GPText: Greenplum Parallel Statistical Text Analysis Framework

Kun Li, Christian Grant, Daisy Zhe Wang, Sunny Khatri, George Chitouras

cgrant@cise.ufl.edu
In-Database Analytics

Transfer time is way too slow! We need another way to go!

Tools don't scale!

We Want In-Database Analytics! We want it now!
In-Database Analytics

• NLP support in RDBMs is limited
In-Database Analytics

- NLP support in RDBMs is limited
- ML Algorithms in RDBMs is non-trivial
In-Database Analytics

• NLP support in RDBMs is **limited**
• ML Algorithms in RDBMs is **non-trivial**
• Text search in RDBMs is **slow**
GPText

A framework for large-scale statistical text analytics over a parallel DBMS.

- The DB provides parallelism and scale.
- Integrated text analytics algorithms with MADlib.
- Specialized architecture for text indexing and search using Solr.
Outline

• Introduction
• GreenplumDB
• GreenplumDB ∪ MADlib
  • In-DB Conditional Random Field package
• GreenplumDB ∪ MADlib ∪ Solr
• Demo Screenshots
• Conclusion
Greenplum DB

- A shared nothing parallel dbms.
- Parallel PostgreSQL instances.
- Queries are distributed over segments with a parallel query optimizer.
Greenplum ∪ MADLib

- An open source library for in-database analytics
- A collaborative effort between Berkeley, Wisconsin, and UF
- Maintained by Greenplum
MADlib Architecture

User Interface

“Driver” Functions
(outer loops of iterative algorithms, optimizer invocations)

High-level Abstraction Layer
(iteration controller, convex optimizers, ...)

RDBMS Built-in Functions

Inner Loops
(implemented as streaming algorithms)

Low-level Abstraction Layer
(matrix operations, C++ to RDBMS type bridge, ...)

RDBMS Query Processing
(Greenplum, PostgreSQL, ...)

SQL, generated from specification

Python with templated SQL

Python

C++
MADlib

MADlib Functions

Data Modeling
- Supervised Learning
  - Naive Bayes Classification
  - Linear Regression
  - Logistic Regression
  - Multinomial Logistic Regression
  - Elastic Net
  - Generic Cross-Validation
  - Decision Tree & Random Forests
  - Support Vector Machines
  - Cox Proportional Hazards Regression
  - Conditional Random Field
- Unsupervised Learning
  - Association Rules
  - k-Means Clustering
  - SVD Matrix Factorization
  - Low ranked Matrix Factorization
  - Parallel Latent Dirichlet Allocation

Descriptive Statistics
- Sketch-based Estimators
  - CountMin (Cormode-Muthukrishnan)
  - FM (Flajolet-Martin)
  - MFV (Most Frequent Values)

Support Modules
- Array Operations
- Conjugate Gradient
- Sparse Vectors
- Random Sampling
- Probability Functions
MADlib Functions

Supervised Learning
- Naïve Bayes Classification
- Linear Regression
- Logistic Regression
- Multinomial Logistic Regression
- Elastic Net
- Generic Cross-Validation
- Decision Tree & Random Forests
- Support Vector Machines
- Cox Proportional Hazards Regression
- Conditional Random Field

Unsupervised Learning
- Association Rules
- k-Means Clustering
- SVD Matrix Factorization
- Low ranked Matrix Factorization
- Parallel Latent Dirichlet Allocation

Descriptive Statistics
- Sketch-based Estimators
  - CountMin (Cormode-Muthukrishnan)
  - FM (Flajolet-Martin)
  - MFV (Most Frequent Values)

Support Modules
- Array Operations
- Conjugate Gradient
- Sparse Vectors
- Random Sampling
- Probability Functions
Conditional Random Fields

- The *linear-chain* CRF is used to find the most likely sequence of token labels.
Conditional Random Fields

- The *linear-chain* CRF is used to find the most likely sequence of token labels.

**CRF Model:**

```
X=tokens  2181  Shattuck  North  Berkeley  CA  USA
  x0  x1  x2  x3  x4  x5

Y=labels  y0  y1  y2  y3  y4  y5
```
CRF Inference

CRF Model:

X=tokens

Y=labels

pos street num street name city state country

0 5 1 0 1 1

1 2 15 7 8 7

2 12 24 21 18 17

3 21 32 32 30 26

4 29 40 38 42 35

5 39 47 46 46 50
CRF Scalability

- Single Host
- 64 MBs, 32 cores
- CoNLL 2000 dataset
Greenplum $\cup$ MADLib $\cup$ Solr
Greenplum ∪ MADLib ∪ Solr
GPText queries

```sql
select * from
gptext.create_index(<schema-name>,<table-name>,
<id_col>,<def-search-column>);

select * from
gptext.index(table(select * from <schema-name>.<table>),<index-name>);

select * from gptext.commit_index(<index-name>);

create table sigmod_terms as
select * from gptext.terms(table(select 1 scatter by 1),
<index-name>, <search-column>, 'sigmod*', 'rows=10');
```
Concept Application
Concept Application
Concept Application
Concept Application
Concept Application
Concept Application

Class: Positive
Location: Arden Hills
So pumped NBA bball is back. #TimberPuppies #unitedwerun
Conclusion

• We discussed our CRF contribution to MADLib.

• GPText is a scalable framework for text analytics in the database.

• We show a concept application supporting fast text search
Thank you

http://dsr.cise.ufl.edu
Extra Slides
CRF MADlib Interface

- select crf_train/test_data()
- select crf_train/test_fgen()
- select lincrf/vcrf_label()
CRF Training Algorithm

- Features extracted with queries.
- The database takes care of the parallelism.
- Each inner loop updates the state until convergence.
CRF Training Algorithm

- We can create a temporary table to store results
- Use a python udf or a with statement to control iterations

CREATE TEMPORARY TABLE iterative_algorithm AS
SELECT 0 AS iteration, NULL AS state

current_iteration = 0
CRF Training Algorithm

- Iterate the algorithm performing the **crf_lbfgs** step
- Increment and check to see if it is complete
CRF Training Algorithm

- If it is converged, finalize the result features.

```sql
SELECT internal_crf_lbfgs_result(state)
FROM iterative_algorithm
WHERE iteration = current_iteration
```