Dremel: Interactive Analysis of Web-Scale Datasets

By Frank Chan

CS297

Outline

- Background
- Data Model
- Data conversion
- Query execution

Background

- Many times users need to be able to query a database to access particular records
- Generally, views can be created for multiple joins, but views are slow
- Alternatively, a replicated table of that join can be created to store a “results” table
- Problem 1: “results” tables are not up to date
- Problem 2: DBMS requires to load the data first, not good with distributed datastores

Data Model

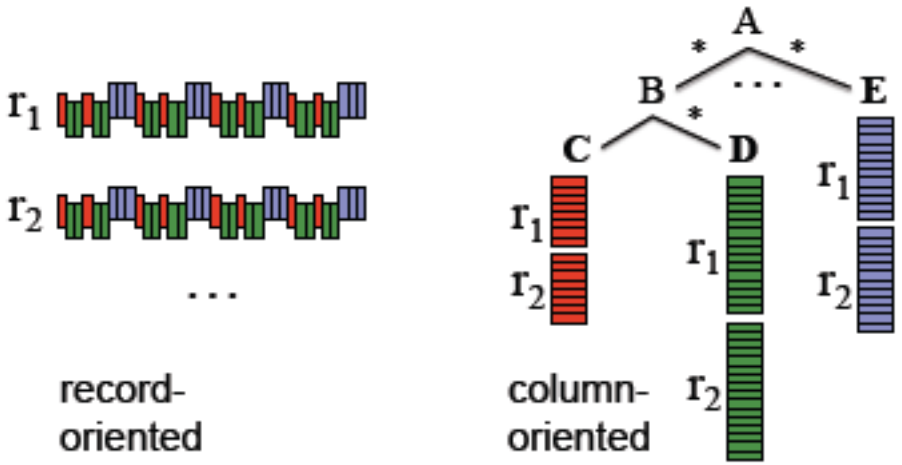

Figure 1: Record-wise vs. columnar representation of nested data

- Conversion of record stores into column oriented model
- Advantage: A, B, C can be stored contiguously so that A, B, C can be retrieved without needing to access D, E

Data Conversion

- Repetition level
  - At what repeated field in the field's path the value has repeated
- Definition level
  - How many fields could be undefined (because it's optional or repeated) are actually present
- Encoding
  - Levels generally are not large, so bits can be used to encode

Data Conversion Example

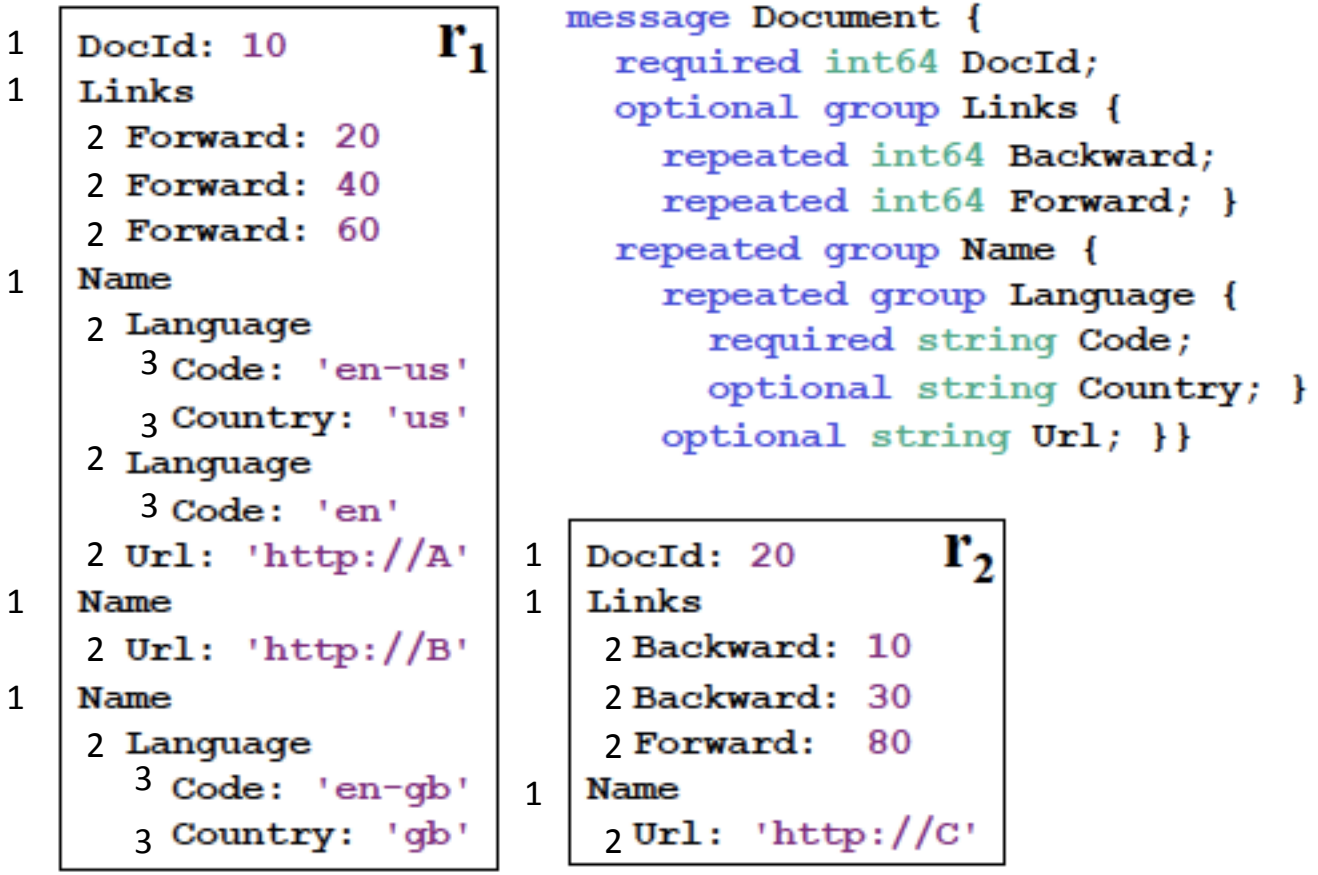

Figure 2: Two sample nested records and their schema

Data Conversion Example (Cont'd)

DocId

value	r	d
10	0	0
20	0	0

Name.Url

value	r	d
http://A	0	2
http://B	1	2
NULL	1	1
http://C	0	2

Links.Forward

value	r	d
20	0	2
40	1	2
60	1	2
80	0	2

Links.Backward

value	r	d
NULL	0	1
10	0	2
30	1	2

Name.Language.Code

value	r	d
en-us	0	2
en	2	2
NULL	1	1
en-gb	1	2
NULL	0	1

Name.Language.Country

value	r	d
us	0	3
NULL	2	2
NULL	1	1
gb	1	3
NULL	0	1

Figure 3: Column-striped representation of the sample data in Figure 2, showing repetition levels (r) and definition levels (d)

Data record reassembled

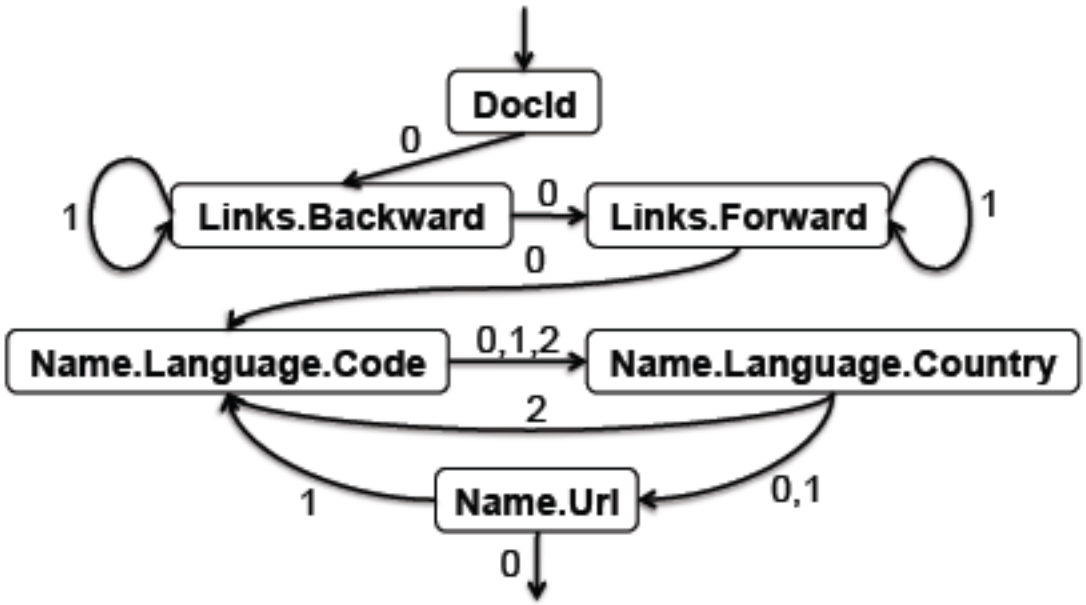

Figure 4: Complete record assembly automaton. Edges are labeled with repetition levels.

- Reassembling the record can be done in a graph
- Refer back to slide 6 to see the data definition for precedence

Query record reassembled

```
SELECT DocId AS Id,
  COUNT(Name.Language.Code) WITHIN Name AS Cnt,
  Name.Url + ',' + Name.Language.Code AS Str
FROM t
WHERE REGEXP(Name.Url, '^http') AND DocId < 20;
```

```
Id: 10                                    t1
Name
  Cnt: 2
  Language
    Str: 'http://A,en-us'
    Str: 'http://A,en'
Name
  Cnt: 0
```

```
message QueryResult {
  required int64 Id;
  repeated group Name {
    optional uint64 Cnt;
    repeated group Language {
      optional string Str; }}}
```

Figure 6: Sample query, its result, and output schema

Query Execution

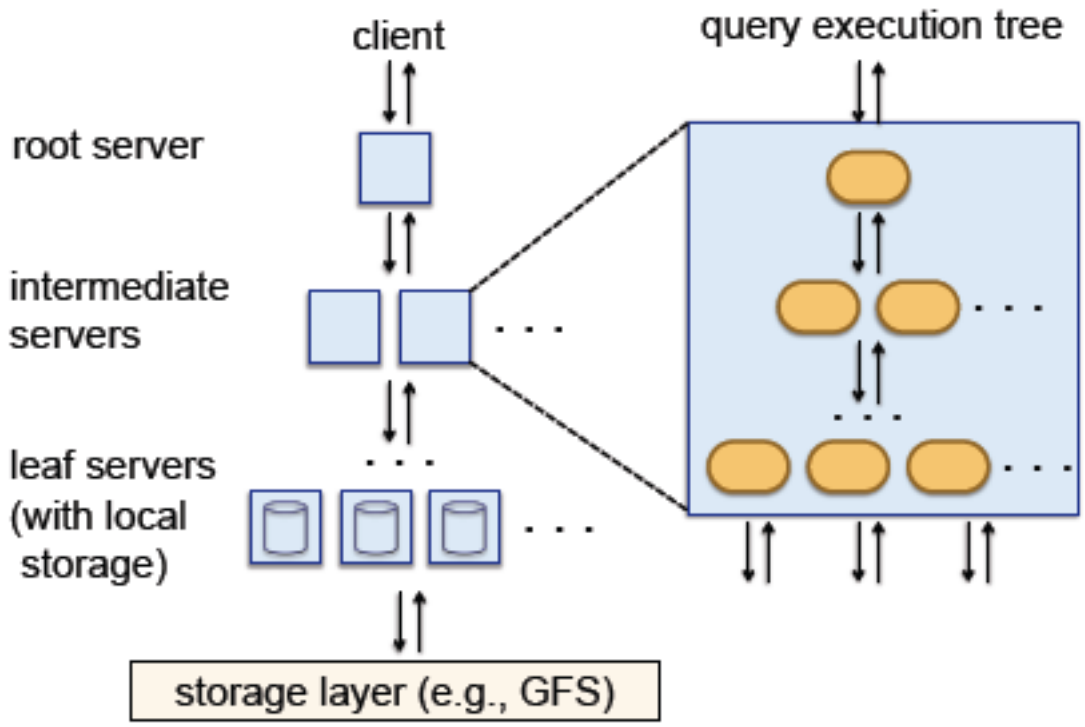

Figure 7: System architecture and execution inside a server node

- Query execution is done in a three step process in a tree architechture

Query Execution Steps

- Root server
  - Retrieves the incoming query
  - Reads metadata from tables
  - Routes queries to the intermediate servers (by doing a rewrite based on the metadata)
- Leaf servers
  - Accesses the local data results retrieved from the predicate
- Intermediate servers
  - Rewrites the query to separate the results to the leaf servers
  - Uses a UNION ALL aggregation to finalize the total results

Query Sample

SELECT A, COUNT(B) FROM T GROUP BY A

- Query is received by the root node

SELECT A, SUM(c) FROM (R_1^1 UNION ALL ... R_n^1) GROUP BY A

- Query is rewritten so that it can be dispersed to the intermediate server

Tables $R_1^1, \ldots, R_n^1$ are the results of queries sent to the nodes $1, \ldots, n$ at level 1 of the serving tree:

R_i^1 = SELECT A, COUNT(B) AS c FROM T_i^1 GROUP BY A

- Queries are given to the leaf nodes based on data locality of T_i

Performance experiment: Columnar vs. Record disk access

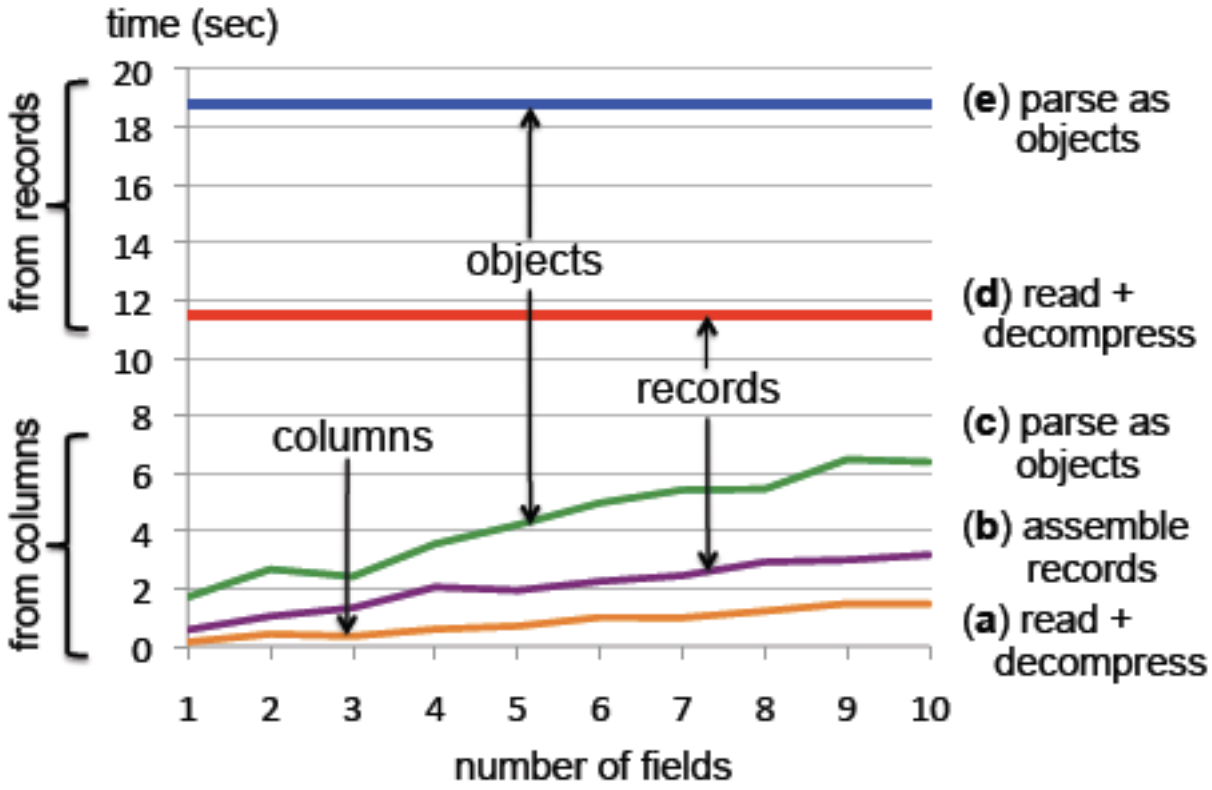

Figure 9: Performance breakdown when reading from a local disk (300K-record fragment of Table T_1)

- As number of fields increases, columnar format increases
- For Record format, operation is static regardless of how many fields need to be operated on

Experiement: MR vs. Dremel

- Both systems have 3000 workers
- Uses the same query:
  - In SQL: SELECT SUM(CountWords(txtField))/ COUNT(*) FROM T1;
  - In MR:

```
numRecs: table sum of int;
numWords: table sum of int;
emit numRecs <- 1;
emit numWords <- CountWords(input.txtField);
```

Experiement: MR vs. Dremel Results

- Switching MR from records to columns, it gained a full order of magnitude (hours to mins)
- Another order of magnitude is gained going from MR-columns to Dremel (mins to secs)

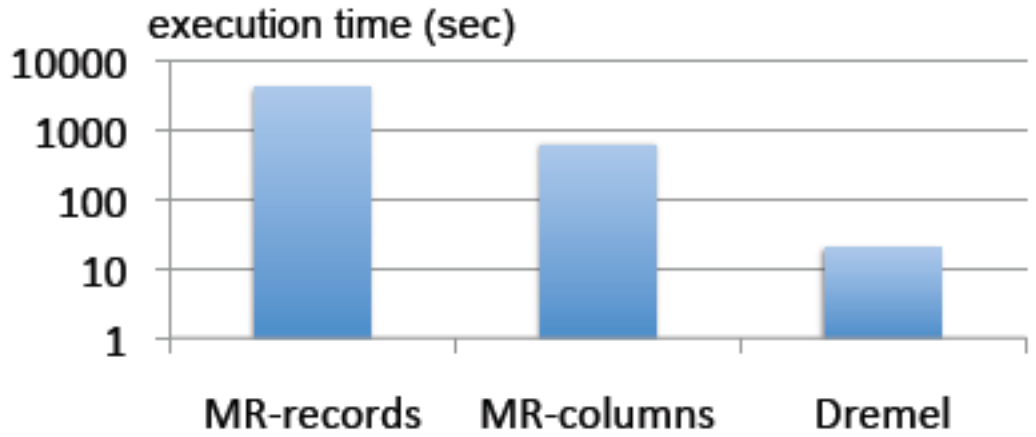

Figure 10: MR and Dremel execution on columnar vs. record-oriented storage (3000 nodes, 85 billion records)