Carnegie Mellon

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:

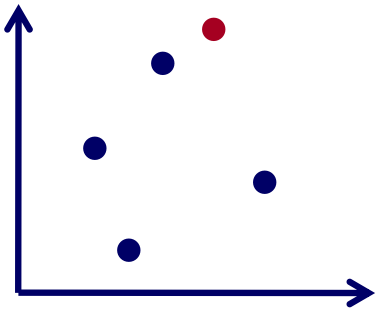
- Allow fast, approximate queries, and
- Find rules/patterns

Problem

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

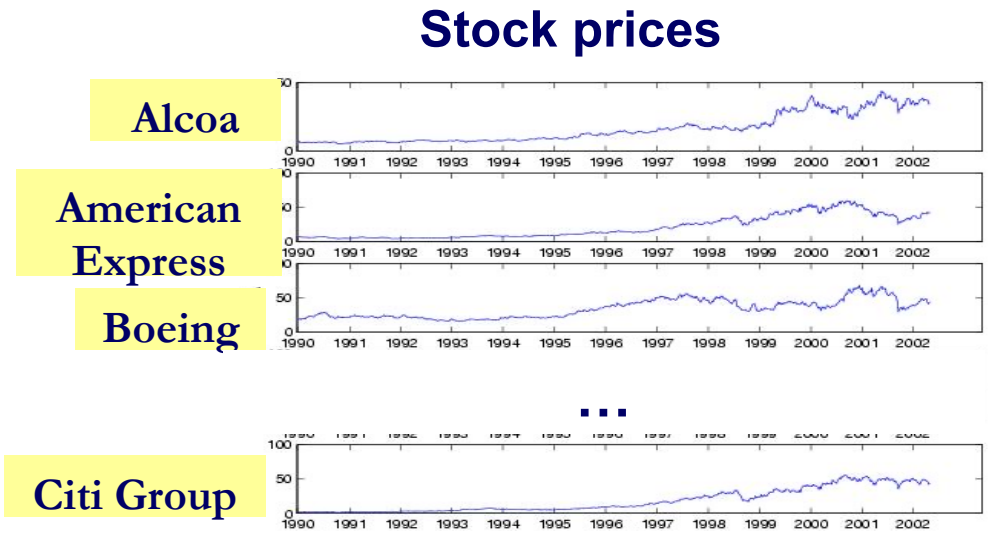
- Allow fast, approximate queries, and
- Find rules/patterns

Q1: Applications, for 'similar'?



Sample queries

- Similarity search
  - Find pairs of branches with similar sales patterns
  - ???



Sample queries

- 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

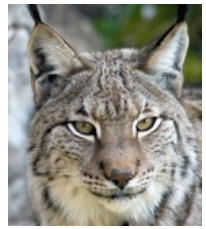
Given a large collection of (multimedia) records, or graphs, find similar/**interesting** things, ie:

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

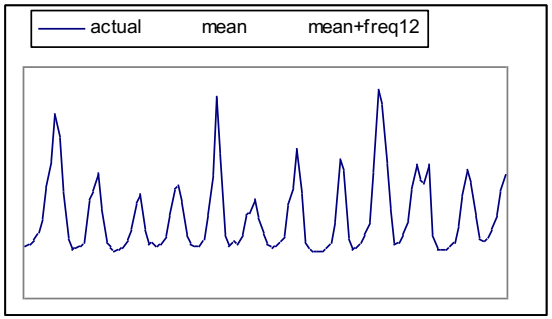
Q1: Examples, for 'interesting'?

Problem

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



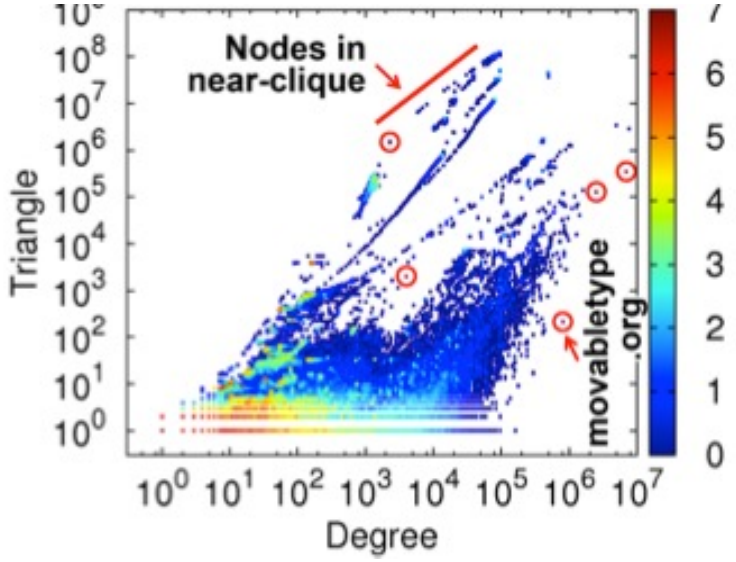
- Allow fast, approximate queries, and
- Find rules/patterns



Q1: Examples, for 'interesting'?

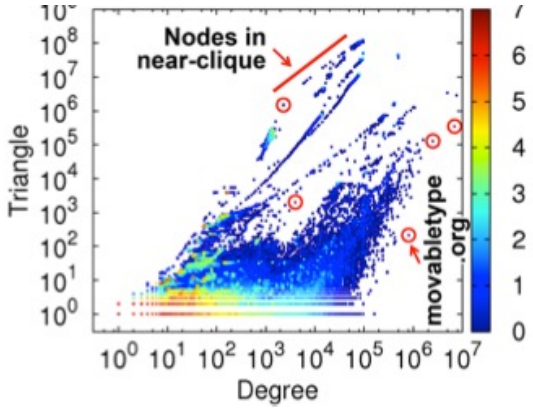
Sample queries –cont'd

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



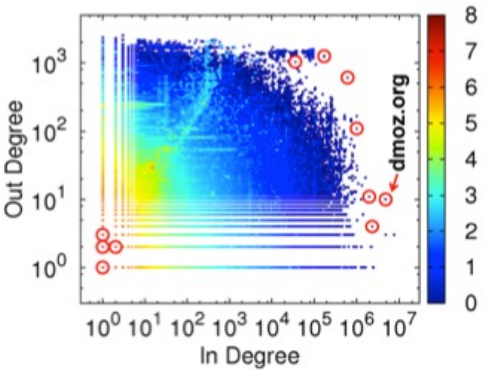
Sample queries –cont'd

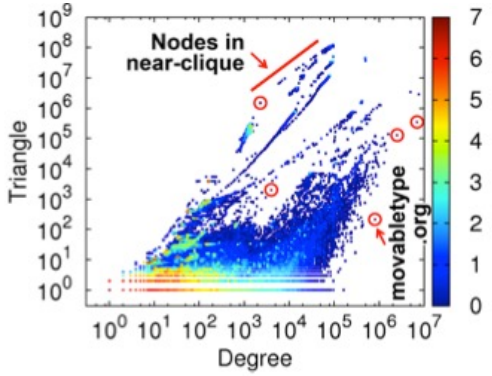
- Rule discovery
  - Clusters (of branches; of sensor data; ...)
  - Forecasting (total sales for next year?)
  - Outliers (eg., unexpected part failures; fraud detection)

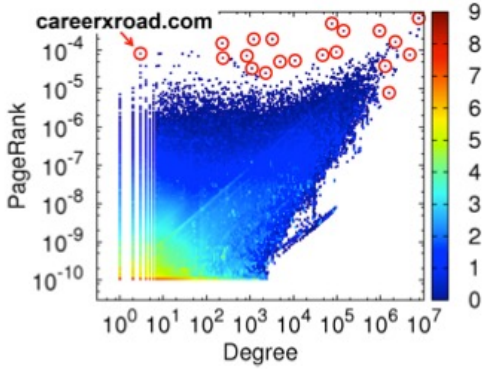


Carnegie Mellon

Example:





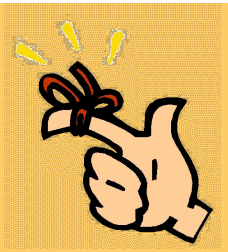


YahooWeb:

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

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

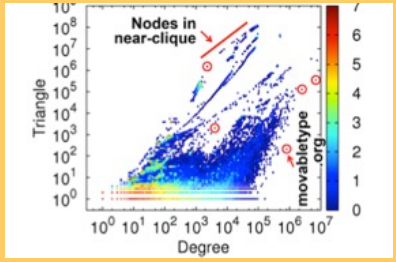
U Kang, Jay-Yoon Lee, Danai Koutra, and Christos Faloutsos. *Net-Ray: Visualizing and Mining Billion-Scale Graphs* PAKDD 2014, Tainan, Taiwan.

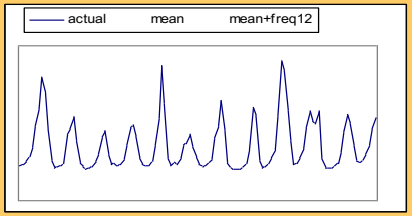


Important Observation:

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?

Detailed Outline

Intro to DB

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

Detailed Outline

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`

Carnegie Mellon

How do DBs work?

linux% sqlite3 mydb # mydb: file

sqlite> create table student (

ssn fixed;

name char(20));

student	
ssn	name

Carnegie Mellon

How do DBs work?

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

sqlite> select * from
student;

student	
ssn	name
123	Smith

Carnegie Mellon

How do DBs work?

sqlite> create table takes (

ssn fixed,

c_id char(5),

grade fixed));

takes		
ssn	c_id	grade

How do DBs work - cont'd

More than one tables - joins

student	
ssn	name

takes		
ssn	c_id	grade

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?

student	
ssn	name

takes		
ssn	c_id	grade

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

student	
ssn	name

takes		
ssn	c_id	grade

Carnegie Mellon

SQL-DML

General form:

select a1, a2, … an

from r1, r2, … rm

where P

[**order by** ….]

[**group by** …]

[**having** …]

Aggregation

Find ssn and GPA for each student

student	
ssn	name

takes		
ssn	c_id	grade
123	603	4
123	412	3
234	603	3

Aggregation

Find ssn and GPA for each student

student	
ssn	name

takes		
ssn	c_id	grade
123	603	4
123	412	3
234	603	3

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

Aggregation

sqlite> select ssn, avg(grade)
from takes
group by ssn;

takes		
ssn	**c_id**	**grade**
123	603	4
123	412	3
234	603	3

ssn	avg(grade)
123	3.5
234	3

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?

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 2hours?

A: build an index

Q': on what attribute? A: `ssn`

Q'': what syntax? A: `create index`

What if slow - #2?

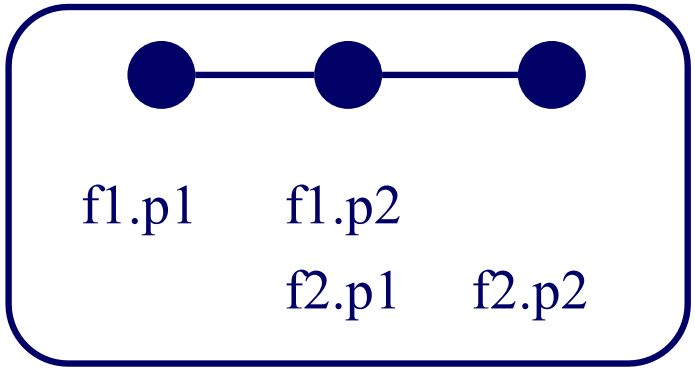
sqlite> create table friends (p1, p2);

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;

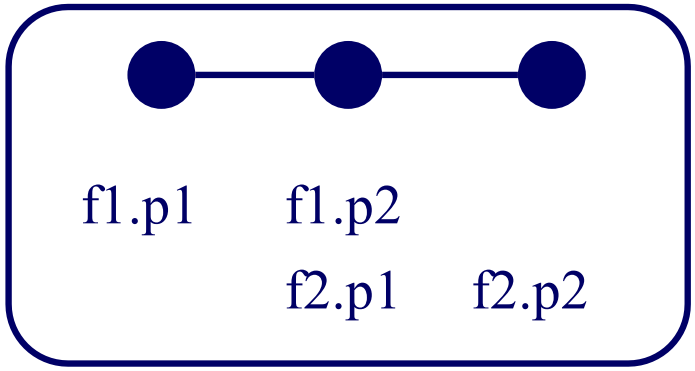


Q: too slow – now what?

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`

Carnegie Mellon

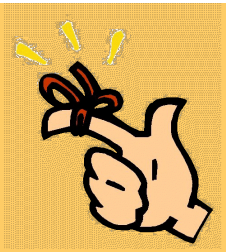
Long answer:

- Check the query optimizer (see, say, Ramakrishnan + Gehrke 3rd edition, chapter15):

Raghu Ramakrishnan, Johannes Gehrke, *Database Management Systems*, McGraw-Hill 2002 (3rd ed).

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 - @ linux.andrew
- Ramakrishnan + Gehrke, 3rd edition
- 15-415/615 web page, eg,
  - http://www.cs.cmu.edu/~christos/courses/dbms.F16

We assume known:

- B-tree indices
- www.cs.cmu.edu/~christos/courses/826.F19/FOILS-pdf/020_b-trees.pdf
- Hashing
- www.cs.cmu.edu/~christos/courses/826.F19/FOILS-pdf/030_hashing.pdf

- (also, [Ramakrishnan+Gehrke, ch. 10, ch.11])