SCHEMA INFERENCE FOR MASSIVE JSON DATASETS

ParisBD 2017

Mohamed-Amine Baazizi[1], Houssem Ben Lahmar[2], Dario Colazzo[3], Giorgio Ghelli[4], Carlo Sartiani[5]

(1) Université Pierre et Marie Curie, France (2) University of Stuttgart, Germany (3) Université Paris-Dauphine, France
(4) Università di Pisa, Italy (5) Università della Basilicata, Italy

JSON IN A NUTSHELL

- Acronym for **J**ava**S**cript **O**bject **N**otation
- Very popular format for data exchange (API services)
- Predominant data model for NoSQL systems (AsterixDB, Mongo, Arango, Couchbase, Elastic, etc)
- A current candidate schema language (IETF JSON-Schema), several query languages (AQL, SQL++, N1QL, etc)

JSON AND SCHEMAS

No a priori prescriptive schema

Flexible data management

Problem: lack the opportunity to:

1) understand the structure of potentially large data

2) reason about the structural properties of data

3) apply schema-based optimizations

Goal: Inferring a posteriori descriptive schemas for JSON

RELATED WORK

Semistructured Data:
- approximate/optimal schemas [Nestorov et al. 97, Nestorov et al. 98]
- data guides [Goldman et al. 97]
- expressive type language [Buneman et al. 99]

XML and RDF:
- concise DTDs [Garofalakis et al. 00, Bex et al. 06]
- summary of XML large collections [Hegewald et al. 06]
- summary of ontology properties [Cebiric et al. 15]

JSON:
- schema inference in MR (sketch) [Colazzo et al.12]
- summarization [Wang et al. 15, Klettle et al. 15]
- extraction of normalized schema [DiScala et al. 16], adaptiving schema [Spoth et al. 2017]

different data model, no account for structural variations, no scalability

SCHEMA INFERENCE FEATURES

1. Captures complex data and its structural variability
2. Produces succinct schemas
3. Processes large dataset

AGENDA

- Context and Motivation
- JSON data model and schema language
- Schema inference mechanism
- Experimental study
- Conclusion

JSON DATA MODEL

- Null, True, False, Numbers, Strings
- Records: { $l_1 : v_1, \dots , l_n : v_n$}
  each l_i is unique in a record
- Arrays: [$v_1, \dots , v_n$]

```
{
"person":
  {
   "firstname": "John",
   "lastname": "Smith",
   "coordinates": [120 , 10 ]
  }
}
```

A JSON value

A SCHEMA LANGUAGE FOR JSON

- Basic Types: ***Null, Bool, Num, Str***
- Record Types: $\{l_1 : T_1q, \dots, l_n : T_nq\}$ $q \in \{!, ?\}$
- Union types: ***T+U***
- Array Types: **[*T**]**

The JSON-Schema proposal, formalized by Pezoa et al. 2016, does not consider union nor compact arrays

```
{
"person":
  {
   "firstname": Null + Str ,
   ("lastname": Str ) ?,
   "coordinates": [Num * ]
  }
}
```

A JSON schema

SCHEMA INFERENCE MECHANISM

[...
123, “abc”
…]

J_1

[...
879,
Null…]

J_2

[...
{“lab”:
758 }]

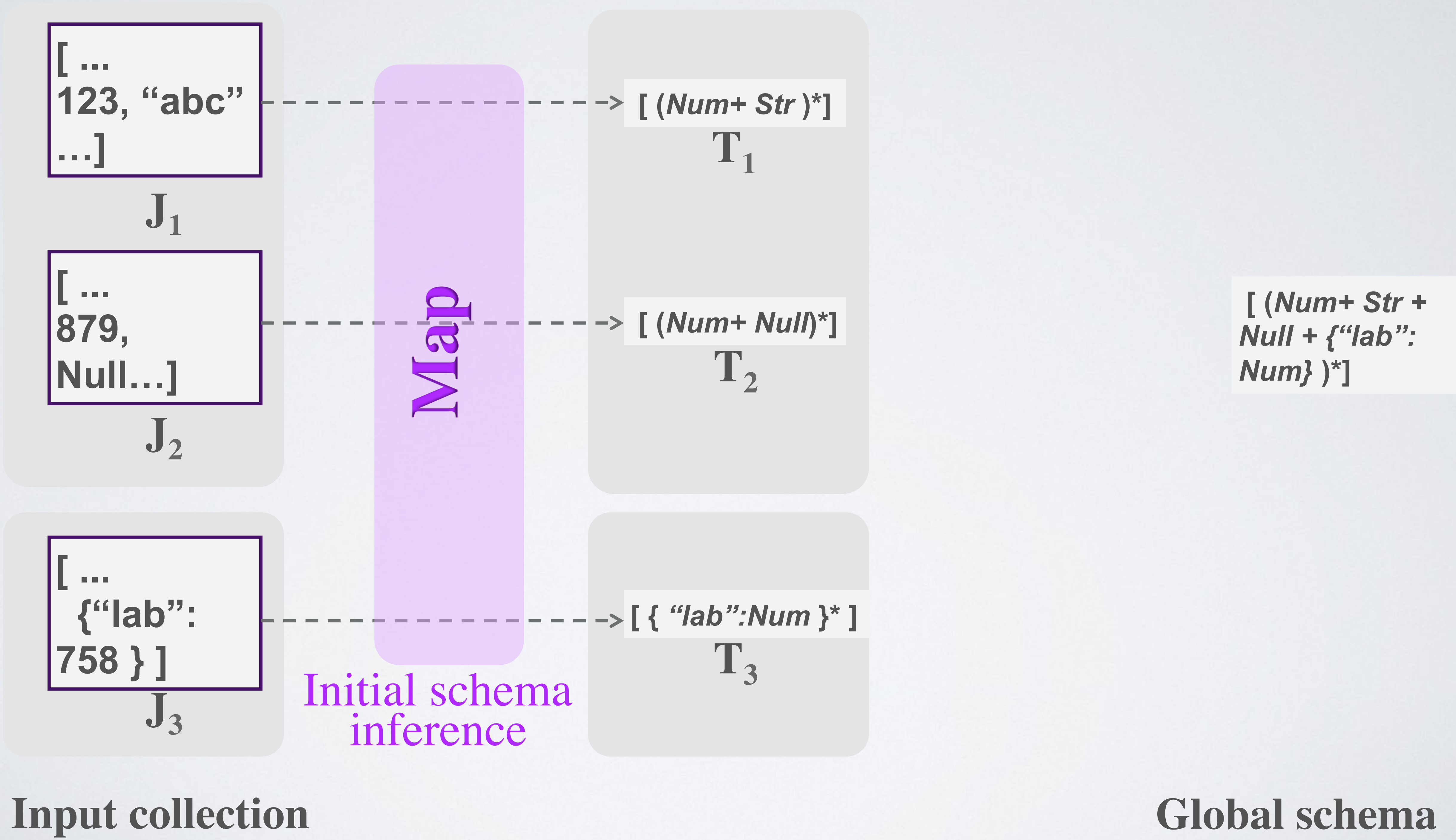
J_3

[(*Num*+ *Str* +
Null + *{“lab”:
Num}*)*]

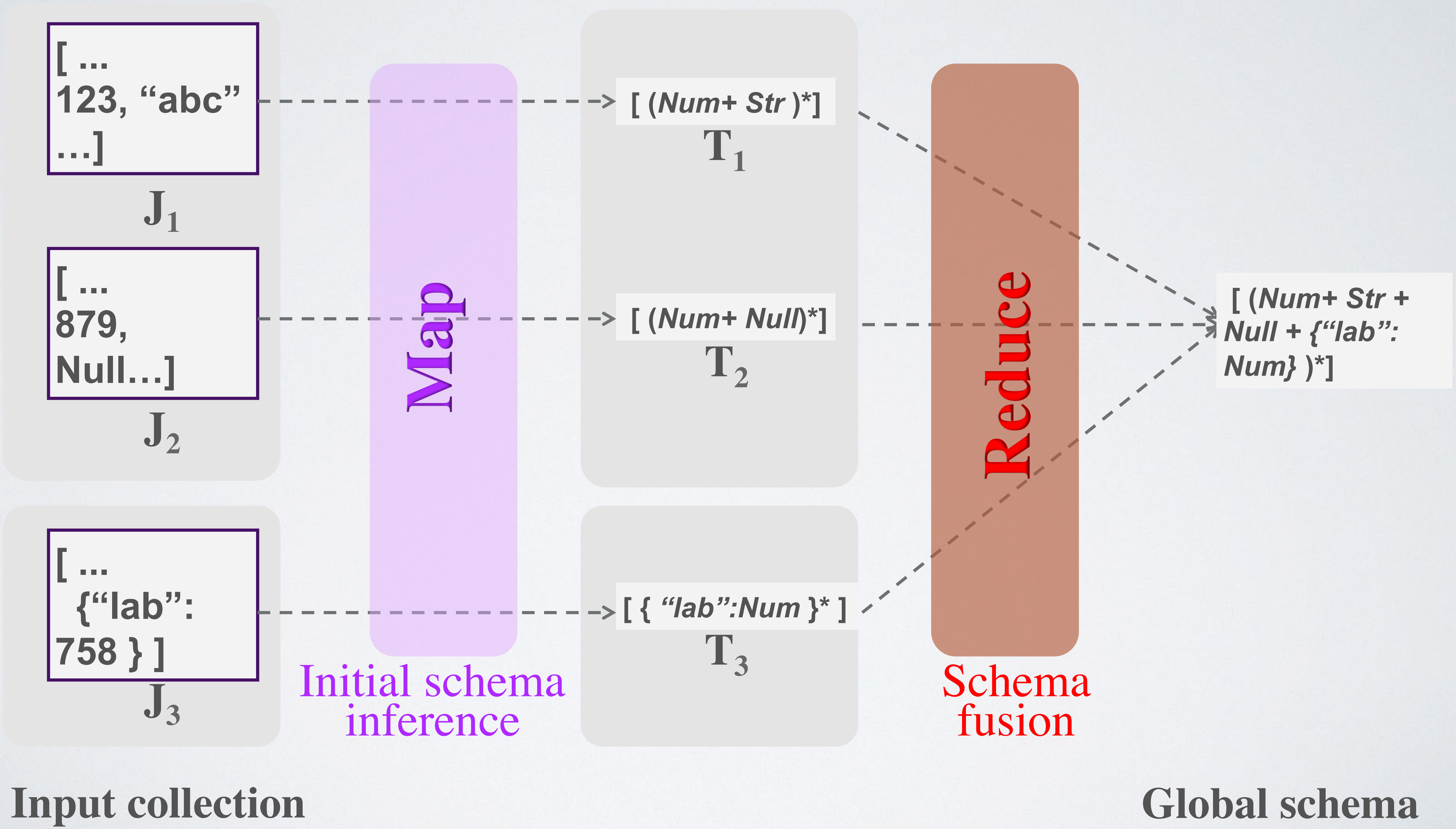
Input collection

Global schema

SCHEMA INFERENCE MECHANISM



SCHEMA INFERENCE MECHANISM



SCHEMA INFERENCE MECHANISM

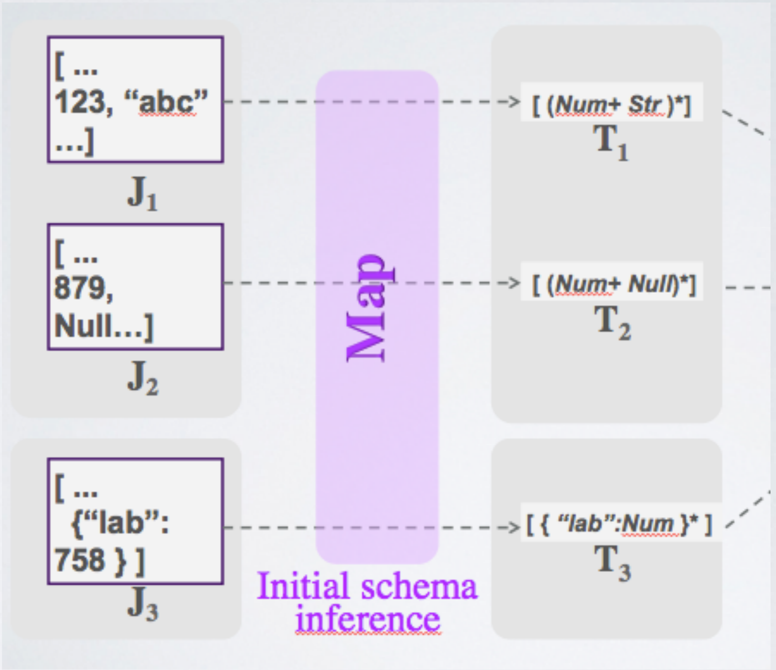
Initial schema inference

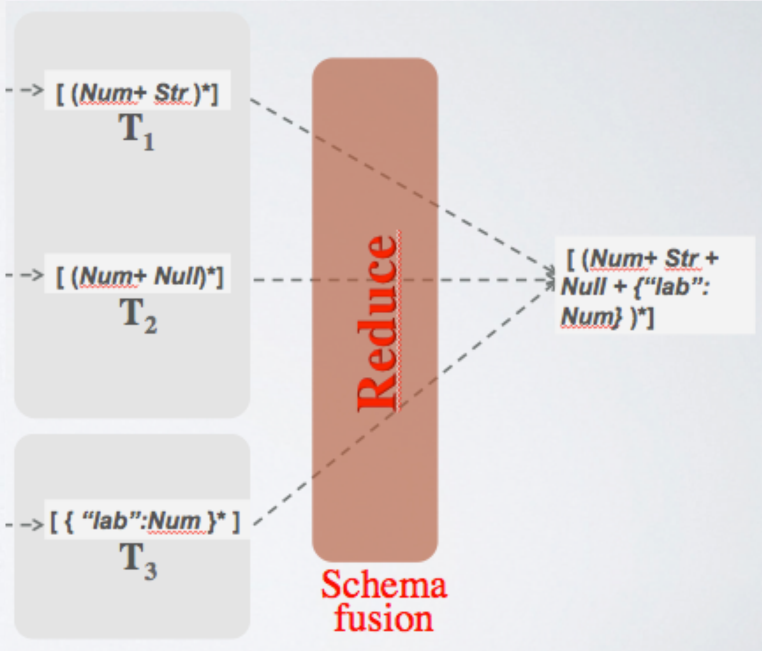
- generalizes values
- compacts array content

Schema fusion: Merge(T,U)

- collapses identical types
- detects optional fields
- captures irregularities

Sound, commutative and associative





FUSION ILLUSTRATED

```
{
"person":
  {
   "firstname": Null ,
   "lastname": Str ,
   "coordinates": [Num * ]
  }
}
```

T

```
{
"person":
  {
   "firstname": Str,

    "coordinates": [Bool * ]
  }
}
```

U

fusion

```
{
"person":
  {
   "firstname": Null + Str ,
   ("lastname": Str ) ?,
   "coordinates": [(Num + Bool) * ]
  }
}
```

Merge(T,U)

EXPERIMENTAL STUDY

- **Main goal**: assess succinctness and efficiency
- Scala-based implementation
  - initial schema inference: extending Json4s [json4s] parser
  - schema fusion: follow the formal specification
- Settings: 6 nodes, 10 dual core, 64GB RAM, Spark 1.6.1
- Datasets: Github, Twitter, and NYTimes stored on HDFS

EXPERIMENTAL RESULTS

Dataset	Github	Twitter	NYTimes
Input Data			
Size	13 GB	21 GB	21.3 GB
# objects	1 million	9.9 million	1.2 million
avg. AST size	495	142	1,238
avg. AST height	4	3	7
Initial Schema inference			
avg. AST size	495	135	109
Schema fusion			
AST size	655	559	139
Execution time			
	0.7 min	1.7 min	2.8 min

CONCLUSION

Inference of a ***descriptive*** schema for a JSON dataset

- mitigate the lack of schema, incomplete data description

A simple, yet informative schema language

- capture the global structure of data and variations

A distributed and incremental inference mechanism

- process large datasets, tackle dynamicity

FUTURE DIRECTIONS

Succinctness vs. precision

- recover information loss (e.g. field correlation)

Schemas enriched with statistics

- cardinality of fields / union branches, typical array size

Impact on storage and query optimization

Analysis of other use cases, visualization of schemas

THANK YOU

REFERENCES (1/2)

- G. J. Bex, F. Neven, T. Schwentick, K. Tuyls. *Inference of Concise DTDs from XML Data*, VLDB'06
- P. Buneman, B. Pierce. *Union Types for Semistructured Data*, DBPL'99
- S. Cebiric, F. Goasdoué, I. Manolescu. Query-Oriented Summarization of RDF Graphs. PVLDB'15
- D. Colazzo, G, Ghelli. C. Sartiani. *Typing Massive JSON Datasets*, XLDI'12
- M. DiScala, D. Abadi. *Automatic Generation of Normalized Relational Schemas from Nested Key-Value Data*, SIGMOD'16
- M. Garofalakis, A. Gionis, R. Rastogi, S. Seshadri, K. Shim, *XTRACT: A System for Extracting Document Type Descriptors from XML Documents,* SIGMOD'00
- R. Goldman, J. Widom. DataGuides. *Enabling Query Formulation and Optimization in Semistructured Databases,* VLDB'97

REFERENCES (2/2)

- J. Hegewald, F. Naumann, M. Weis. *XStruct: Efficient Schema Extraction from Multiple and Large XML Documents*, ICDE'06
- M. Klettle. U. Störl, S. Scherzinger. *Schema Extraction and Structural Outlier Detection for JSON-based NoSQL DataStores*, BTW'15
- S. Nestorov, S. Abiteboul, R. Motwani. Infererring *Structure in Semistructured Data,* SIGMOD'97
- S. Nestorov, S. Abiteboul, R. Motwani. *Extracting Schema from Semistructured Data,* SIGMOD'98
- F. Pezoa, J. Reutter, F. Suarez, M. Ugarte, D. Vrgoc. *Foundations of JSON Schema,* WWW'16
- W. Spoth, B. Sadat Arab, E.S. Chan, D. Gawlick, … Adaptive Schema Databases. CIDR'17
- L.Wang, O. Hassanzadeh, S. Zhang, J. Shi, L. Jiao, J. Zou, C. Wang. *Schema Management for Document Stores*, VLDB'15