Network Calculation Documentation

Sarah Read

2/14/2016

1 Introduction

This document outlines the files used to create the river network tree out of the data, calculate the river network value, and calculate the flow in and out of each habitat. The original data was transformed to make the original nodes the edges of the new graph and the original edges the nodes of the new graph. All original labels are maintained, meaning that each habitat/river segment has the same label in both graphs as do the barriers.

2 Files

The following files are used in the data tranformation:

- TransformData.java: Java file that takes the 3 original data files and puts them in the correct JSON format. The edges in the original data represent habitats and the nodes represent barriers and locations where river segments converge. In the final tree the edges represent barriers and converging river segments while the nodes represent habitas.
- Deerfield_barriers_coordinates_10_16_15.txt: Original file that holds all barrier information.
- Deerfield_edges_10_16_15.txt: Original file that holds the habitat information.
- nodes_no_loops_barriers.txt: Original file that holds all of the node information.
- transformedData.txt: Output of the TransformData.java file, which holds the JSON representation of the transformed tree.

The following files are used in the network calculations:

- RiverNetworkCalculations.js: Contains all of the functions to calculate the network values, including the flow into and out of each node and the total network value. This file also contains the functions to change barrier passage values.
- transformedTree.json: This file contains the JSON representation of the river network.

3 RiverNetworkCalculations.js

The following functions are meant to be called by an outside program to calculate different river network values:

- readTree(): Takes the JSON representation and creates the tree and nodes. Then the network value are computed. This function must be called before any other functions. No parameters. No return.
- getNetworkInformation(): Creates JSON that holds the total network value, the flow in for each node, and the flow out for each node. No parameters. Returns a string.
- updateBarriers(newBarriers): Updates the barrier passage scores that are passed in. The parameter is a 2-D array. Each row contains information about the barrier that needs to be updated, formated as [barrier id, upstream passage score, downstream passage score]. No return.
- updateChildAndParentForTotalNetworkCalculations(child, parent): A child-parent pair is requied to calculate the value of the river network. Any child-parent pair can be used except if the parent is the root. The parameters are the child and parent labels. No return.

All other functions in this file are not meant to be called by outside files. These functions either calculate the alpha, beta, and gamma messages, are helper functions, or are used to display values on a very simple UI (meant for small trees). It is important to note that the transformedTree.json file will have to be included so that the readTree function can load the data.

4 River Network Information JSON

The string returned by the getNetworkInformation function is formatted in the following way:

```
{
    networkValue: 10,
    flowInto: [[a,2],[b,5], ... ],
    flowOut: [[a,4],[b,1], ... ]
}
```

Where a and b are habitat ids.