Big Data Benchmarking needs Big Metadata Generation

Boris Glavic¹, Patricia C. Arocena², Radu Cicanu³, Renée J. Miller²

IIT DBGroup¹ University of Toronto² University of Oxford³
1) Motivation
2) Requirements for Big Metadata Generation
3) The iBench Metadata Generator
4) Preparing iBench for Big Data
5) Conclusions
Motivation

• **Volume, Velocity, Variety**
  – Heterogeneous structure, formats, content

• **Pay-as-you-go philosophy**
  – Integration and data preparation when needed for analysis

• **Data integration and curation tasks**
  – To stress test a system’s capabilities in this regard using a benchmark with a small simple “schema” is not enough!
Motivation

• Many integration tasks use metadata as input
  – Schema matching
  – Record linkage
  – Mapping discovery
  – Data exchange
  – Virtual data integration
  – Data cleaning (constraint based)
Motivation

- Fixed schema benchmarks do not stress test systems implementing tasks
  - How does my schema matching algorithm scale when I vary the size of the schemas to be matched?
  - How is the precision/recall influenced by naming conventions in the schemas?
Outline

1) Motivation
2) Requirements for Big Metadata Generation
3) The iBench Metadata Generator
4) Preparing iBench for Big Data
5) Conclusions
Big Metadata Generation

Requirements

• Generate diverse metadata, that is nonetheless realistic
• Control over the characteristics of the generated metadata
  – Size
  – Structure
  – ...
• Integrate with a data generator to generate data that matches the generated meta-data
  – Data should obey integrity constraints
Outline

1) Motivation
2) Requirements for Big Metadata Generation
3) The iBench Metadata Generator
4) Preparing iBench for Big Data
5) Conclusions
The iBench Metadata Generator

• A flexible metadata generator
  – Initially developed for empirical evaluation of metadata-heavy integration tasks

• Generates
  – Schemas
  – Schema matches
  – Constraints (e.g., PK, FK, FDs, …)
  – Mappings
  – Transformations (SQL for now)

• Goal
  – Produce inputs as well as “gold standard” output for evaluation of integration systems
The iBench Metadata Generator

• History
  – We needed a metadata generator to evaluate our data integration systems
  – We considered STBenchmark
    • lacked of features that we needed
  – We started to pile on new functionality
  – We used the new system in evaluations
  – We realized that it this system may be useful as a general tool for metadata generation
The iBench Metadata Generator

• Targeted integration tasks
  – Data exchange
  – Virtual data integration
  – Schema matching
  – Mapping discovery
  – Constraint discovery?
  – Data cleaning (actually that’s another story/collaboration)
The iBench Metadata Generator

- **Example Task: Schema matching**
  - Find correspondences between elements of two schemas
  - **Input:** two or more schemas
  - **Output:** set of matches (attribute pairs)

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cust</td>
<td>Customer</td>
</tr>
<tr>
<td>Name</td>
<td>Name</td>
</tr>
<tr>
<td>Addr</td>
<td>Addr</td>
</tr>
<tr>
<td>Emp</td>
<td>Loyalty</td>
</tr>
<tr>
<td>Name</td>
<td>Person</td>
</tr>
<tr>
<td>Company</td>
<td></td>
</tr>
</tbody>
</table>

**Big Data Benchmarking needs Big Metadata Generation**
The iBench Metadata Generator

- What distinguishes data integration evaluation from using say a TPC benchmark to evaluate the performance of say Hive?
- Need diverse, realistic, large metadata
  - E.g., testing schema matching on a single fixed schema is not really meaningful
- Need control over the generated metadata
  - E.g., how does my schema matcher scale when I
  - … increase the size of the input schemas
  - … change the size of relations
  - … change how source and target relations are related
The iBench Metadata Generator

• The solution to a task may not be uniquely defined
  – What is the correct result of an entity resolution task?
  – System may produce false positives or false negatives
  – System may be approximately correct
  – Need “gold standard” to compare result to
  – Need “quality measures” to compare solutions
The iBench Metadata Generator

- How are integration systems typically evaluated?
- Small real-world integration scenarios
  - Advantages:
    - Realistic ;-)  
  - Disadvantages:
    - Not possible to scale (schema-size, data-size, …)  
    - Not possible to vary characteristics (e.g., attribute naming)  
- Ad-hoc synthetic data
  - Advantages:
    - Can influence scale and characteristics  
  - Disadvantages:
    - Often not very realistic metadata  
    - Diversity requires huge effort
• Approach
  – Assembles complex schemas/mappings from simple mapping “primitives”
    • E.g., vertically partitioning of a relation
  – User controls which primitives are used
  – Characteristics of metadata controlled through “scenario parameters”
    • E.g., number of attributes per relation
The iBench Metadata Generator

Big Data Benchmarking needs Big Metadata Generation
• Primitive
  – A “template” for schemas, mappings, constraints, ...

Source
  Cust
    Name
    Addr
  Emp
    Name
    Company

Target
  Customer
    Name
    Addr
  Person
    Id
    Name
  WorksAt
    EmpRec
    Firm
    Id
The iBench Metadata Generator

- Primitive
  - A “template” for schemas, mappings, constraints, ...

Source
- Cust
  - Name
  - Addr
- Emp
  - Name
  - Company

Target
- Customer
  - Name
  - Addr
  - Loyalty
- Person
  - Id
  - Name
  - WorksAt
  - EmpRec
  - Firm
  - Id

Vertical Partitioning

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

- **Primitive**
  - A “template” for schemas, mappings, constraints, ...

```plaintext
Source
Cust
  Name
  Addr
Emp
  Name
  Company

Target
Customer
  Name
  Addr
  Loyalty

Person
  Id
  Name
  WorksAt
    EmpRec
      Firm
      Id
```

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• A primitive defines
  – Source and target schemas
  – Mappings
  – Schema matches
  – Primary Key and Foreign Key constraints

• Moving parts (more about that later)
  – Number of attributes per relation
  – “Join size”: e.g., number of partitions in VP
  – Create PKs or not?

21

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

- Primitives Types (excerpt)
  - Vertical partitioning (IS-A, HAS-A, N-TO-M)
  - Horizontal partitioning
  - Copy (complete, add-attribute, delete-attribute)
  - Create/Drop Table
  - Merge (inversion of VP)
  - Surrogate key generation
  - ...

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• Scenario parameters
  – Control over characteristics that influence the whole integration scenario
    • E.g., create data or not?
    • E.g., reusing of relations across primitives
  – Influence primitives
    • E.g., number of attributes, “join size”
  – Control additional generation phases that were not modeled as primitives on purpose
    • E.g., creation of random functional dependencies

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• Input configuration
  – A configuration file (1-2 pages of text)
  – Determines number of primitives to create for each type
  – Sets scenario parameters given as mean and min/max values
  – Determines what types of metadata/data to generate

• Output
  – An XML file containing all the metadata
  – Data as CSV files (optional)

24
The iBench Metadata Generator

• iBench Generation Process
  – Read configuration file
  – Create primitives one at a time to incrementally generate integration scenario
    • Roll dice every time for scenario parameters
    • Input file can specify a global RNG seed
  – Additional processing to deal with orthogonal stuff
    • Generation of additional random functional dependencies
    • Inject incompleteness
  – Generate data
    • We use ToxGene for now
    • PK and FK constraints are taken into account

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• Primitive Generation Example
  – I want 1 copy and 1 vertical partitioning
• Primitive Generation Example
  – I want 1 copy and 1 vertical partitioning

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cust</td>
<td>Customer</td>
</tr>
<tr>
<td>Name</td>
<td>Name</td>
</tr>
<tr>
<td>Addr</td>
<td>Addr</td>
</tr>
<tr>
<td></td>
<td>Loyalty</td>
</tr>
</tbody>
</table>
• Primitive Generation Example
  – I want 1 copy and 1 vertical partitioning

Source
Cust
  Name
  Addr
Emp
  Name
  Company

Target
Customer
  Name
  Addr
  Loyalty
Person
  Id
  Name
  EmpRec
  Firm
  Id

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• Reuse across primitives
  – Primitives cover many patterns that occur in the real world
  – however in the real world these primitives do not occur in isolation

• Enable primitives to share parts of the schema
  – Scenario parameters: source reuse, target reuse
  – Probabilistically determine whether to reuse previously generated relations
The iBench Metadata Generator

• Example

Source

Cust
Name
Addr
Emp
Name
Company
Executive
Name
Position

Target

Customer
Name
Addr
Loyalty
Person
Id
Name
WorksAt
EmpRec
Firm
Id

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• **Use case: Value invention**
  – Translate mappings written
    • from expressive, less well-behaved language (SO tgds)
    • into less expressive, more well-behaved language (st-tgds)
  – **Input**: schemas, integrity constraints, mappings
  – **Output**: translated mappings (if possible)
  – **Evaluation Goal**: how often do we succeed
  – **Why iBench**: need a large number of diverse mappings to get meaningful results
  – **Evaluation Approach**: generated 12.5 million integration scenarios based on randomly generated configuration file

Big Data Benchmarking needs Big Metadata Generation
• Use case: Vagabond
  – Finding explanations for data exchange errors
    • User marks attribute values in generated data as incorrect
    • System enumerates and ranks potential causes
  – Input: schemas, integrity constraints, mappings, schema matches, data, errors
  – Output: enumeration of causes or incremental ranking
  – Evaluation Goal: evaluate scalability, quality
  – Why iBench:
    • Control characteristics for scalability evaluation
    • Scale real-world examples
• **Use case: Mapping Discovery**
  – Learning mappings between schemas using statistical techniques
  – **Input**: schemas, data, constraints
  – **Output**: mappings

  – **University of California, Santa-Cruz**
  • Lise Getoor
  • Alex Memory
  • [https://linqs.soe.ucsc.edu/people](https://linqs.soe.ucsc.edu/people)
The iBench Metadata Generator

• Related Work
  – STBenchmark
    • Pioneered the primitive approach
    • No support for controlling reuse among primitives
    • No support for
  – Datagenerators
    • Our approach currently is quite naïve using ToxGene
    • We currently do not provide too much control over the data generation

34

Big Data Benchmarking needs Big Metadata Generation
The iBench Metadata Generator

• Performance

![Graph showing generation time vs. number of attributes]

- (0, 0, 0)
- (25, 0, 0)
- (0, 25, 0)
- (0, 0, 25)

Big Data Benchmarking needs Big Metadata Generation
Outline

1) Motivation
2) Requirements for Big Metadata Generation
3) The iBench Metadata Generator
4) Preparing iBench for Big Data
5) Conclusions
Preparing iBench for Big Data

- Support semi-structured data
  - E.g., JSON
- Support unstructured data?
- Integration with scalable data generators
  - E.g., PDGF or Myriad
- Distribute metadata generation
  - Relatively straight-forward
  - Really necessary?
Outline

1) Motivation
2) Requirements for Big Metadata Generation
3) The iBench Metadata Generator
4) Preparing iBench for Big Data
5) Conclusions
Conclusions

• Overview of iBench
  – Comprehensive metadata generator
  – Produces inputs and outputs (gold standards) for a variety of integration tasks
  – Control over characteristics of the generated metadata

• Why metadata generation for Big Data Benchmarking

• Some hints of what is missing
Future Work

• Immediate plans
  – Integration with scalable data generator
  – Give more control over data generation
  – Orchestration of mappings
    • Serial and parallel
  – Improve renaming options to better support schema matching

• Longer term ideas
  – Generate workloads that match data
  – Incorporate implementations of quality measures
Big Data Benchmarking needs Big Metadata Generation