LDBC SNB Datagen: Under the hood
Arnau Prat
9th LBDC TUC meeting
9/10 February Walldorf (Germany)
Why a synthetic graph generator?

• Real graphs are sometimes difficult to obtain
  • Not practical to distribute TeraBytes of data
  • Privacy concerns

• Real data do not always have the desired characteristics
  • Many dimensions to be tested (size, distributions, structural characteristics, etc.) as they can affect the performance of the tested systems
  • Difficult to obtain real data for all the desired dimension combinations
Wish list of a synthetic data generator

- Scalable
  - From GigaBytes to TeraBytes of data
- Realistic
  - Distributions: attributes, degrees, etc.
  - Correlations: attributes, edges, etc.
  - Structural characteristics: clustering coefficient, largest connected component, diameter, etc.
- Flexible
  - Allow choosing the characteristics of the generated data
  - Support different output formats
LDBC SNB DATAGEN

- DATAGEN is a fork of S3G2[1]
- Started development during LDBC European Project as the data generator for the LDBC Social Network Benchmark Workload
- Available at: https://github.com/ldbc/ldbc_snb_datagen

LDBC SNB DATAGEN

- Generates a Social Network graph
  - Uses dictionaries extracted from Dbpedia to populate the dataset with realistic attributes
    - e.g. Person names, countries, companies, tags (interests)
  - Correlated attributes
    - e.g. Person names with countries, correlations between tags, etc.
  - Correlated Friendship subgraph
    - i.e. Edges between persons sharing interests and universities are more likely
  - Realistic distributions
    - Facebook-like degree distribution, attribute distributions etc.
  - Event-based user activity generation
    - Mimick spikes of activity around specific events
LDBC SNB DATAGEN

- Built on top of Hadoop
  - Able to generate Terabytes of data with a small commodity cluster
  - Billion edge graphs in few hours

- Deterministic
Data Generation Process

- Person Generation
- Knows Graph Generation
- Knows graph serialization
- Activity Generation
- Activity serialization

Execution
Person Generation

- A 4-machine cluster
- 100,000 Person network
- Block size $m = 10,000 \rightarrow 10$ blocks in total
Data Generation Process

Execution
Knows Graph Generation

- Edge Generation Substep (Main Interest)
- Edge Generation Substep (University-age-gender)
- Edge Generation Substep (Random)
- Edges Merge

One substep for each correlation dimension
Edge Generation Substep

- Sort by correlation dimension:
  - e.g. Main interest, University-age, random
- Rank Person keys as their position in the sorted array (between 0 and N-1)
Edge Generation Substep

- **Persons.file**
  - Key = person id
  - Value = Person

- **Parallel sort and rank**

- **Persons.file.sorted**
  - Key = Rank
  - Value = Person

- **Edge generation**
  - Block 0
  - Block 1
  - Block 2
  - Block 3
  - Block 4
  - Block 5
  - Block 6
  - Block 7
  - Block 8
  - Block 9

- **Block n**

- **Independent state**

- **Persons.file.sorted**
  - Key = Rank
  - Value = Person

- **Person.Edge.file.n**
  - Key = person id
  - Value = Person

**Key concepts**:
- The probability of creating an edge decreases geometrically with the distance.
- Persons with similar characteristics (close in the sorted array) are more likely to be connected, producing a correlated graph.
- The amount of edge a person can create depends on its assigned target degree.
- A weight is assigned to each edge, which can be overridden by the user.
Edge Generation Substep

Person.Edges.file.0 ➔ Merge edges ➔ Person.Edges.file.final

Person.Edges.file.1 ➔

Person.Edges.file.2 ➔

To eliminate duplicate edges between the same pair of Persons
Data Generation Process

Execution

Person Generation

Knows Graph Generation

Knows graph serialization

Activity Generation

Activity serialization
Activity Generation

Person.Edges.file.final

Parallel sort and rank

Persons.Edges.file.sorted

Activity generation

Block 0
Block 4
Block 8
Block 1
Block 5
Block 9
Block 2
Block 6
Block 3
Block 7

serialized files
Activity Generation

- Split into two phases: Spiky vs uniform activity generation
- For each Person
  Generate Wall
  - Generate members (Person friends)
  - Generate message cascade
  Generate Groups
  - Generate members
  - Generate message (Person friends and others in the block)
- Uniform:
  - Cascade initiator topic is correlated with author interests
  - Creation Date is selected uniformly from max(author creation date, parent creation date) until end of simulation
Spiky - Activity Generation

TIMELINE

VOLUME

Person creationDate

Proportionally to its volume

Flashmob Topic
Spiky - Activity Generation

Post/Comment creationDates are clustered around the flashmob tag following this shape.

Data Generation Process

- Person Generation
- Knows Graph Generation
- Knows graph serialization
- Activity Generation
- Activity serialization

Execution
Other features

- Control the size of the graph
  - person based
  - knows graph based
- Generate only the knows graph without all the activity
- Customize:
  - the Knows graph degree distribution
  - edge weights
  - serializers
  - the knows generation step
  - message text generation
  - data formatting
Conclusions, known issues and next steps

- Datagen allows you to generate a realistic Social Network based on a Map/Reduce approach
- It scales to terabytes of data and billion edge graphs
- Monolithic execution model
  - Things are generated even if they are not needed
  - Why do we need to generate all Person attributes if we only need 20% of them when generating the graph for Graphalytics?
  - Why do we need to populate “Knows” with person attributes if we are not going to generate activity?
- Leads to a bad use of resources and larger execution times
- LDBC Datagen 2:
  - New architecture.execution model,
  - In-Place data generation
  - Language driven data/properties definition
THANK YOU!
(and we are recruiting)