Bump hunting In The Dark: Local Discrepancy Maximization on Graphs

Presented by: Zohreh Raghebi
Spring 2016:
Authors:

Aristides Gionis, Michael Mathioudakis, Antti Ukkonen

Helsinki Institute for Information Technology HIIT
Aalto University, Finland
firstname.lastname@aalto.fi

Publication:
- ICDE 2015

Type:
- Research Paper
Bump hunting is a common approach to extracting insights from data

Looking for regions of a dataset
- Where a property of interest occurs frequently

In this paper, we apply that approach on graphs

A subset of the nodes exhibit a property of interest

The goal is to find a connected subgraph (the “bump”)

Query nodes appear more often compared to non-query nodes
Maximum Discrepancy

- Discovered nodes
- Discovered edges
- The maximum discrepancy connected component
Main Idea

- We find such a subgraph by maximizing the linear discrepancy
- The possible weighted difference between the number of query and non-query nodes in the subgraph
- We consider the problem under a local-access model
- While all other nodes and edges can be discovered only via calls to a costly get-neighbors function
- For example, accessing the social graph of an online social network (e.g., Twitter) is based on such a function
[16] aims to find cardinality-constrained trees in node weighted graphs
  - With non-negative weights
Maximize the sum of node weights
The cardinality (size) of the result subgraph is not specified as input in our problem
We consider a different weighting scheme for nodes
  - Common positive value for query nodes
  - Common negative value for non-query nodes,
  - Rather than individual positive-value weights
Scenario 1: Twitter Social Network

- When one submits a text query to Twitter’s search engine
- Twitter returns a list of messages that match the query, along with the author of each message.
- For example, when one submits “Ukraine,” Twitter returns all recent messages that contain that keyword, together with the users who posted them.
- We wish to perform the following task:
- Among all Twitter users who posted a relevant message
- Find a subset of them that form a local cluster on Twitter’s social network
- Our goal is to discover a community of users who talk about that topic
  - Our input consists only of those users who have recently published a relevant message
Scenario 2: Non-Materialized Similarity Graph

- Consider an online library system that allows its users to perform text search on top of a bibliographic database.
- Upon receiving a query, the system returns a list of authors that match the query.
- For example, when one submits the keyword “discrepancy,”
- The system returns a list of authors who have published about the topic.
We wish to perform the following task:

Among all authors in this particular result set, identify a subset of similar authors.

Nodes represent authors, and edges indicate pairs of authors whose similarity exceeds a user-defined threshold.

As the similarity threshold is user-defined the graph cannot be pre-computed.

An all-pairs similarity join to materialize the graph at query time is impractical.

We can solve the problem without materializing the entire similarity graph.
Contribution

- To devise algorithms that, given as input a set of query nodes
- Find the maximum-discrepancy connected subgraph
- Using as few calls to the get-neighbors function as possible
- Consider the discrepancy-maximization problem under an unrestricted-access model
- The algorithm can access nodes and edges of the graph in an arbitrary manner
The solution for the local-access model based on a two-phase approach

- In the first phase, we make calls to the get-neighbors function to retrieve a subgraph
- In many cases, the optimal subgraph can be found without retrieving the entire graph
- In the second phase, we can use any algorithm for the unrestricted-access model on the retrieved subgraph
Main idea

- Prove that the problem of linear-discrepancy maximization on graphs is NP-hard in the general case
- Prove that the problem has a polynomial-time solution when the graph is a tree
- To define fast heuristic algorithms to produce solutions
Paper Information

Authors:

Charu C. Aggarwal
IBM T. J. Watson Research Center
Yorktown Heights, New York, USA
Email: charu@us.ibm.com

Philip S. Yu
University of Illinois at Chicago
Illinois, Chicago, USA
Email: psyu@cs.uic.edu

Publication:
- ICDE 2015

Type:
- Research Paper
Motivation

- A single system may collect data from different sensors
- Large number of sensors collecting data streams over long periods of time generate huge amount of data
- A user may want to query data to identify
  - Behaviour of one stream
  - Behaviour of multiple streams during certain time frame
- Main Problem addressed by paper: how to store these streams in the form of “summary representations” to enable efficient access and diagnosis
Related Work

Stream Summarization

- Various techniques proposed for summarization of streams
- Representation of streams to allow for reconstruction to certain degree of accuracy
  - Compression of continuous values
  - Aggregation of frequency counts of discrete values
Proposed Method

1. Observe Sensor Network Hierarchy
2. Quantize Sensor Signal
3. Identify representative Sketch
4. Query the Sketches
Proposed Method Cntd.

- Representative SKETCH
  - For a given time horizon of length $h$, determine the average signal value for a given sensor node.
  - For a given time horizon of length $h$, determine the number of abnormally large signal occurrences in a given set of sensors.
  - For a given historical time instant, determine the exact signal value.
  - For a given time horizon of length $h$, determine the distance in the signal between two sensor nodes.
Temporal Spatial Keyword Top-K Publish/Subscribe

Presented by: Siddhant Kulkarni

Spring 2016
Authors:

Lisi Chen #1, Gao Cong #2, Xin Cao †3, Kian-Lee Tan *4

# Nanyang Technological University,
{1chen012@e., 2gaocong}@ntu.edu.sg

† Queen’s University Belfast,
3x.cao@qub.ac.uk

* National University of Singapore,
4tankl@comp.nus.edu.sg

Publication:

► ICDE 2015

Type:

► Research Paper
Lot of Geotagged Textual data is available

- Twitter
- Facebook
- Instagram

Find the posts that are posted near you recently and are the most relevant

- Top-k relevant and recent posts posted near you
Related Work

- Publish Subscribe Systems for Geotagged objects
- Problem:
  - Very few or too many objects found
  - Irrelevant objects may be found because of location
TaSK

- Temporal Spatial keyword Top-K subscription query

Method

- Conditional Influence Region
  - Region in which we are looking for the new incoming Geo tagged objects
  - NON Overlapping
- Every incoming object that falls in CIR is ranked
  - Check if the rank falls in Top-K
  - If it does, update results
- Cost based model for identifying cells in which to look for the geotagged text objects
Scalable Distributed Transactions across Heterogeneous Stores

Presented by: Dardan Xhymshiti
Spring 2016
Authors:

Akon Dey  Alan Dekete  Uwe Rohm
School of Information Technologies, University of Sydney NSW 2006 Australia.

Publication:
- ICDE 2015

Type:
- Research Paper
KEY WORDS

- **NoSQL**: databases that model the data in other means than tabular relations (key-value, columnar, graph, document)
- **Multi-item transaction**: a transaction that involves operations in one or more data stores immediately.
- **ACID**: (Atomicity, Consistency, Isolation, Durability)
Many applications are moved into cloud because it offers:

- Scalability and high performance.
- High availability and fault tolerance.
- Distribution of data across multiple nodes.

Limitations:

Cloud infrastructure mostly offers NoSQL database models which have some limitations:

- Limited query capabilities (restricted to access primary key, no joins etc.).
- Typically suited for single key access.
- Limited **transaction** capabilities
  - Single record transactions
  - No multi-item ACID transactions.
Related work

There exist some ways to support multi-item transactions

1. Every application manages transactional access to the various data stores
   - Complex, Prone to programmer error

2. Use middleware to coordinate transactional access to the data store
   - Suitable for situations where the applications are deployed in a controlled environments.
Main Idea

- Create a library called Cherry Garcia which places the transaction support in a client-side.
- This library deals with:
  - Transaction coordination
  - Datastore abstraction classes
```java
public void UserTransaction() {
    Datastore cds = Datastore.create("credentials.xml");
    Datastore gds = Datastore.create("goog_creds.xml");
    Datastore wds = Datastore.create("msft_creds.xml");
    Transaction tx = new Transaction(cds);
    try {
        tx.start();
        Record saving = tx.read(gds, "saving")
        Record checking = tx.read(wds, "checking");
        int s = saving.get("amount");
        int c = checking.get("amount");
        saving.set("amount", s - 5);
        checking.set("amount", c + 5);
        tx.write(gds, "saving", saving);
        tx.write(wds, "checking", checking);
        tx.commit();
    } catch (Exception e) {
        tx.abort();
    }
}
```
Contribution

- Define a client library that enables multi-item transactions across heterogeneous key-value stores.
- Description of this library implementation in Java.
- Evaluation of the performance and validity of the implementation using YCSB+T benchmark for evaluation web-scale transactional systems.