Think Before You Discard:

Accurate Triangle Counting in Graph Streams with Deletions

Kijung Shin (kijungs@cs.cmu.edu)

March-12-2018

1 General Information

- Version: 1.0
- Date: March-12-2018
- Authors: Kijung Shin (kijungs@cs.cmu.edu)

2 Introduction

ThinkD (Think before you **D**iscard**)** is a streaming algorithm for triangle counting in a fully dynamic graph stream with edge additions and deletions. **ThinkD** estimates the counts of global triangles and local triangles incident to each node by making a single pass over the stream. **ThinkD** has the following advantages:

- *Accurate*: **ThinkD** is up to *4.3X more accurate* than its best competitors within the same memory budget
- Fast: **ThinkD** is up to *2.2X faster* than its best competitors for the same accuracy requirements
- Theoretically Sound: **ThinkD** always maintains unbiased estimates

Detailed information about the method is explained in the following paper

- Kijung Shin, Jisu Kim, Bryan Hooi, and Christos Faloutsos, "*Think Before You Discard: Accurate Triangle Counting in Graph Streams with Deletions*", ECML/PKDD 2018 (submitted)

3 Installation

- This package requires that java 1.7 or greater be installed in the system and set in PATH.
- For compilation (optional), type *./compile.sh*
- For packaging (optional), type *./package.sh*
- For demo (optional), type *make*

4 Input File Format

The input file lists the additions and deletions in an *undirected* and *unweighted* graph in the order that they arrive. Each line corresponds to an edge addition or deletion. Each line consists of a source node id, a destination node id, and an indicator (1 for addition and 0 for deletion), which are integers separated by a tab. Additionally, we assume that there are *no parallel edges*. That is, if an edge has been added and has not been deleted yet, the same edge cannot be added.

See *example_graph.txt* for an example input file.

5 Output Files Format

Two output files are created for each trial:

- *global(trial#).txt*: this file has the estimated number of global triangles.
- *local(trial#).txt*: this file lists the estimated number of local triangles of each node. Each line consists of the node id and the number of its local triangle count, separated by a tab.

output_fast directory contains example output files.

6 Running ThinkDFast (Batch Mode)

6.1 How to Run

```
./run_fast.sh input_path output_path sampling_ratio number_of_trials
```

6.2 Parameters

- *input_path*: path of the input file. See 4 for the detailed format of the input file
- *output_path*: path of the directory for output files. See 5 for the detailed format of the output files
- *sampling_ratio*: probability that each inserted edge is sampled
- *number_of_trials*: number of trials

7 APIs for ThinkDFast (Incremental Mode)

7.1 Package: *thinkd*

7.2 Class: *ThinkDFast*

7.3 Methods:

- public *ThinkDFast* (*double sampling_ratio, int random_seed*)
    - create a ThinkDFast object

  - *sampling_ratio*: probability that each inserted edge is sampled.
  - *random_seed*: an integer
- public void *processAddition* (int src, int dst)
  - insert an edge
  - *src*: id of the source node
  - *dst*: id of the destination node
- public void *processDeletion* (int src, int dst)
  - delete an edge
  - *src*: id of the source node
  - *dst*: id of the destination node
- public double *getGlobalTriangle* ()
  - return the estimated number of global triangles
- public it.unimi.dsi.fastutil.ints.Int2DoubleMap *getLocalTriangle* ()
  - return the estimated number of local triangles of each node
  - *return*: a map whose keys are node ids and values the estimated number of local triangle counts of the corresponding node.

7.4 Example Code: see *ExampleFast.java* for an example code using ThinkDFAST.

8 Running ThinkDACC (Batch Mode)

8.1 How to Run

```
./run_acc.sh input_path output_path memory_budget number_of_trials
```

8.2 Parameters

- *input_path*: path of the input file. See 4 for the detailed format of the input file
- *output_path*: path of the directory for output files. See 5 for the detailed format of the output files
- *memory_budget*: maximum number of sampled edges (an integer greater than or equal to 2)
- *number_of_trials*: number of trials

9 APIs for ThinkDACC (Incremental Mode)

9.1 Package: *thinkd*

9.2 Class: *ThinkDAcc*

9.3 Methods:

- public *ThinkDAcc* (*int memory_budget, int random_seed*)
    - create a ThinkDAcc object
    - *memory_budget*: maximum number of sampled edges

      (an integer greater than or equal to 2)
    - *random_seed*: an integer
- public void *processAddition* (int src, int dst)
    - insert an edge
    - *src*: id of the source node
    - *dst*: id of the destination node
- public void *processDeletion* (int src, int dst)
    - delete an edge
    - *src*: id of the source node
    - *dst*: id of the destination node
- public double *getGlobalTriangle* ()
    - return the estimated number of global triangles
- public it.unimi.dsi.fastutil.ints.Int2DoubleMap *getLocalTriangle* ()
    - return the estimated number of local triangles of each node
    - *return*: a map whose keys are node ids and values the estimated number of local triangle counts of the corresponding node.

9.4 Example Code: see *ExampleAcc.java* for an example code using ThinkDAcc.