General Purpose Bucketizer Design

Author: Dave Miers

Created: May 28, 2003

Last Modified: May 28, 2003 08:16

1 Purpose of Document

The purpose of this document is to provide a design for a general purpose “bucketizer” to compare clusters. For the purposes of this document, “comparing clusters” means comparing the sequence ID membership of one cluster to another. It does not mean that it is comparing that actual sequences.

2 Introduction

There has been a repeated need for an algorithm that compares sets of clusters and determines what type of relationship the clusters have based on their cluster members. Depending on the cluster relationship, the clusters are written to one of the following buckets:

- **0 to 1** - Contains clusters from the second cluster set that are not associated with any clusters in the first cluster set.
- **1 to 0** - Contains clusters from the first cluster set that are not associated with any clusters in the second cluster set.
- **1 to 1** - Contains clusters from each cluster set that are only associated with each other.
- **1 to N** - Contains clusters from the first cluster set that are associated with multiple clusters from the second cluster set.
- **N to 1** - Contains clusters from the second cluster set that are associated with multiple clusters from the first cluster set.
- **N to N** - Contains multiple clusters from the first cluster set that are associated with multiple clusters from the second cluster set.

The general purpose bucketizer will compare two sets of clusters and separate them into buckets according to the relationships defined above.

3 Definitions

The following definitions describe some of the terminology used in this document:

- **Bucketizer** - The algorithm used to separate the clusters into buckets based on their relationships with each other.
- **Cluster** - A group of sequence IDs that are related to each other. The cluster itself also has an ID that is known as the cluster ID.

- **Cluster Member** - One of the sequences that make up a cluster.
- **Cluster Pair** - Two clusters from different cluster sets that have a relationship with each other based on their cluster members.
- **Cluster Set** - A group of clusters from a particular data source (file or database) that is used as input to the bucketizer.

4 Invocation

The bucketizer will be a function in a Python library that can be imported by other scripts that need the functionality. Depending on what input sources are used, the bucketizer can be called in any of the following ways:

4.1 Both input sources are files

```
bucketize(file1=input_filename, file2=input_filename, prefix=output_filename)
```

4.2 Source 1 is a file and source 2 is the database

```
bucketize(file1=filename,
          table2=tablename, cid2=cluster_ID_column, cmid2=cluster_member_ID_column,
          prefix=output_filename)
```

4.3 Source 1 is the database and source 2 is a file

```
bucketize(table1=tablename, cid1=cluster_ID_column, cmid1=cluster_member_ID_column,
          file2=filename,
          prefix=output_filename)
```

4.4 Both input sources are the database

```
bucketize(table1=tablename, cid1=cluster_ID_column, cmid1=cluster_member_ID_column,
          table2=tablename, cid2=cluster_ID_column, cmid2=cluster_member_ID_column,
          prefix=output_filename)
```

5 Inputs

The bucketizer needs two sources of input, representing the sets of clusters. Each input source can be a file or a database table.

5.1 File Input

When a file is used to input a set of clusters, each record in the file should be formatted as follows:

```
Cluster-ID<TAB>Cluster-Member-ID
```

The name of the file is provided to the bucketizer as a parameter. There should be no duplicate records in the file.

5.2 Database Input

When a database table is used to input a set of clusters, the following information is provided to the bucketizer as parameters:

- Table name
- Cluster-ID column name
- Cluster-ID member column name

6 Outputs

The bucketizer will produce one output file for each type of cluster relationship. The prefix that is used for each file name is provided to the bucketizer as a parameter. The suffix for each file name is based on the cluster relationship. If the prefix is “bucket”, the file names are as follows:

Table 1: Output File Names

Cluster Relationship	Bucket File Name
0:1	bucket.0to1
1:0	bucket.1to0
1:1	bucket.1to1
1:N	bucket.1toN
N:1	bucket.Nto1
N:N	bucket.NtoN

If there are no clusters that satisfy a particular relationship, that bucket file will be created, but it will be empty.

7 Processing

The bucketizer will perform the following steps to create the output files based on the records in the input sources:

1. If the first input source is a file, load the cluster ID/cluster member ID pairs into a temporary table. This table is referred to as **cluster set 1**. If the first input source is already a table, it is referred to as **cluster set 1**.

2. If the second input source is a file, load the cluster ID/cluster member ID pairs into a temporary table. This table is referred to as **cluster set 2**. If the second input source is already a table, it is referred to as **cluster set 2**.
3. Open the output files in write mode.
4. Find all clusters in cluster set 2 that are not associated with any clusters in cluster set 1. Write these clusters, along with their cluster members, to the output file for **0:1** relationships.
5. Find all clusters in cluster set 1 that are not associated with any clusters in cluster set 2. Write these clusters, along with their cluster members, to the output file for **1:0** relationships.
6. Join cluster set1 and cluster set 2 on the cluster member ID columns, with the cluster IDs from each table going into separate columns of another temporary table. This table is referred to as the **cluster mapping**.
7. Find all records in the cluster mapping where both cluster IDs are unique. Write these clusters pairs to the output file for **1:1** relationships and delete them from the cluster mapping.
8. Find all records in the cluster mapping where a cluster ID in the first column maps to multiple cluster IDs in the second column. Write these cluster pairs to the output file for **1:N** relationships and delete them from the cluster mapping.
9. Find all records in the cluster mapping where a cluster ID in the second column maps to multiple cluster IDs in the first column. Write these cluster pairs to the output file for **N:1** relationships and delete them from the cluster mapping.
10. Write all remaining cluster pairs in the cluster mapping to the output file for **N:N** relationships.
11. Close all the output files.

8 Assumptions

The bucketizer will make the following assumptions when it is called:

1. If one or both of the input sources is a database table, the calling script should have already established a connection to the database. In addition, the function **db.useOneConnection(1)** should have been used to insure that the connection is not closed after each call to **db.sql()**. Otherwise, temporary tables will be lost between processing steps.
2. If an input source is a file, it should not contain any duplicate records.