Final Project

SciDB for Clustering Analysis of MovieLens Ratings Data

Liu, Wusuo
Lee, Chia Ying

CSCI E-63 Big Data Analytics
Harvard University Extension School
Prof. Zoran B. Djordjević

Introduction: MovieLens Data

- MovieLens.org maintains a database of movie ratings

- Full dataset:
  1. 27,000 rated movies
  2. 138,000 users
  3. 20 million ratings
- Small dataset:
  1. 9,000 rated movies
  2. 700 users
  3. 100,000 ratings

userId	movieID	rating
1	31	2.5
2	10	4.0
2	17	5.0
2	39	5.0
2	47	4.0
2	50	4.0

- Big Data ML Goal:

  Cluster users according to similar movies they liked!

- Challenge: linear algebra on huge matrices

SciDB for Massive Scale Math and Stats

SciDB is designed for fast linear algebra on large arrays.
Efficient architecture for data storage and distributed computing suitable for mathematical and statistical operations

- Demo of SciDB's capabilities:

  1) *Sparse matrix multiply*
  2) *Singular Value Decomposition*

- Demo 2 clustering methods using SciDB:

  1)Cluster users based on correlations between users' ratings.
  2)Cluster users by Principle Component Analysis.

- Use small dataset due to limited resources.

Installation Notes

- Follow the instructions on:
https://paradigm4.atlassian.net/wiki/display/ESD/SciDB+Community+Edition+Installation+Guide

- The VM that has scidb installed provided by paradigm4
https://drive.google.com/drive/folders/0B7yt0n33Us0rT1FJdmxFV2g0OHc
has several problems:
- The pre-installed *scidb* R package is severely outdated. Several functions documented in the package are not available.
- There exists a problem with the system's MPI library that would prevent us to perform many important linear algebra operations like *gesvd*().

- Install *scidb* 15.12 on Ubuntu 14.04 from scratch.

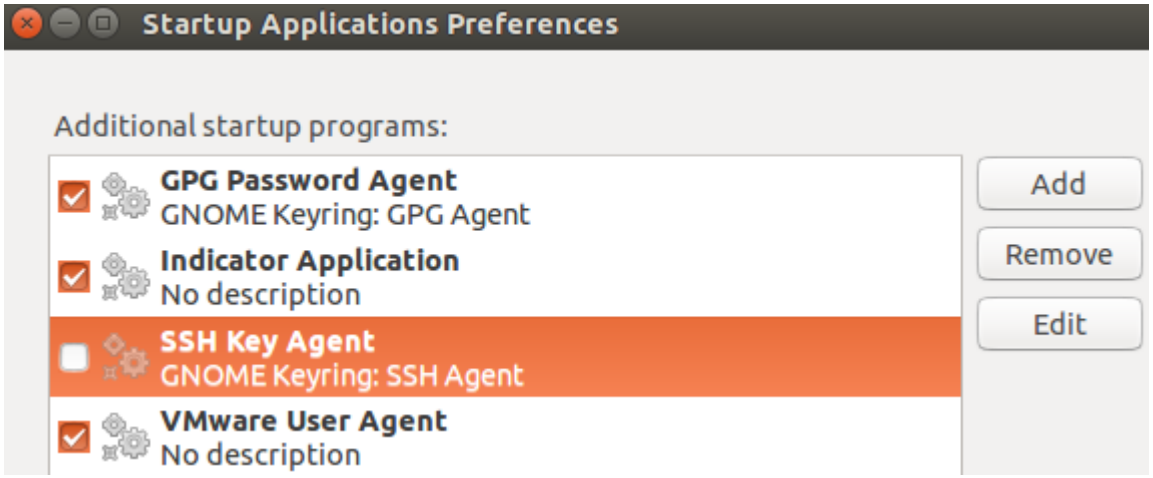
Installation Notes

- Ubuntu 14.04 download and set-up:
  - http://releases.ubuntu.com/14.04/
  - scidb installation requires passwords for both the root and scidb users on all the hosts. So the root login must be enabled on Ubuntu. This link explains the setup: https://askubuntu.com/questions/44418/how-to-enable-root-login

- SSH public key
  - Following section "Providing Passwordless SSH" in the installation document
    - Copy the generated public key file *id_rsa.pub* in the */home/.ssh* directory to the *./ssh* directories of all the users living on the cluster of one or more hosts, and rename it as "*authorized_keys*"
    - In our case, 1 host 2 uses, "scidb" and "root", so copy and rename *id_rsa.pub* to /home/.ssh and /root/.ssh.
    - The /.ssh directories are hidden by default in Ubuntu 14.04.

Installation Notes

- "Agent admitted failure to sign using the key"
  - After deploying scidb access to the users, to confirm the connection, the above error could happen.
  - Cause: Ubuntu desktop system uses gnome-keyring, which doesn't always handle specific formats of SSH keys correctly.
  - To confirm the cause, add *SSH_AUTH_SOCK=0* in front of ssh connecting command, e.g.
    - *SSH_AUTH_SOCK=0 ssh scidb@127.0.0.1*
  - Solution: uncheck SSH key Agent in "Startup Applications Preferences".



Installation Notes

- Build from source, *./run.py make*
  - Assign as much memory as possible to the VM. Several compiling steps need considerable amount of memory. If the memory is not enough, it runs into fatal errors.
  - Assign appropriate number of processors to the VM in order to compile the source code in parallel fashion, e.g. *./run.py make -j4* (4 threads)

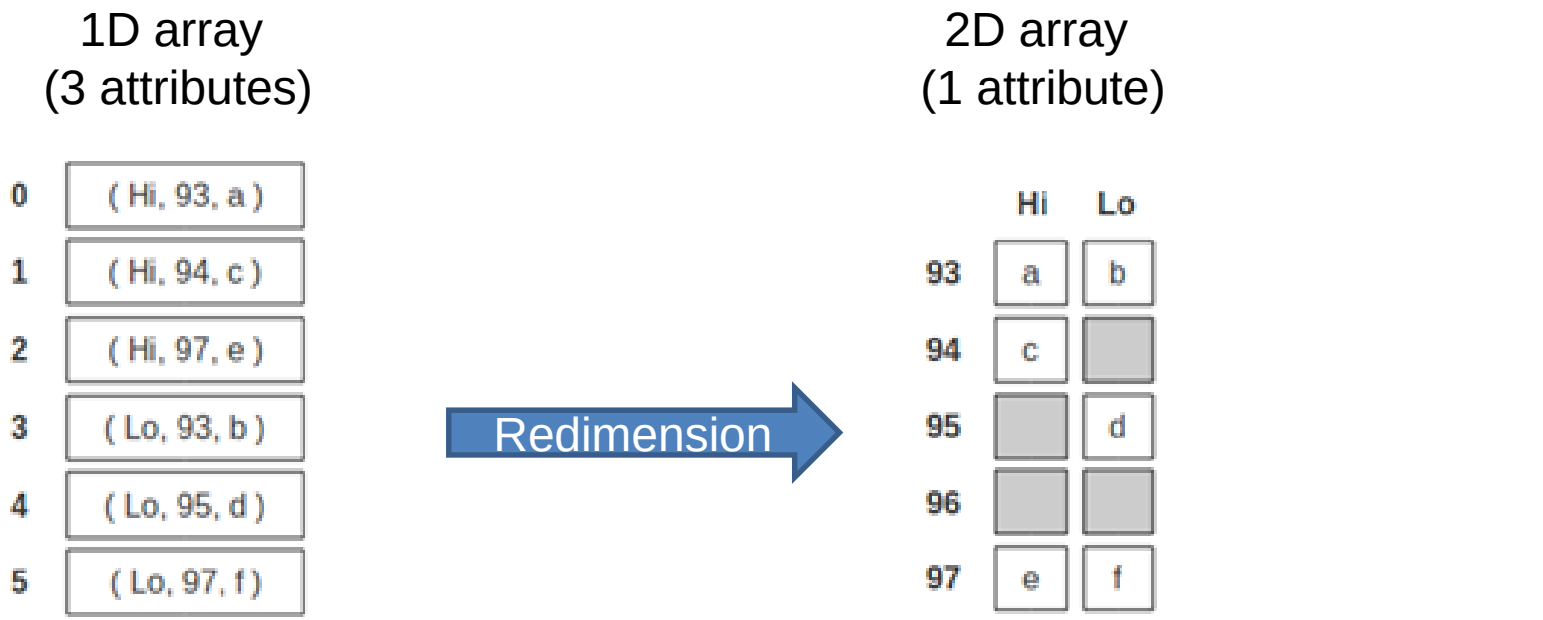
- Install R, Rstudio, R package scidb
  - Do not install the CRAN package scidb 2.0.0, which currently is not stable.
  - Install the package from github by *devtools::install_github("Paradigm4/SciDBR")*
  - To install devtools, first in terminal do
    - *sudo apt-get install libcurl4-openssl-dev libssl-dev*
  - In R terminal, execute *install.packages("devtools")*

Installation Notes

- Install *shim*
  - *shim* is a simple SciDB client that exposes limited SciDB functionality through a simple HTTP API.
  - Will enable us to connect to *localhost:8080* (if the host IP is set as 127.0.0.1) and submit *iquery.*
  - Will enable us to connect *scidb* in R.
  - Instruction site: https://github.com/Paradigm4/shim
  - Head to section “LD_LIBRARY_PATH issues” if a library linking error happens when trying to run *shim*

SciDB Arrays

- SciDB arrays are the basic data structure
- Arrays have one or more dimensions, and one or more attributes
  - Dimensions index the array cell
  - Attributes store the data



Poliakov, Brown. *Using SciDB and SciDB-R.* Paradigm4 2013

Dimensions play a similar role to primary keys in other databases

SciDB Schema

- The SciDB schema defines the array structure and data types

```
> scidb(dbConnect, "ratings_matrix")
SciDB expression  ratings_matrix
SciDB schema  <rating:double NOT NULL> [userId=0:671,1000,0,movieID=0:163949,1000,0]
  variable dimension   type nullable start    end chunk
1   userId      TRUE  int64    FALSE     0    671  1000
2  movieID      TRUE  int64    FALSE     0 163949  1000
3   rating     FALSE double    FALSE
```

- Arrays are stored efficiently in ***sparse*** format
- Arrays are stored in chunks may overlap to facilitate math ops.

```
> ratings_matrix <- iquery(dbConnect, "filter(ratings_matrix, TRUE)", return = T)
> head(ratings_matrix)
  userId movieID rating
1      1      31    2.5
2      2      10    4.0
3      2      17    5.0
4      2      39    5.0
5      2      47    4.0
6      2      50    4.0
```

SciDB Query Language and R package

- AQL is the basic SciDB query language
- AFL is the query language for functions in SciDB

```
scidb@ubuntu:~$ iquery
AQL% set lang AFL;
AFL%
```

- The scidb package provides an interface to R

```
> library(scidb)
> dbConnect <- scidbconnect()
> ratings <- iquery(dbConnect, "ratings", return = T)
```

Loading Data from CSV File into SciDB

- **Step 1a**: Create a 1D array `ratings`

```
AFL% CREATE ARRAY ratings <userId:int64, movieID:int64,
rating:double NOT NULL, timestamp:int64>
[i=0:?,1000000,0];
```

- **Step 1b**: Load from CSV into array

```
AFL% load(ratings, 'ratings_noHeader.csv', -2, 'csv');
```

- **Step 2**: Redimension from 1D to 2D array `ratings_matrix`

```
AFL% store(redimension(ratings, <rating:double NOT
NULL>[userId=0:671,?,0,movieID=0:163949,?,0]),
ratings_matrix);
```

Computing Statistics in SciDB

- What is the average rating of each user?

```
AFL% aggregate(ratings_matrix, avg(rating), userId)
```

userId	rating_avg
1	2.550000
2	3.486842
3	3.568627
4	4.348039
5	3.910000
6	3.261364

- What is the maximum rating a movie has received?

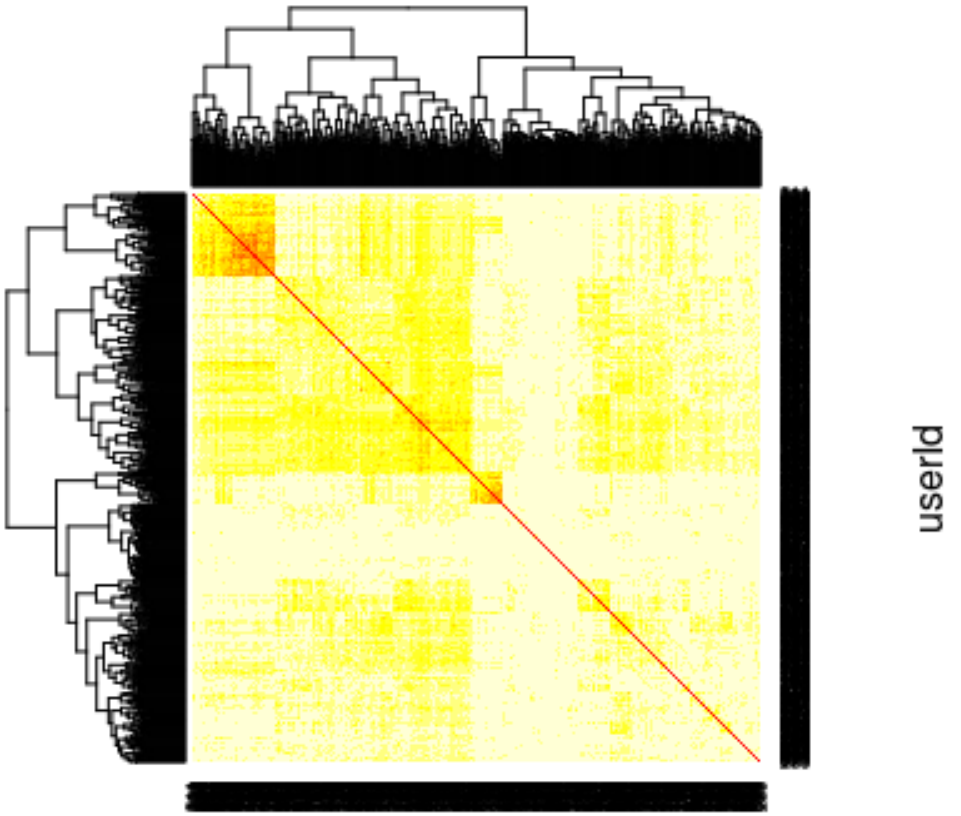
```
AFL% aggregate(ratings_matrix, max(rating), movieID)
```

movieID	rating_max
1	5.0
2	5.0
3	5.0
4	3.5
5	5.0
6	5.0

Clustering Analysis #1: Based on Correlation

- Correlation matrix: $M * M^T$

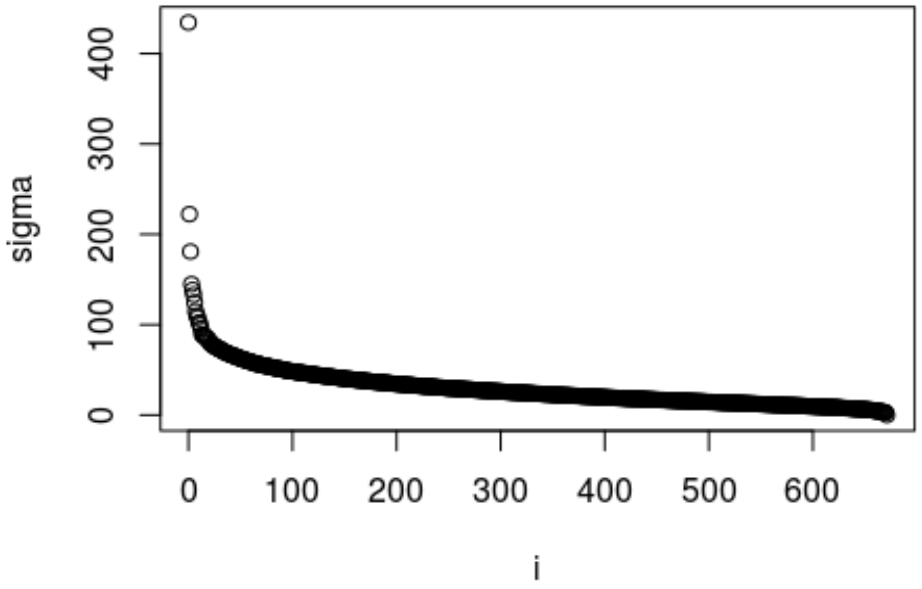
```
AFL% load_library('linear_algebra')
AFL% spgemm(ratings_matrix, transpose(ratings_matrix))
```

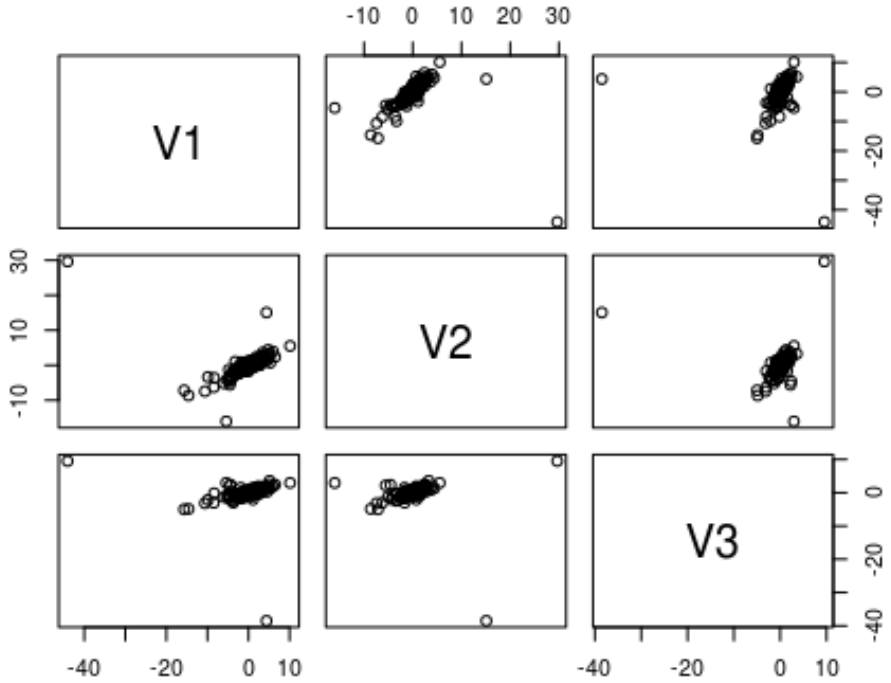


Clustering Analysis #2: based on PCA and SVD

- Singular Value Decomposition: M = U * S * V^T

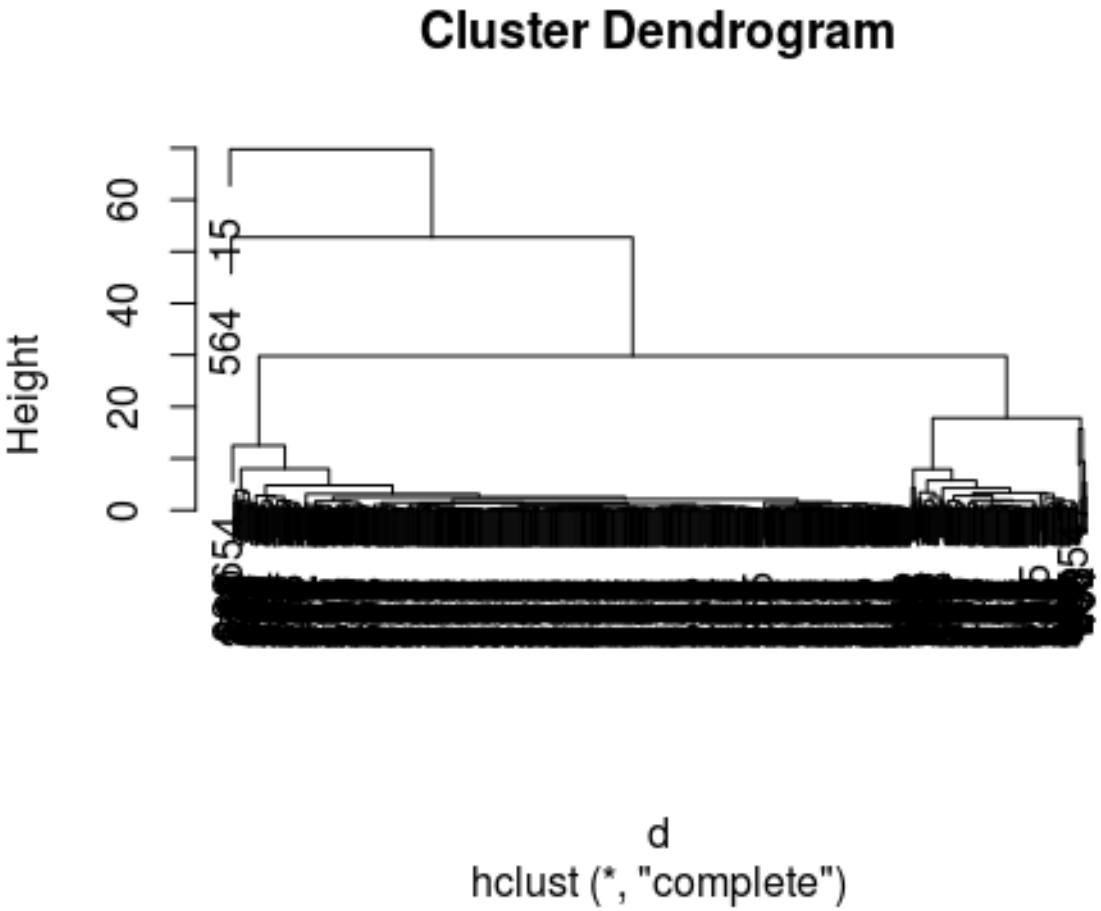
```
AFL% load_library('dense_linear_algebra')
AFL% gesvd(ratings_matrix_centered, 'S')
AFL% gesvd(ratings_matrix_centered, 'U')
```





Clustering Analysis #2: based on PCA and SVD

- Hierarchical clustering using top 3 singular vectors.



Conclusion

- SciDB is an efficient database designed for mathematical and statistical operations on large data
- SciDB has functionality to perform sparse matrix multiplication and SVD on massive matrices.
- SciDB can be used in conjunction with R language to perform Machine Learning tasks.
- We have shown the potential of using SciDB in clustering users based on their movie ratings.
- SciDB Enterprise Edition has enhanced functionality
  - E.g. Truncated SVD, glm, etc.

YouTube URLs, Last Page

- Two minute (short): https://youtu.be/1o5jLqXAXZs
- 15 minutes (long): https://youtu.be/qlwy4fOXLvk