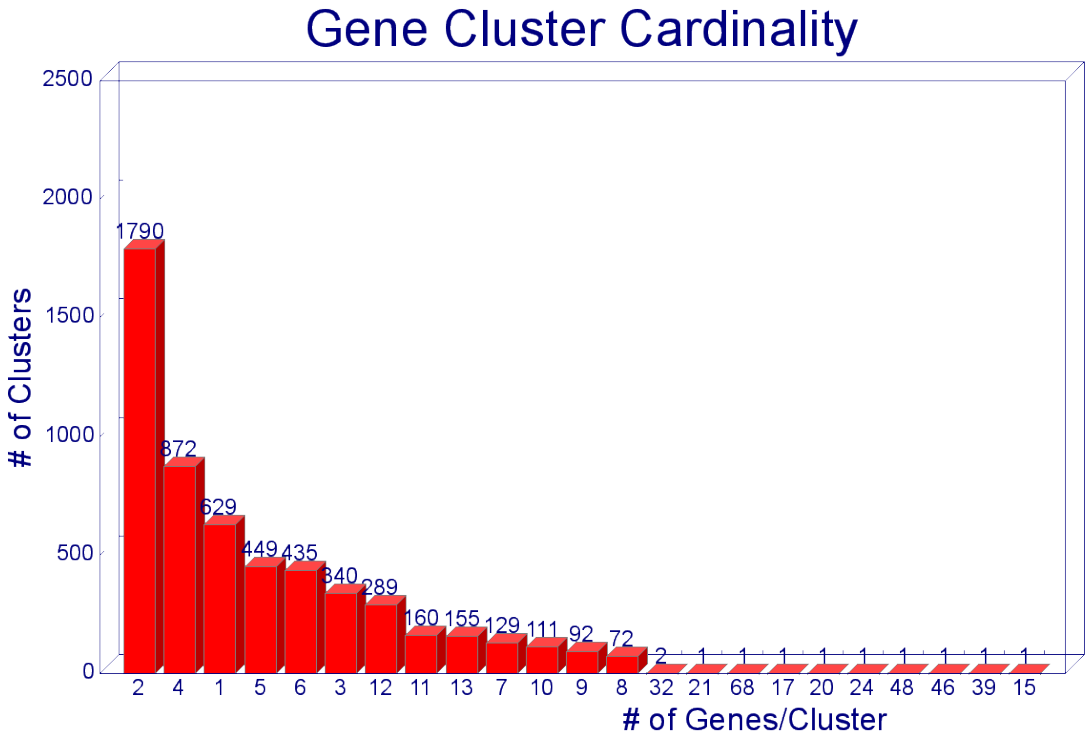
Contents

Abstract

...

Background

When a blast query is made, results are recorded with the identifier of the query as well as the identifier for the result or hit. The convention used to name the each query is a combination of gene and contig number. The records of blast output can be organized into several different types of meaningful information.



One way in particular is to group records related by query and or hit information as clusters. We can then provide statistical information on clusters. By grouping clusters with the same cardinality of edges, we can analyze trends between different families of bacteria. One can analyze that similar families can grow at the same rate or that similar families can recombine with similar genes. This is the benefit of ortholog analysis of gene clusters.

Clusters

A Cluster is a set of edges where the query identifiers match. For example, in the following blast results:

```
AP206_contig00001_4923-1612 cinerea_contig00013_10696-7277 47.65 1129 517 23 29 1103
31 1139 0.0  884
AP206_contig00001_4923-1612 elongata_contig01464_47682-51131 46.11 1156 545 24 7 1103
13 1149 0.0  851
```

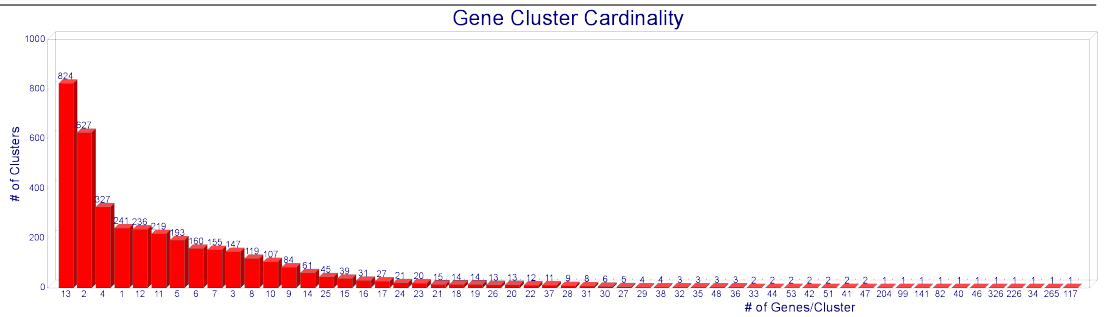
The records belong in the same cluster because they were results of the same query. Not all records that match the query identifier become edges. A gene may relate to another gene cluster by defining a cross-cluster relationship. For example, if we added the following data:

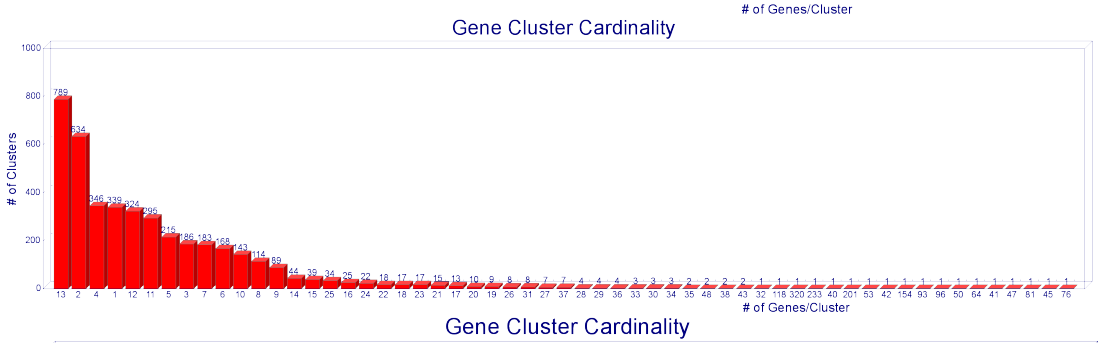
```
sicca_contig00265_4037-6052 AP206_contig00001_4923-1612 24.92 309 170 14 408 671 812
1103 4e-04 42.4
```

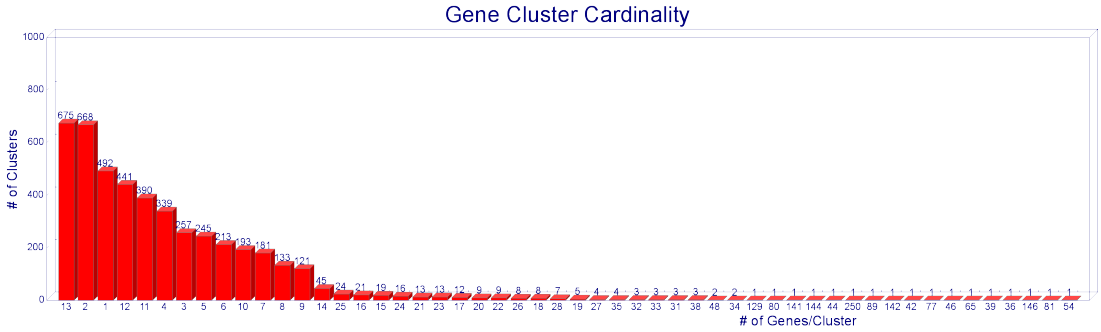
The above will create an edge between *AP206_contig00001_49231612* and *sicca_contig00265_4037-6052* clusters. Then, all of the hits relating to the *sicca_contig00265_40376052* query will suddenly be added to the cluster as well.

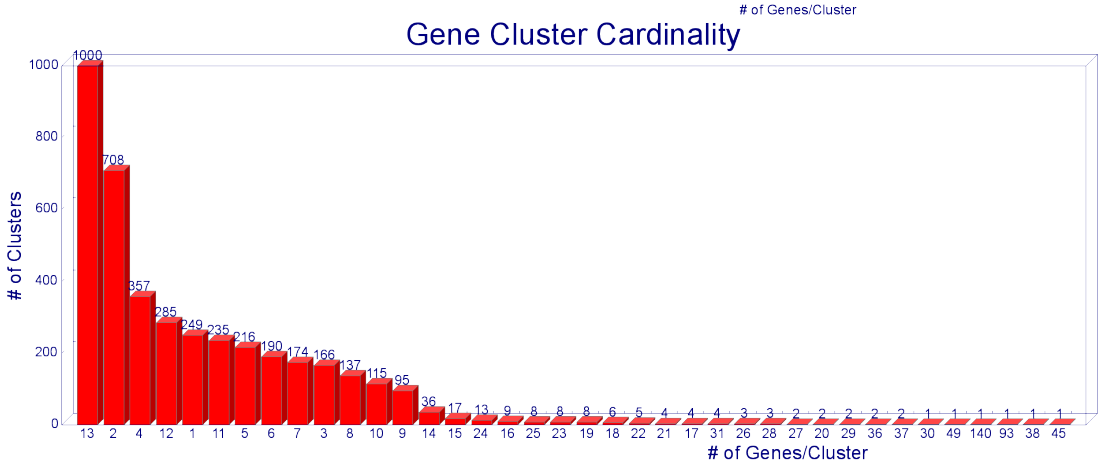
The edge must also fulfill identity percentage and alignment length ratio requirements. The identity percentage must be with in the bounds specified at runtime. The identity percentage requirement is within the range $\in \{30, 45, 60, 75, 90\}$. Likewise, the alignment length ratio must be within the range $\in \{50, 70, 90\}$ which is defined at runtime. Again, the identity percentage and alignment length ratio requirements are variable and determined at runtime. This study provided results on each permutation of identity percentage and alignment length ratio.

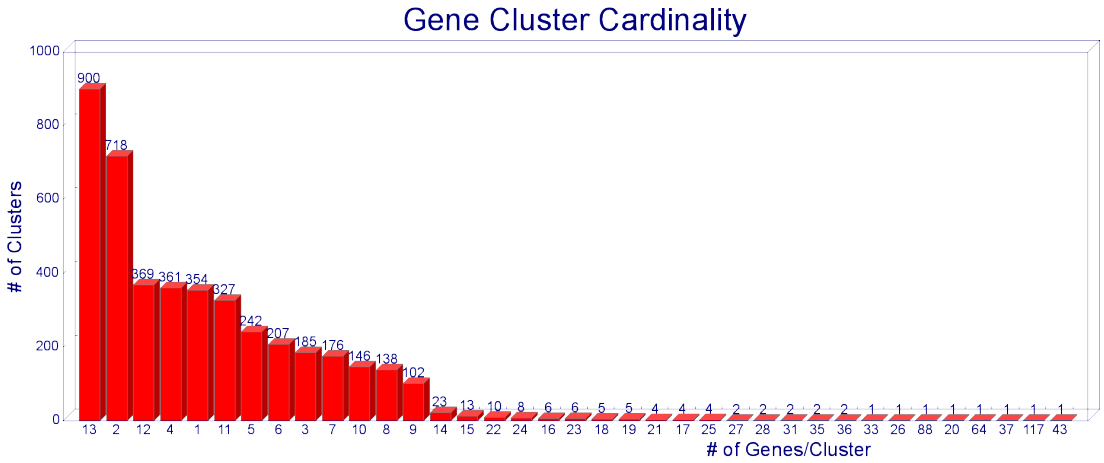
Conclusion

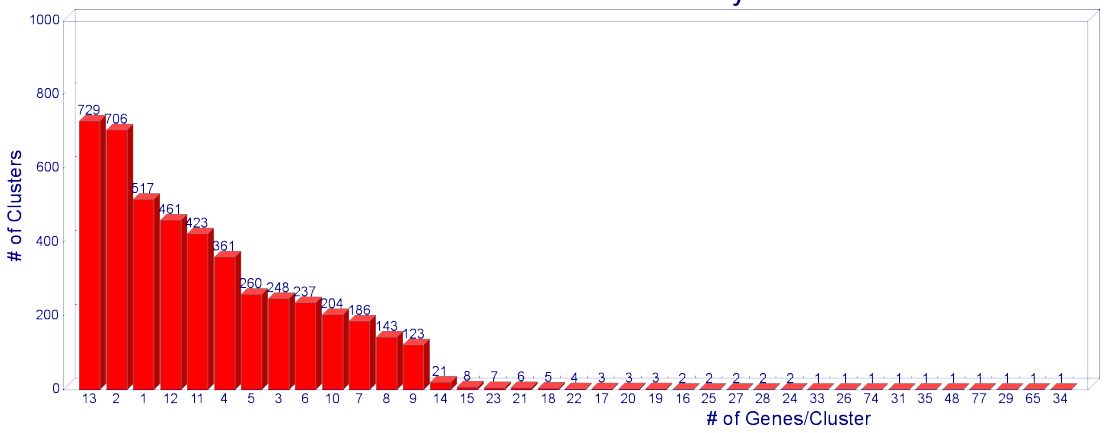

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster

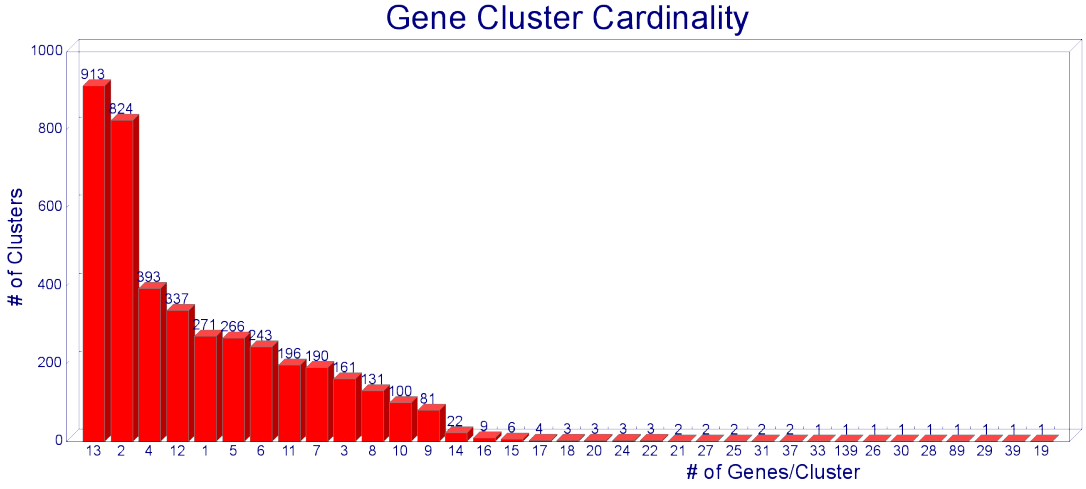

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster

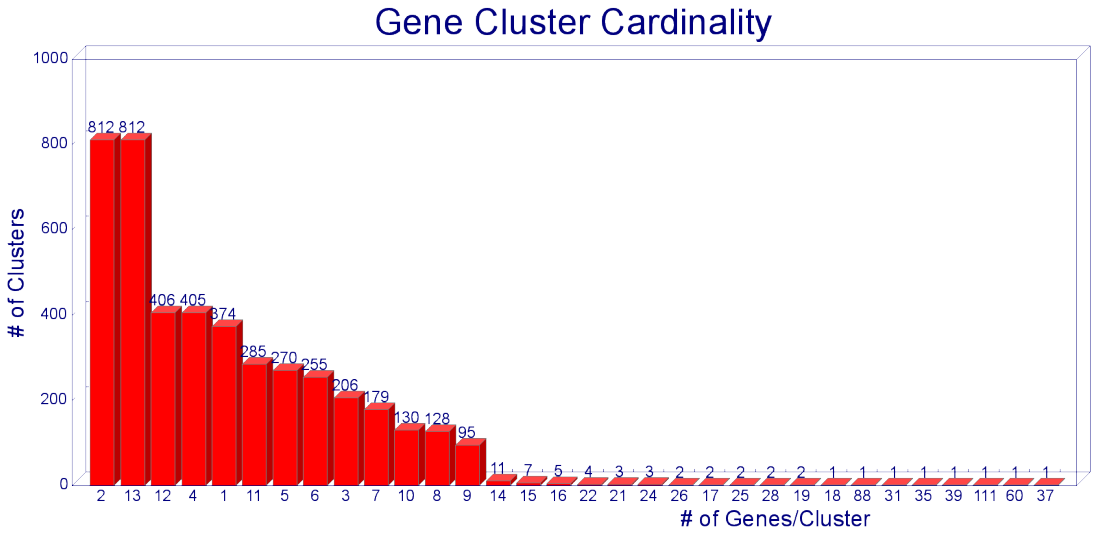

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster

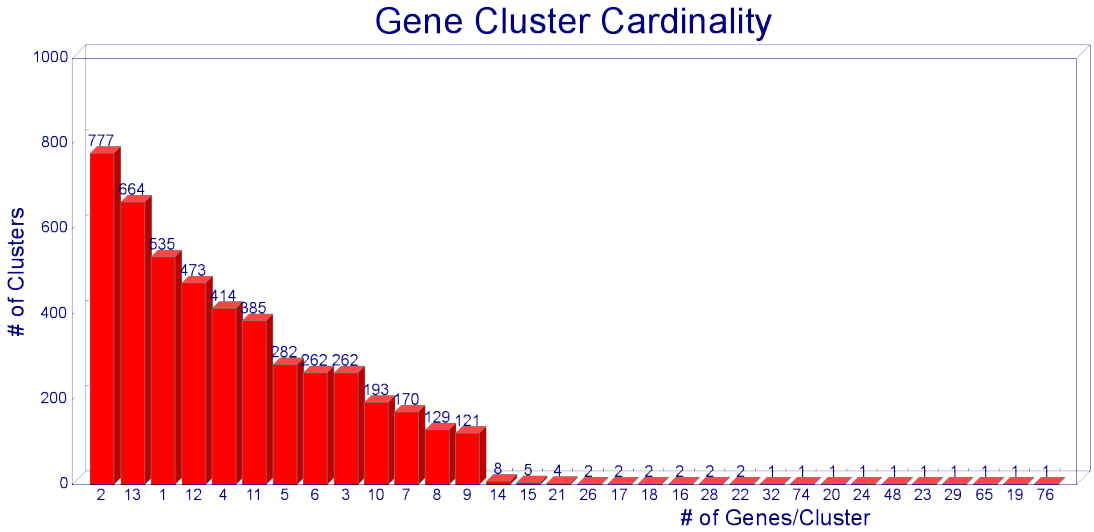

Gene Cluster Cardinality
# of Clusters
1000
800
600
400
200
0
13 2 4 12 1 11 5 6 7 3 8 10 9 14 15 24 16 25 23 19 18 22 21 17 31 26 28 27 20 29 36 37 30 49 140 93 38 45
# of Genes/Cluster

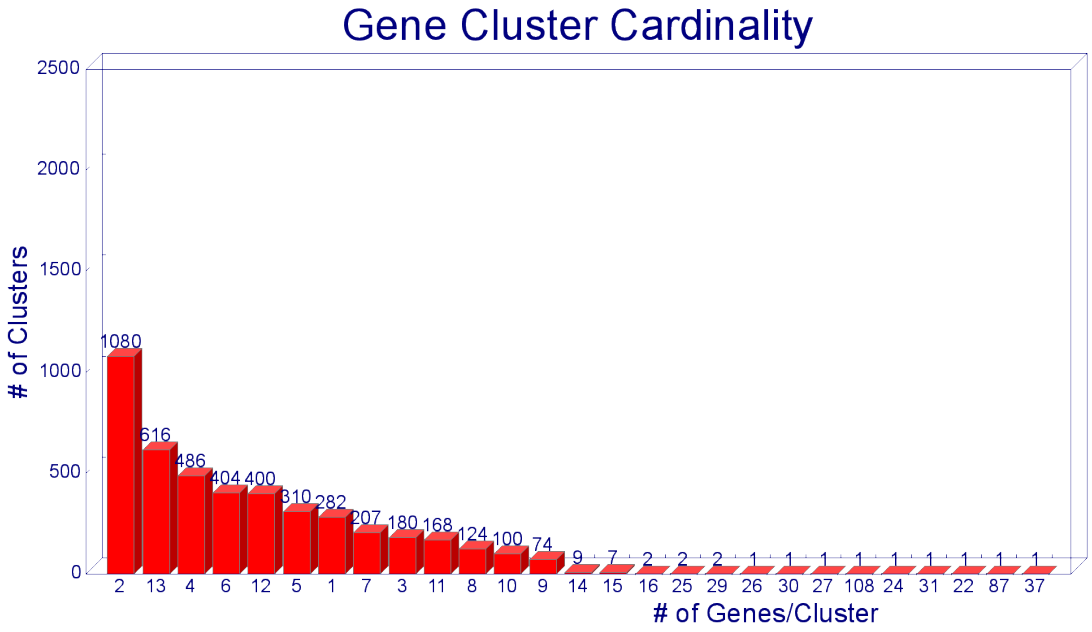

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster
1000
800
600
400
200
0
900 718 369 361 354 327 242 207 185 176 146 138 102 23 13 10 8 6 6 5 5 4 4 4 2 2 2 2 2 1 1 1 1 1 1 1 1
13 2 12 4 1 11 5 6 3 7 10 8 9 14 15 22 24 16 23 18 19 21 17 25 27 28 31 35 36 33 26 88 20 64 37 117 43

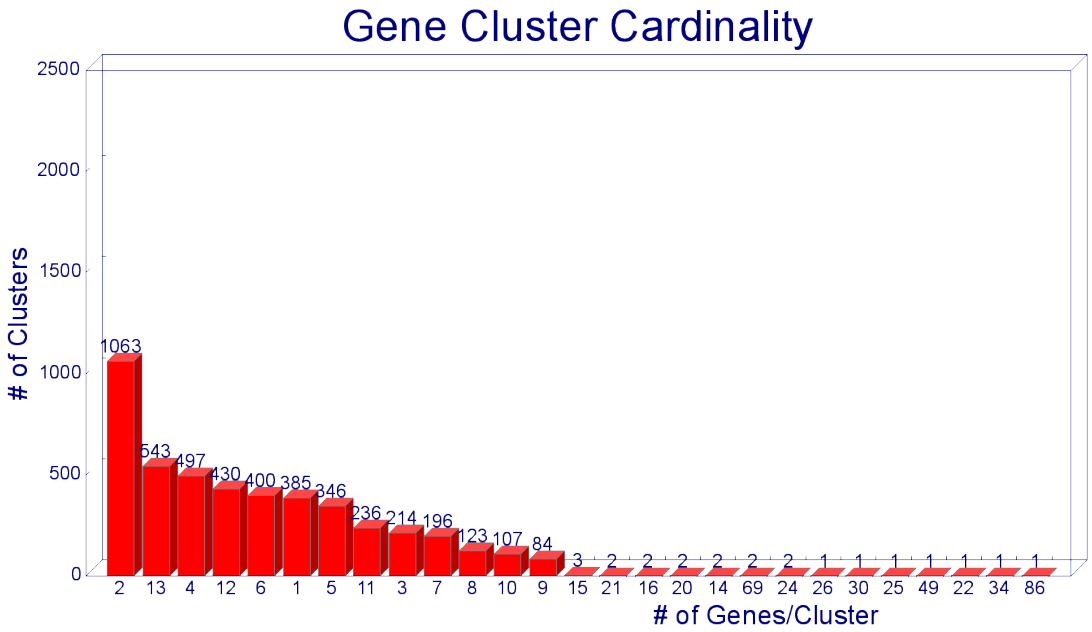

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster
1000
800
600
400
200
0
729 706 517 461 423 361 260 248 237 204 186 143 123 21 8 7 6 5 4 3 3 3 2 2 2 2 2 1 1 1 1 1 1 1 1 1
13 2 1 12 11 4 5 3 6 10 7 8 9 14 15 23 21 18 22 17 20 19 16 25 27 28 24 33 26 74 31 35 48 77 29 65 34

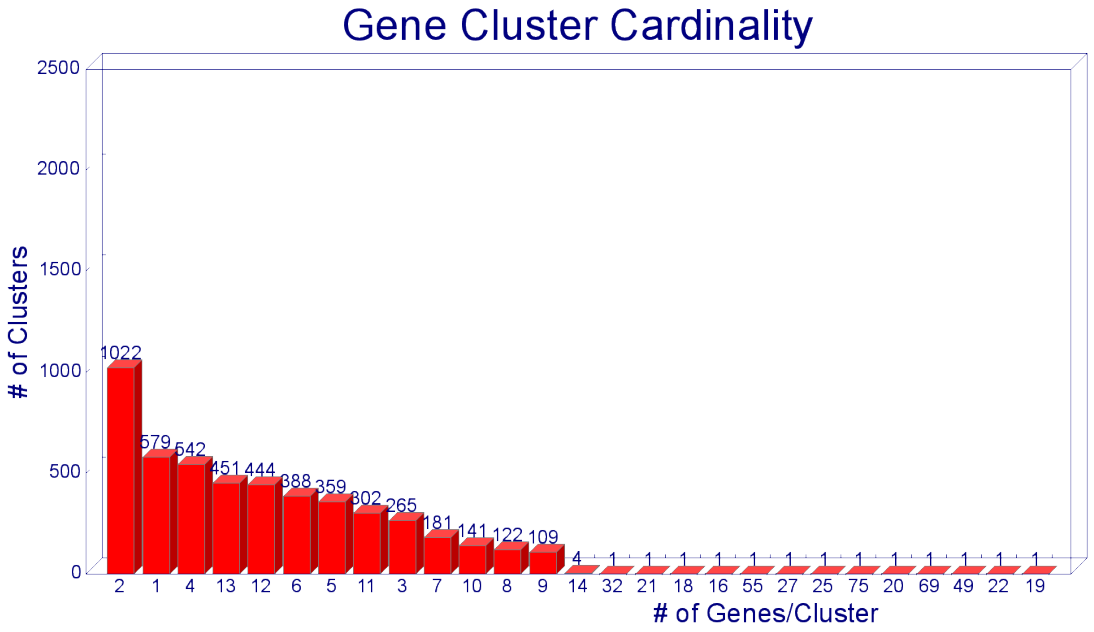

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster
1000
800
600
400
200
0
913 824 393 337 271 266 243 196 190 161 131 100 81 22 9 6 4 3 3 3 3 2 2 2 2 2 1 1 1 1 1 1 1 1 1
13 2 4 12 1 5 6 11 7 3 8 10 9 14 16 15 17 18 20 24 22 21 27 25 31 37 33 139 26 30 28 89 29 39 19

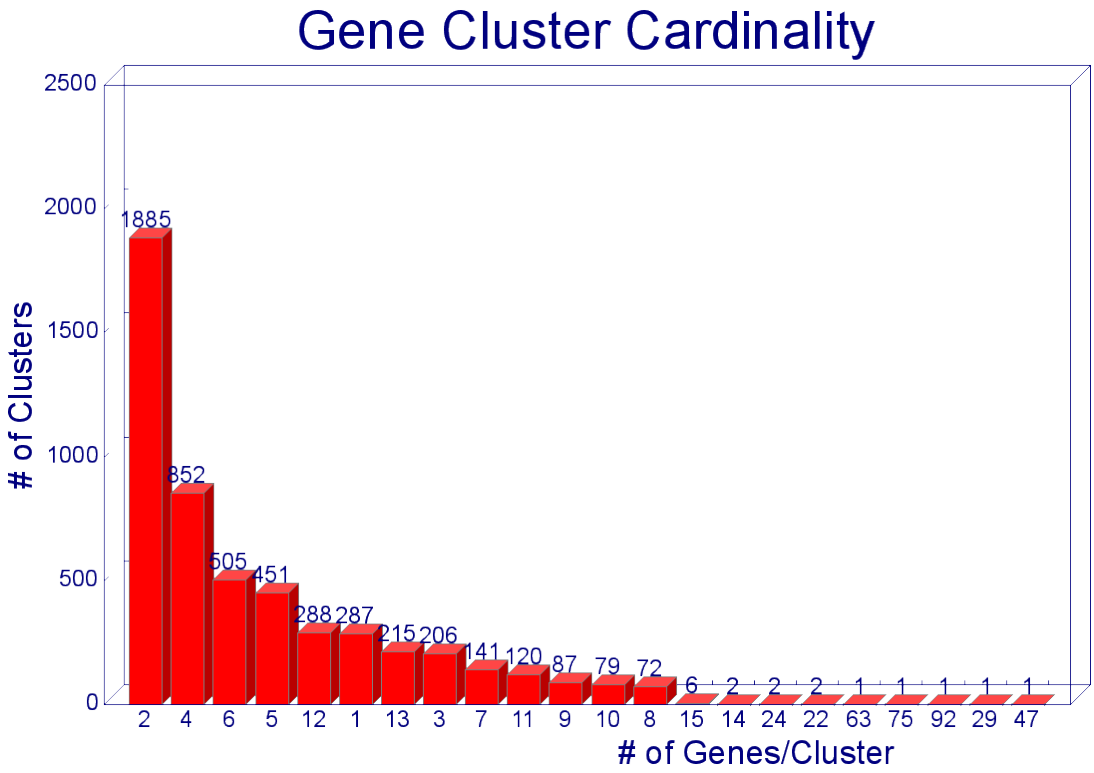

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster
1000
800
600
400
200
0
812 812 406 405 374 285 270 255 206 179 130 128 95 11 7 5 4 3 3 2 2 2 2 2 1 1 1 1 1 1 1 1
2 13 12 4 1 11 5 6 3 7 10 8 9 14 15 16 22 21 24 26 17 25 28 19 18 88 31 35 39 111 60 37

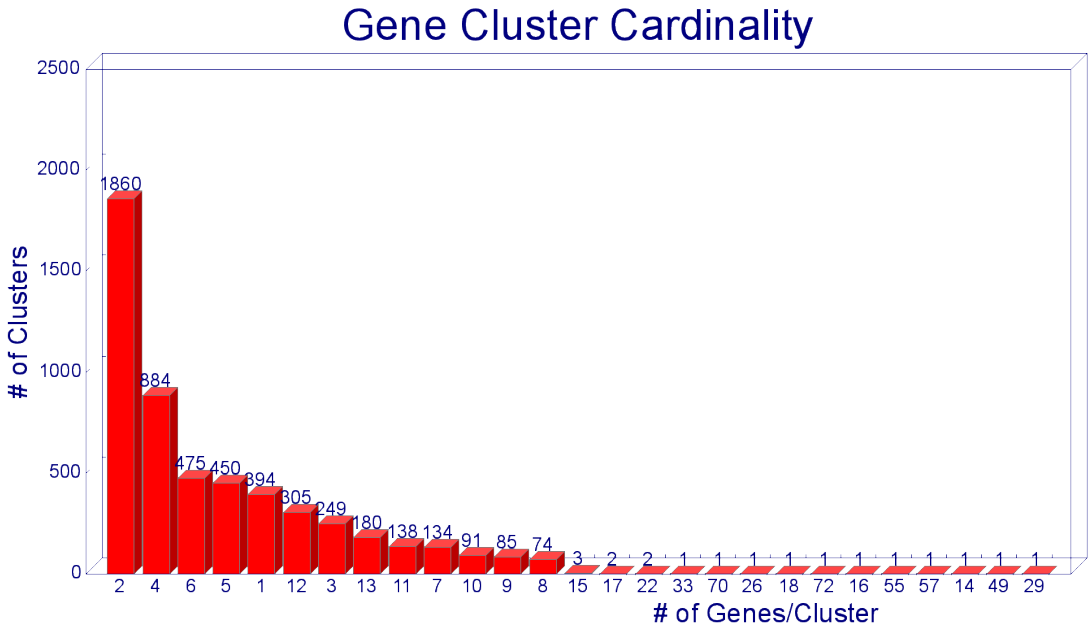

Gene Cluster Cardinality
# of Clusters
# of Genes/Cluster
1000
800
600
400
200
0
777 664 535 473 414 385 282 262 262 193 170 129 121 8 5 4 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
2 13 1 12 4 11 5 6 3 10 7 8 9 14 15 21 26 17 18 16 28 22 32 74 20 24 48 23 29 65 19 76

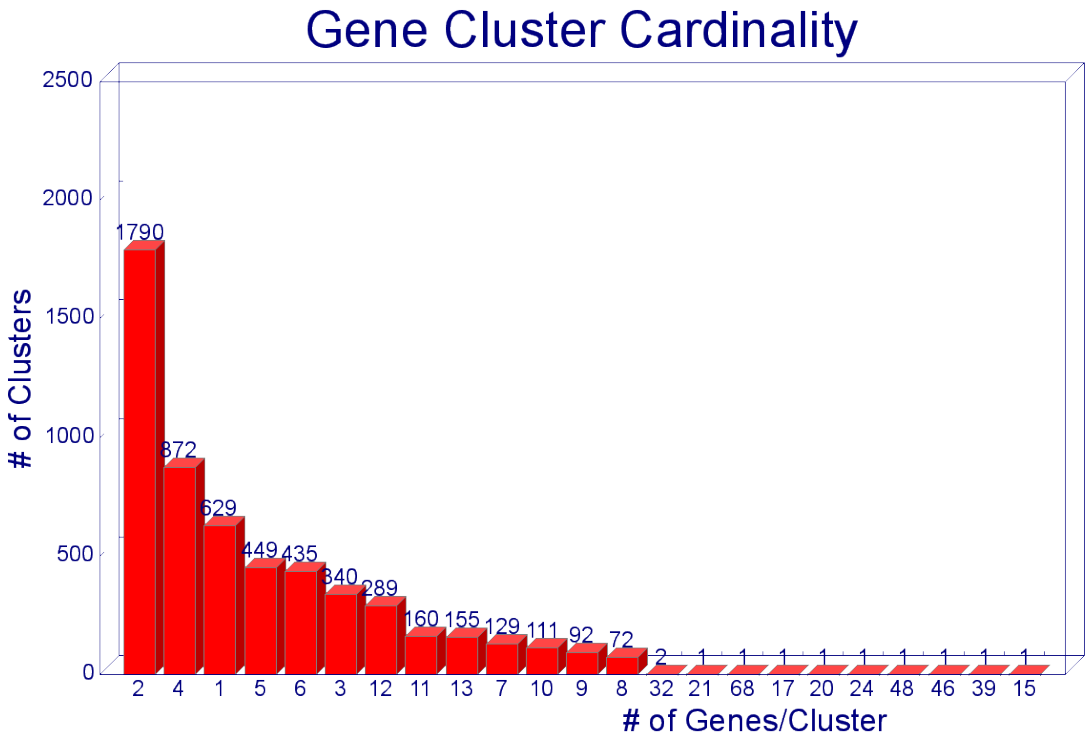

Gene Cluster Cardinality
# of Clusters
2500
2000
1500
1000
500
0
1080 616 486 404 400 310 282 207 180 168 124 100 74 9 7 2 2 2 1 1 1 1 1 1 1 1 1
2 13 4 6 12 5 1 7 3 11 8 10 9 14 15 16 25 29 26 30 27 108 24 31 22 87 37
# of Genes/Cluster


Gene Cluster Cardinality
# of Clusters
2500
2000
1500
1000
500
0
1063 543 497 430 400 385 346 236 214 196 123 107 84 3 2 2 2 2 2 2 1 1 1 1 1 1 1
2 13 4 12 6 1 5 11 3 7 8 10 9 15 21 16 20 14 69 24 26 30 25 49 22 34 86
# of Genes/Cluster

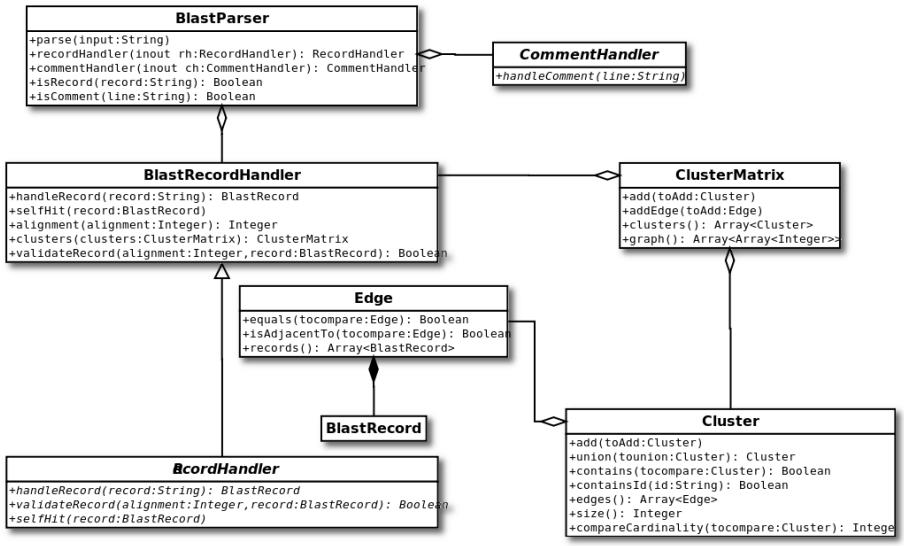

Gene Cluster Cardinality
# of Clusters
2500
2000
1500
1000
500
0
1022
579
542
451
444
388
359
302
265
181
141
122
109
4
1
1
1
1
1
1
1
1
1
1
1
1
1
2 1 4 13 12 6 5 11 3 7 10 8 9 14 32 21 18 16 55 27 25 75 20 69 49 22 19
# of Genes/Cluster


Gene Cluster Cardinality
# of Clusters
2500
2000
1500
1000
500
0
1885
852
505
451
288
287
215
206
141
120
87
79
72
6
2
2
2
1
1
1
1
1
2 4 6 5 12 1 13 3 7 11 9 10 8 15 14 24 22 63 75 92 29 47
# of Genes/Cluster


Gene Cluster Cardinality
# of Clusters
2500
2000
1500
1000
500
0
1860
884
475
450
394
305
249
180
138
134
91
85
74
3
2
2
1
1
1
1
1
1
1
1
1
1
2 4 6 5 1 12 3 13 11 7 10 9 8 15 17 22 33 70 26 18 72 16 55 57 14 49 29
# of Genes/Cluster


Gene Cluster Cardinality
# of Clusters
2500
2000
1500
1000
500
0
1790
872
629
449
435
340
289
160
155
129
111
92
72
2
1
1
1
1
1
1
1
1
1
2 4 1 5 6 3 12 11 13 7 10 9 8 32 21 68 17 20 24 48 46 39 15
# of Genes/Cluster

Classes


BlastParser
+parse(input:String)
+recordHandler(inout rh:RecordHandler): RecordHandler
+commentHandler(inout ch:CommentHandler): CommentHandler
+isRecord(record:String): Boolean
+isComment(line:String): Boolean
CommentHandler
+handleComment(line:String)
BlastRecordHandler
+handleRecord(record:String): BlastRecord
+selfHit(record:BlastRecord)
+alignment(alignment:Integer): Integer
+clusters(clusters:ClusterMatrix): ClusterMatrix
+validateRecord(alignment:Integer,record:BlastRecord): Boolean
ClusterMatrix
+add(toAdd:Cluster)
+addEdge(toAdd:Edge)
+clusters(): Array<Cluster>
+graph(): Array<Array<Integer>>
Edge
+equals(tocompare:Edge): Boolean
+isAdjacentTo(tocompare:Edge): Boolean
+records(): Array<BlastRecord>
BlastRecord
RecordHandler
+handleRecord(record:String): BlastRecord
+validateRecord(alignment:Integer,record:BlastRecord): Boolean
+selfHit(record:BlastRecord)
Cluster
+add(toAdd:Cluster)
+union(tounion:Cluster): Cluster
+contains(tocompare:Cluster): Boolean
+containsId(id:String): Boolean
+edges(): Array<Edge>
+size(): Integer
+compareCardinality(tocompare:Cluster): Intege

1 Class `BlastParser`

Description

Class mainly responsible for parsing blast input into `BlastRecord` instances.

Author: *Leo Przybylski (przybyls@arizona.edu)*

Default Constructor

Constructs the `BlastParser` from its attributes. A `RecordHandler` is required. If the `CommentHandler` is not provided, a default is used.

Parameters

`rh` - a `RecordHandler` instance

`ch` - a `CommentHandler` instance

Method `parse`

Opens the file directed by the provided filename and parses it into `BlastRecord` instances. It then passes the `BlastRecord` instances to the `RecordHandler`

Parameters

`input` - path of the file to parse

Method `parseRecord`

Takes a line from the blast output file and parses it into an array of fields

Parameters

a record from the blast output file

Method `isRecord`

Determines if the given line is indeed a record. If it's not a record, it's probably a comment

Parameters

a record from the blast output file

Method `isComment`

Determines if the given line is indeed a comment. If it's not a comment, it's probably a record

Parameters

a record from the blast output file

Getter/Setter `commentHandler`

Getter/Setter for the commentHandler

Parameters

`commentHandler` **to set (optional)**

Returns

Gets the `commentHandler`. Only returns something if there is no parameter present.

Getter/Setter `recordHandler`

Getter/Setter for the recordHandler

Parameters

`recordHandler` **to set (optional)**

Returns

Gets the `recordHandler`. Only returns something if there is no parameter present.

2 Class `BlastRecordHandler`

Description

```
Allows for different types of record handling of Blast output. Used
as an adapter passed to the BlastParser for different handling of
blast information.
```

Author: ***Leo Przybylski (przybyls@arizona.edu)***

Inherits From: `RecordHandler`

Default Constructor

Constructs a `BlastRecordHandler` from its attributes

Parameters

a `ClusterGraph`. When it handles a record, an `Gene` is added to the `ClusterGraph`. This makes the `BlastRecordHandler` stateful.

Method `handleRecord`

Creates a `BlastRecord` and handles it.

Parameters

`record` - an array of fields used to populate a `BlastRecord`

Method `isSelfHit`

Handles *self hit* blast records

Parameters

`record` - `BlastRecord` instance

Getter/Setter `graph`

Getter/Setter for the cluster graph.

Parameters

`graph` to set (optional)

Returns

Gets the `graph`. Only returns something if there is no parameter present.

Getter/Setter `current`

Getter/Setter for the current cluster.

Parameters

`current` **to set (optional)**

Returns

Gets the `current`. Only returns something if there is no parameter present.

Getter/Setter `alignments`

Getter/Setter for the alignment length requirements hash. Each alignment length is stored with the query id as the key.

Parameters

`alignments` **to set (optional)**

Returns

Gets the `alignments`. Only returns something if there is no parameter present.

Method `validate`

Validates a `BlastRecord` or `Gene` using the self hit alignment information. If this record is valid, we can use that information to determine if it is an gene or not.

A valid `BlastRecord` has a % `identity` larger than that of the requirement. The % `identity` requirement is determined at the point when the `ClusterGraph` instance is created. That is, the `ClusterGraph` knows what the requirement is. The same goes for the `alignment` ratio requirement. The `ClusterGraph` also knows what that is. The `alignment` ratio is determined by the record alignment/self hit alignment. In order to obtain the self hit for a given record, it is regarded that the `Cluster` the `BlastRecord` belongs in has an `Gene` somewhere with a subject that is the same as the `BlastRecord`'s query which would make its query and subject the same (a self hit.)

Take note that this only works if the `Cluster` that the `BlastRecord` belongs to has a self hit. If there isn't one, then we just say it's valid. When the self hit is discovered, this `BlastRecord` will be re-evaluated.

Parameters

`record` **- The** `BlastRecord` **or** `Gene` **to validate**

Returns

`1` if the record is valid, `0` otherwise.

Method `clusterForRecord`

Lookup the `Cluster` belonging to a `BlastRecord`. Uses both `query` and `subject` properties of the `BlastRecord`

Parameters

`record` **- The** `BlastRecord` **or** `Gene` **to lookup a** `Cluster` **for**

Returns

A `Cluster`

3 Class `BlastRecord`

Description

Class representation of line items from blast output. `BlastRecord` instances have

query id
subject id
identity
alignment length

Author: *Leo Przybylski (przybyls@arizona.edu)*

Default Constructor

Constructs the `BlastRecord` from its attributes. None are required though.

Parameters

`query` **id**
`subject` **id**
`identity`
`alignment` **length**
`mismatches` **- undetermined**
`qstart` **- undetermined**
`qend` **- undetermined**
`sstart` **- undetermined**
`send` **- undetermined**
`evalue` **- undetermined**

Method `isSelfHit`

A Record is considered a *self hit* if its query and subject are the same. This method compares them and returns the results. It's case-sensitive.

Returns

`1` if is *self hit*; otherwise, returns `0`

Getter/Setter `query`

Getter/Setter for the query

Parameters

`query_id` **to set (optional)**

Returns

Gets the `query_id`. Only returns something if there is no parameter present.

Getter/Setter `subject`

Getter/Setter for the subject

Parameters

`subject_id` to set (optional)

Returns

Gets the `subject_id`. Only returns something if there is no parameter present.

Getter/Setter `identity`

Getter/Setter for the identity

Parameters

`identity` to set (optional)

Returns

Gets the `identity`. Only returns something if there is no parameter present.

Getter/Setter `alignment`

Getter/Setter for the alignment

Parameters

`alignment` to set (optional)

Returns

Gets the alignment. Only returns something if there is no parameter present.

Getter/Setter `mismatches`

Getter/Setter for the mismatches

Parameters

`mismatches` to set (optional)

Returns

Gets the mismatches. Only returns something if there is no parameter present.

Getter/Setter `qstart`

Getter/Setter for the qstart

Parameters

`qstart` to set (optional)

Returns

Gets the qstart. Only returns something if there is no parameter present.

Getter/Setter `qend`

Getter/Setter for the qend

Parameters

`qend` to set (optional)

Returns

Gets the qend. Only returns something if there is no parameter present.

Getter/Setter `sstart`

Getter/Setter for the sstart

Parameters

`sstart` to set (optional)

Returns

Gets the sstart. Only returns something if there is no parameter present.

Getter/Setter `send`

Getter/Setter for the send

Parameters

`send` to set (optional)

Returns

Gets the send. Only returns something if there is no parameter present.

Getter/Setter `evalue`

Getter/Setter for the evalue

Parameters

`evalue` to set (optional)

Returns

Gets the evalue. Only returns something if there is no parameter present.

4 Class `Cluster`

Description

A cluster is basically like a set genes in a digraph of genes where adjacent genes are grouped together. One Gene is known to be adjacent to another gene if its query points to the subject or query of another or its subject points to the query or subject of another. ...

Being that a Cluster is a Set, there is no duplication

Author: ***Leo Przybylski (przybyls@arizona.edu)***

Method `add`

Adds an `Gene` instance to the `Cluster`

Parameters

`toadd` - `Gene` instance to add

Method `union`

Unions this `Cluster` instance with another `Cluster` instance. The result is a completely new `Cluster`.

Parameters

`other` - a `Cluster` instance to union with this

Returns

A new `Cluster` instance containing all `Gene` instances from both `Cluster` instances. If a union is not possible, nothing is returned

Method `contains`

Traverses the `Cluster` for a given `Gene`

Parameters

`tocompare` - `Gene` to test for existence

Returns

`1` if `$tocompare` is contained in the `Cluster`; `0` otherwise.

Method `indexOf`

Traverses the `Cluster` looking for the array index of the `Gene`

Parameters

`tocompare` - `Gene` to find

Returns

`index` integer location of `$tocompare` within the `Cluster` array of `Gene` instances; `-1` otherwise.

Method `remove`

Locates and removes an `Gene` from the `Cluster`. This probably only happens when an `Gene` has been found to be invalid.

Parameters

`tocompare` - `Gene` to locate and remove

Method `containsId`

Parameters

`tocompare` -

Returns

`1` if `$tocompare` is contained in the `Cluster`; `0` otherwise.

Getter `graph`

Returns

A reference to an array instance containing `Gene` instances for this `Cluster`

Getter `genes`

Returns

A reference to an array instance containing `Gene` instances for this `Cluster`

Getter `ids`

Returns

A reference to an array instance containing `Gene` instances for this `Cluster`

Getter `size`

Returns

The number of `Gene` instances that are part of this `Cluster`

Method `hasAdjacentGene`

Compares this `Cluster` to another to see if the two might have genes that are adjacent to each other

Parameters

`tocompare` - a `Cluster` whose `Gene` instances to compare to this one for adjacency

Returns

`1` if `tocompare` shares at least 1 adjacent gene with this `Cluster` instance or `0` if it doesn't.

Method `hasGeneAdjacentTo`

Compares `Gene` instances in this `Cluster` to another to see if the the other is adjacent to any `Gene` instances in this cluster.

Parameters

`tocompare` - a `Gene` who compare to others in this `Cluster` for adjacency

Returns

`1` if `tocompare` shares at least 1 adjacent gene with this `Cluster` instance or `0` if it doesn't.

Method `compareCardinality`

Compares the cardinality (the number of `Gene` instances) of this `Cluster` instance to another `Cluster` instance.

Parameters

`tocompare` - a `Cluster` instance to compare this against

Returns

`-1` if this `Cluster` is smaller in cardinality than `tocompare`

`0` if this `Cluster` is shares the same cardinality as `tocompare`

`1` if this `Cluster` is larger in cardinality than `tocompare`

Getter `geneByHit`

Gets an `Gene` from the graph by the subject id. It will iterate through the genes until it finds one with the subject id it's looking for.

Parameters

`subject` id of the `Gene` to find

Returns

An `Gene` instance

5 Class `CommentHandler`

Description

An interface for handling comments in blast output.

Author: *Leo Przybylski (przybyls@arizona.edu)*

Default Constructor

Constructs the `CommentHandler` from its attributes. None are required though.

Method `handleComment`

Stub method for handling comments. This is the method that the `BlastParser` will call when it encounters a comment.

6 Class `RecordHandler`

Description

Abstract class for creating instances used by the BlastParser for handling records.

Author: *Leo Przybylski (przybyls@arizona.edu)*

Default Constructor

Constructs the `RecordHandler` from its attributes. None are required though.

Method `handleRecord`

Stub method for handling blast output records. This is the method that the `BlastParser` will call when it encounters a record.

Method `handleRecord`

Stub method for validating blast output records. Typically, the result will be an `Edge` added to a `Cluster`