# Think before You Discard: Accurate Triangle Counting in Graph Streams with Deletions (Software User Guide)

Kijung Shin (kijungs@cs.cmu.edu)

## 1 General Information

- Version: 2.0
- Date: July 23, 2018
- Author: 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 by making a single pass over the stream. ThinkD has the following advantages:

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

Detailed information about ThinkD is explained in the following papers:

- Kijung Shin, Jisu Kim, Bryan Hooi, and Christos Faloutsos, "Think before You Discard: Accurate Triangle Counting in Graph Streams with Deletions", ECML/PKDD 2018
- Kijung Shin, Sejoon Oh, Jisu Kim, Bryan Hooi, and Christos Faloutsos, "Fast, Accurate and Provable Triangle Counting in Fully Dynamic Graph Streams", TKDD Journal (Accepted)

## 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 for ThinkD$_{\mathbf{FAST}}$ and ThinkD$_{\mathbf{ACC}}$

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 $-1$ 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 File Format for ThinkD$_{\text{FAST}}$ and ThinkD$_{\text{ACC}}$

Two output files are created for each trial:

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

The directory named *output_fast* contains example output files.

# 6 Running ThinkD$_{\text{FAST}}$ (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 Section 4 for the detailed format of the input file.
- *output_path*: path of the directory for output. files. See Section 5 for the detailed format of the output files.
- *sampling_ratio*: probability that each added edge is sampled.
- *number_of_trials*: number of trials.

# 7 APIs for ThinkD$_{\text{FAST}}$ (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 added edge is sampled.
  - *random_seed*: a non-negative 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 numbers of local triangles.
  - return: a map whose keys are node ids and values the estimated number of local triangle counts of the corresponding nodes.

### 7.4 Example Code:

See *ExampleFast.java* for an example code using *ThinkDFast*.

## 8 Running ThinkD$_{\mathrm{ACC}}$ (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 Section 4 for the detailed format of the input file
- *output_path*: path of the directory for output. files. See Section 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 ThinkD$_{\mathrm{ACC}}$ (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*: a non-negative 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 numbers of local triangles.
  - return: a map whose keys are node ids and values the estimated number of local triangle counts of the corresponding nodes.

### 9.4 Example Code:

See *ExampleAcc.java* for an example code using *ThinkDAcc*.

# 10 Input File Format for ThinkD-Spot

The input file lists the edges in an undirected and unweighted graph in the order that they arrive. Each line consists of a source-node id, a destination-node id, and a timestamp in milliseconds, which are integers separated by a tab. Additionally, we assume that there are no parallel edges. That is, the same edge cannot be repeated multiple times. See *example_graph_with_timestamps.txt* for an example input file.

# 11 Output File Format for ThinkD-Spot

The output file, named *time_to_global.txt*, lists the estimated count of global triangles at each timestamp. Each line consists of a timestamp and the estimated count of global triangles at the timestamp, separated by a tab. The directory named *output_spot* contains example output files.

# 12 Running ThinkD-Spot (Batch Mode)

## 12.1 How to Run

**./run_spot.sh input_path output_path memory_budget time_window threshold**

## 12.2 Parameters

- *input_path*: path of the input file. See Section 10 for the detailed format of the input file.
- *output_path*: path of the directory for output files. See Section 11 for the detailed format of the output file.
- *memory_budget*: maximum number of sampled edges (an integer greater than or equal to 2).
- *time_window*: size of the time window in seconds during which edges are maintained (an integer greater than or equal to 1).
- *threshold*: threshold on the estimated count of global triangles (an integer greater than or equal to 0). Timestamps where the estimated count is less than or equal to the threshold are omitted in the output file.

# 13 APIs for ThinkD-Spot (Incremental Mode)

## 13.1 Package: *thinkd*

## 13.2 Class: *ThinkDSpot*

## 13.3 Methods:

- public *ThinkDSpot* (int *memory_budget*, int *time_window*, int *threshold*, int *random_seed*)
  - create a *ThinkDSpot* object.
  - *memory_budget*: maximum number of sampled edges (an integer greater than or equal to 2).
  - *time_window*: size of the time window in seconds during which edges are maintained (an integer greater than or equal to 1).
  - *threshold*: threshold on the estimated count of global triangles (an integer greater than or equal to 0).
  - *random_seed*: a non-negative integer.
- public boolean *process* (int *src*, int *dst*, long *timestamp*)
  - process an edge and return whether the estimated global triangle exceeds the threshold after processing the edge.
  - *src*: id of the source node.
  - *dst*: id of the destination node.
  - *timestamp*: timestamp in milliseconds.

  - return: whether the estimated global triangle exceeds the threshold after processing the edge.

- public double *getGlobalTriangle*()
  - return the estimated number of global triangles.
- public *it.unimi.dsi.fastutil.ints.Int2DoubleMap getLocalTriangle*()
  - return the estimated numbers of local triangles.
  - return: a map whose keys are node ids and values the estimated number of local triangle counts of the corresponding nodes.

## 13.4 Example Code:

See *ExampleSpot.java* for an example code using *ThinkDSpot*.