15-826: Multimedia Databases and Data Mining

Lecture#1: Introduction

Christos Faloutsos
CMU
www.cs.cmu.edu/~christos

Outline

Goal: ‘Find similar / interesting things’

• Intro to DB
• Indexing - similarity search
• Data Mining
Problem

Given a large collection of (multimedia) records, or graphs, find similar/interesting things, ie:

• Allow fast, approximate queries, and
• Find rules/patterns

Q1: Applications, for ‘similar’?
Sample queries

• Similarity search
  – Find pairs of branches with similar sales patterns
  – ???

![Stock prices chart]

• Similarity search
  – Find pairs of branches with similar sales patterns
  – find medical cases similar to Smith's
  – Find pairs of sensor series that move in sync
  – Find shapes like a spark-plug
  – (nn: ‘case based reasoning’)
Problem

Given a large collection of (multimedia) records, or graphs, find similar/interesting things, ie:
- Allow fast, approximate queries, and
- Find rules/patterns

Q1: Examples, for ‘interesting’?
Sample queries –cont’d

• Rule discovery
  – Clusters (of branches; of sensor data; ...)
  – ???

• Rule discovery
  – Clusters (of branches; of sensor data; ...)
  – Forecasting (total sales for next year?)
  – Outliers (eg., unexpected part failures; fraud detection)
Example:

YahooWeb:
(a) In-degree vs. Out-degree  (b) Degree vs. Triangles  (c) Degree vs. PageRank

~1B nodes (web sites)
~6B edges (http links)
‘YahooWeb graph’

U Kang, Jay-Yoon Lee, Danai Koutra, and Christos Faloutsos. 
Net-Ray: Visualizing and Mining Billion-Scale Graphs
PAKDD 2014, Tainan, Taiwan.

Important Observation:

Find similar/interesting things: are related:
- Similar things ->
  - clusters/patterns
  - outliers
- Similar past waves -> forecasting
Outline

Goal: ‘Find similar / interesting things’

- (crash) intro to DB
- Indexing - similarity search
- Data Mining

Detailed Outline

Intro to DB

- Relational DBMS - what and why?
Intro to DB

- Relational DBMS - what and why?
  - inserting, retrieving and summarizing data
  - (views; security/privacy)
  - (concurrency control and recovery)
How do DBs work?

We use sqlite3 as an example, from http://www.sqlite.org

```
linux% sqlite3 mydb  # mydb: file
sqlite> create table student ( 
    ssn fixed; 
    name char(20) );
```

<table>
<thead>
<tr>
<th>student</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
How do DBs work?

sqlite> insert into student
values (123, “Smith”);

sqlite> select * from student;

<table>
<thead>
<tr>
<th>ssn</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Smith</td>
</tr>
</tbody>
</table>

How do DBs work?

sqlite> create table takes ( 
    ssn fixed,
    c_id char(5),
    grade fixed));

<table>
<thead>
<tr>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
</tr>
<tr>
<td>-------</td>
</tr>
</tbody>
</table>
How do DBs work - cont’d

More than one tables - joins

<table>
<thead>
<tr>
<th>student</th>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
<td>ssn</td>
</tr>
<tr>
<td>name</td>
<td>c_id</td>
</tr>
<tr>
<td></td>
<td>grade</td>
</tr>
</tbody>
</table>

sqlite> select name
from student, takes
where student.ssn = takes.ssn
and takes.c_id = “15826”

Q: What does this do?
How do DBs work - cont’d

`sqlite>` select name
    from student, takes
    where student.ssn = takes.ssn
    and takes.c_id = “15826”

Q: What does this do?
A: class roster

<table>
<thead>
<tr>
<th>student</th>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
<td>ssn</td>
</tr>
<tr>
<td>name</td>
<td>c_id</td>
</tr>
<tr>
<td></td>
<td>grade</td>
</tr>
</tbody>
</table>

SQL-DML

General form:
```sql
select a1, a2, … an
from r1, r2, … rm
where P
[order by ….]
[group by …]
[having …]
```
## Aggregation

Find ssn and GPA for each student

<table>
<thead>
<tr>
<th>student</th>
<th>takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssn</td>
<td>ssn</td>
</tr>
<tr>
<td>name</td>
<td>c_id</td>
</tr>
<tr>
<td></td>
<td>grade</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>123</th>
<th>603</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>412</td>
</tr>
<tr>
<td>234</td>
<td>603</td>
</tr>
</tbody>
</table>

How many lines of python/C++/Java code?
Aggregation

\[ \text{sqlite} > \text{select ssn, avg(grade) from takes group by ssn;} \]

<table>
<thead>
<tr>
<th>ssn</th>
<th>c_id</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>603</td>
<td>4</td>
</tr>
<tr>
<td>123</td>
<td>412</td>
<td>3</td>
</tr>
<tr>
<td>234</td>
<td>603</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ssn</th>
<th>avg(grade)</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>3.5</td>
</tr>
<tr>
<td>234</td>
<td>3</td>
</tr>
</tbody>
</table>

Detailed Outline

Intro to DB
- Relational DBMS - what and why?
  - inserting, retrieving and summarizing data
  - views; security/privacy
  - (concurrency control and recovery)
- What if slow?
- Conclusions
What if slow?

```
sqlite> select * from irs_table where ssn= ‘123’ ;
```

Q: What to do, if it takes 2hours?

A: build an index

Q: on what attribute?

Q’’: what syntax?
What if slow?

sqlite> select * from irs_table where
       ssn= ‘123’;

Q: What to do, if it takes 2 hours?
A: build an index
   Q’ : on what attribute? A: ssn
   Q’’ : what syntax? A: create index

What if slow - #2?

sqlite> create table friends (p1, p2);

Q: Facebook-style: find the 2-step-away people
What if slow - #2?

sqlite> create table friends (p1, p2);
sqlite> select f1.p1, f2.p2
       from friends f1, friends f2
       where f1.p2 = f2.p1;

Q: too slow – now what?

A: ‘explain’:
sqlite> explain select

```sql
f1.p1  f1.p2
f2.p1  f2.p2
```
Long answer:

- Check the query optimizer (see, say, Ramakrishnan + Gehrke 3rd edition, chapter 15):

Conclusions

- (relational) DBMSs: electronic record keepers
- customize them with create table commands
- ask SQL queries to retrieve info
Conclusions cont’d

Data mining **practitioner’s guide**:

- **group by** + aggregates
- If a query runs slow:
  - `explain select` – to see what happens
  - `create index` – often speeds up queries

For more info:

- Sqlite3: [www.sqlite.org](http://www.sqlite.org) - @ linux.andrew
- Ramakrishnan + Gehrke, 3rd edition
- 15-415/615 web page, eg,
  - [http://www.cs.cmu.edu/~christos/courses/dbms.F16](http://www.cs.cmu.edu/~christos/courses/dbms.F16)
We assume known:

- B-tree indices
- Hashing
- (also, [Ramakrishnan+Gehrke, ch. 10, ch.11])