Synthetic Data Generation: Theory, Techniques, and Application Domains

Joseph E. Hoag
Dissertation Topic Defense
Industry needs an efficient, flexible means of generating synthetic data for:

- Regression testing
- Secure application development
- Testing of data mining applications
- Science and industry applications
- Secure data deployment (mix synthetic and real data together)
Impact

- Several companies have expressed interest in our prototype generator, including Wal-Mart, NCR, Acxiom, and Deloitte.
- We generated 10 years (5 TB, 70B rows) of store-item-daily sales data and loaded it into Oracle 10g for Wal-Mart, to prove that grid computing offers faster, cheaper DBMS.
Synthetic data generation has theoretical underpinnings in the areas of database constraints and data inference. Data generation constraints can be captured in a standardized description language. A synthetic data generator can be written to execute efficiently and run in parallel. Finally, synthetic data generation has applications in the traditional tabular domain and beyond.
Scope

• The field of synthetic data generation has a potentially infinite scope. Any program with output can be considered to be a data generator.
• For this dissertation, we will restrict the scope of generation to finite tabular data, maps, and strings accepted by a given formal language mechanism (finite state machine, PDA, CFG).
• No attempt will be made to generate all possible values for a specified problem.
Related Technologies

- Data Modelling
- Simulation
- Privacy/Security
- Data Mining
- Statistics
- Cryptology
Related Work

- Jon White's MS thesis
- UC-Riverside Generator
- Commercialized Generators:
  - Turbo Data (turbodata.ca)
  - GS Data Generator (GSApps.com)
  - DTM Data Generator (sqledit.com)
  - RowGen (iri.com)
Techniques:
SDG Input Language

• Synthetic Data Generation Language (SDGL)
  • XML-based
  • Codifies constraint types, language
  • Pools
  • Modular
Techniques:
SDG Architecture

- Front-End GUI
  - Simplifies data description process
  - Can import source table info into SDGL file
- Back-End Data Generator
  - Efficient single-parse logic for formula processing
  - Parallel execution capability
    - No communication => Linear speedup
  - Deterministic output regardless of platform or number of processors
Techniques:
SDG Pools

- Pools are flexible data dictionaries, which can model:
  - Simple domains ("FR", "SO", "JR", "SR", "GR")
  - Tables of reference data (State/City/Zip code)
  - Graphs
  - State machines
  - ???
- Elements can be weighted
- Pools can be modular
Techniques:
SDG Constraint Types

- Min/Max
- Statistical Distribution
- Formulas (incl. Pool references)
- Iterations (Query, Pool, or Numerical)
- Query Pools
- Iteration Variables
- Data types: int, real, string, date, time, timestamp, boolean
Theory:
Representational Power - Mapping common data models to SDG

- Can SDGL capture various data and relation concepts in the relational and E-R models? If so, how?
  - Domain Constraints
  - Foreign Key Constraints
  - Cardinality Constraints
  - Participation Constraints
  - Relationship Types (1:1, 1:N, M:N)
  - Composite Attributes
Theory:
Security, Privacy, and Inference

- Data inference relates to privacy and security.
- Identification of data transparency levels:
  - 5: Use original data
  - 0: Use completely random data, or identical data for every record
  - 1-4: ??
- Techniques for generating data at various transparency levels.
Theory: Random but Deterministic

- There are good and bad ways to implement this. It’s possible that the current method might end up “repeating” random patterns.
Application Domains:
Tabular

- Generating 10 years of synthetic POS daily summary data
- Generating “Enterprise” supply-chain RFID data
Application Domains:  
Non-Tabular

Apply SDG to the following:

• Generate a map (topology, roads, etc…)
• Generate legal strings for a given finite state machine or context-free grammar. (Can pools effectively model these?)