Euler Tour Tree Representation

Robert Bakarić
rbakaric@irb.hr
bakaric@evolbio.mpg.de
01.04.2015
EulerTour-1.0

Abstract

The Euler tour tree representation is a program for representing trees (including subtrees) as euler circuts of directed graphs produced by converting a tree (undirected graph) into a directed graph. The procedure is achieved by replacing each undirected edge in a tree with two directed ones. As a result a list of vertices is computed as they are visited during the traversal of one such graph. This program is implemented in C++.

Contents

1 Installation

The simplest way to compile this program is to:

1. Unpack the EulerTour package (`eulertour-XXX.tar.gz`):

   ```
   tar -xvzf eulertour-XXX.tar.gz
   ```

2. Change the current directory to `eulertour-XXX`:

   ```
   cd eulertour-XXX/
   ```

3. Configure the program for your system (–bindir is optional):

   ```
   ./configure --bindir=/absolute/directory/path/eulertour-xxx/bin
   ```

4. Compile the program:

   ```
   make
   ```

5. Install the program:

   ```
   make install
   ```

Your binaries should be located in your local bin directory if `--bindir` option has been set. Otherwise installation needs to be carried out with root privileges in order to be installed into `/usr/local/bin` directory.

2 Input files

The EulerTour takes a simple two column (tab separated) file containing integers. An example of the input file can be found in `./eulertour-xxx/demo` and it should look like this:

`Graph`:

```
1     0
2     1
4     1
8     2
5     1
67    2
6     3
14    4
15    6
68    3
3     2
11    67
17    67
```

3 Program options

I order to see program options type:

```
./bin/EulerTour -h
```

Expected output:

```
Usage: ./program [options]

*******************************************************************************
                        EulerTour - Euler Tour Tree Representation
                                            by
                                      Robert Bakaric

CONTACT:
 Code written and maintained by Robert Bakaric,
 email: rbakaric@irb.hr , bakaric@evolbio.mpg.de

ACKNOWLEDGEMENT:


LICENSE:
 The program is distributed under the GNU General Public License. You should have
 received a copy of the licence together  with this software. If not, see
 http://www.gnu.org/licenses/
*******************************************************************************


Allowed options:
  -h [ --help ]           produce help message
  -v [ --version ]        print version information
  -i [ --input-file ] arg input file
```

4 Functions and classes

Graph class:

`Graph` : Graph class.

`make` : Explicit constructor.

`destroy` : Explicit destructor.

`MakeGraph` : Function converts two given vectors into a graph data structure representation.

Euler Tour class:

`EulerTour` : EulerTour - Euler Tour Tree Representation class.

`make` : Explicit constructor.

`destroy` : Explicit destructor. Destroys the local information and graph container.

`Traverse` : Given a starting vertex (element of the tree nodes), function computes the tour. It computes a set of vertices as they are visited during the traversal and the distance (depth) values of each vertex from the starting point.

`Clean` : Function is used for erasing results computed with `Traverse` function

`GetVertexIdVec` : Function returns the computed vertices as they were visited during the traversal.

`GetDepthVec` : Function returns the computed distance values of vertices as they were visited during the traversal.

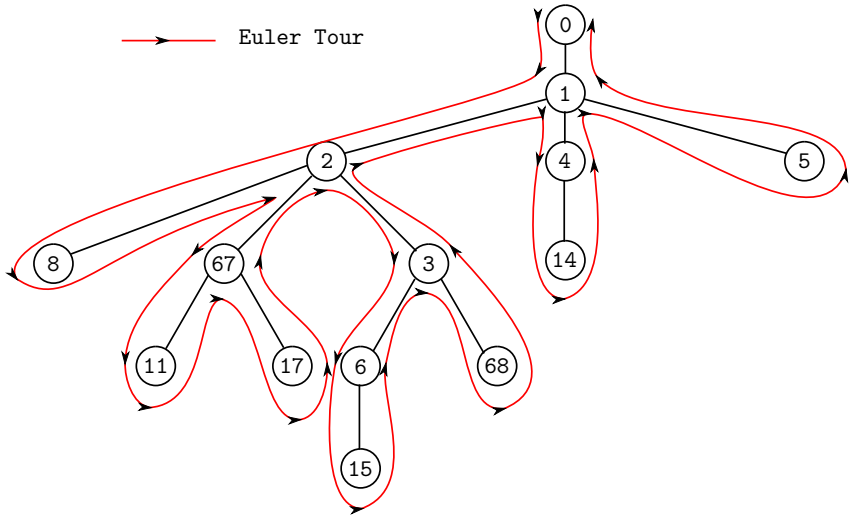

Figure 1: Euler tour tree computation. The tour is computing by making a graph traversal as illustrated by the red line.

5 Example

5.1 EulerTour.cpp

A minimal example demonstrating the usage of EulerTour demo program:

```
./bin/EulerTour -i demo/Graph.txt
NOTE: Ctrl-c to quit!

List of vertices:
0 1 2 3 4 6 67
Choose start vertex:0


Euler Tour:

Depth:    0 1 2 3 2 3 4 3 4 3 2 3 4 5 4 3 4 3 2 1 2 3 2 1 2 1 0

VertexId: 0 1 2 8 2 67 11 67 17 67 2 3 6 15 6 3 68 3 2 1 4 14 4 1 5 1 0

NOTE: Ctrl-c to quit!

List of vertices:
0 1 2 3 4 6 67
Choose start vertex:3


Euler Tour:

Depth:    0 1 2 1 0 1 0

VertexId: 3 6 15 6 3 68 3
```

5.2 EulerTour.hpp

Adding the `EulerTour.hpp` header file to your program will allow you to include all the functions described in section 4. A minimal example:

```
 #include<vector>
#include<EulerTour.hpp>

vector<int|long|unsigned|double> parent {0,1,1,2,1,2,3,4,6,3,2,67,67};
vector<int|long|unsigned|double> child  {1,2,4,8,5,67,6,14,15,68,3,11,17};

/* Make Graph */

    /* Construction */
    Graph<int|long|unsigned|double> graph(parent,child);
    /*  OR  */
    Graph<int|long|unsigned|double> graph;
    graph.make(parent,child);

/*  Make ET  */

    /* Construction */
    EulerTour<int|long|unsigned|double> et(0,parent,child);
    /*  OR  */
    EulerTour<int|long|unsigned|double> et(parent,child);
    /*  OR  */
    EulerTour<int|long|unsigned|double> et;
    et.make(0,parent,child);
      /*  or  */
    et.make(parent,child);

/*  Functions */

    et.Traverse(2)        // traverses the tree with 2 as a start and stop point

    et.Clean()            // cleans the treversed path (both dept and value)

    et.GetDepthVec()      // returns the depth vector
                          //    result for 2:      0 1 0 1 2 1 2 1 0 1 2 3 2 1 2 1 0
    et.GetVertexIdVec()   // returns the set of vertices (vector)
                          //    result for 2:      2 8 2 67 11 67 17 67 2 3 6 15 6 3 68 3 2
```

6 Acknowledgement

7 Future work

1. Implement Euler path (circuit) computation algorithm for general graphs.