Note:

1. Review resolution and data fusion ppt, discuss which one or two method will be implemented in our pro.
2. Individual database,
3. Normalize data format within code

Entity Resolution

Atomic string matching;

Decision Tree;

Unsupervised Learning;

Clustering:(QI)

Based on release year

**NOTE** from Derek: a very SERIOUS problem found. There are many actors missing the value of birthday, even for the top 3000 popular actors. Below is the the condition of data from IMDB:

> db.actor.find({birthday: null}).count()

4419

> db.movie.find({Year: null}).count()

672

So, we need to think out a way to correctly cluster, or just simply pairwise match between two sorted Array.

transitivity

OUTPUT: 2d array: Array[clusterID][artificial KEY]

Pairwaise Matching(GUANHAO)

For Actor: Jaro->Birthday->Place of birth->Jaccord

**NOTE** from Derek: We should think there could be nil value in every field except name. If we follow this decision flow, the desicion is kind of many and very hard to find a good strategy. So I think out another procedure: 1) Name && Birthday: match => TRUE; else, compute the weighted distance between the two Actors, if the similarty is high than some threshold, then they match.

For Movie: Jaro->release year-> jaccord-> director

Note: the same as Actor.

OUTPUT: 2d array Array[ID][entitiy ID]

Data Fusion:(YUE)

Reference to details as follow:

Portal Schema

Update on April 1st:

I will try to use Mongodb as my database. Here are some references:

<http://chenzhou123520.iteye.com/blog/1637397>

<https://docs.mongodb.org/getting-started/shell/introduction/>

Beginning Video from imooc.com

<http://www.imooc.com/learn/295>

For data fusion and data resolution, we have to come up two methods for each as mentioned in professor Chen’s letter…..

Movie = {

(index) **title: String** space as seperator;  **(1. Voting, 2. Longest, 3. Trust score)**

(index) **year: Integer; (1. Voting, 2. Longest, 3. Trust score)**

(index) rating: (2.1); weighted score

(index) **directors: Array; union**

casts: Hash {actor:role}; union

**main\_casts: Array; union**

total\_time: Integer (mins); voting

languages: Array of String ([‘English’, ‘Chinese’]); standardlization

alias: Array of String; union

country: Array of String? union

genre: Array of String; union

writers: Array of String; union

filming\_locations: Array of String; null->non-null

keywords: Array of String

}

Actor = {

(index) **name: String** (space as seperator)**; (1. Voting, 2. Longest, 3. Trust score)**

**birthday: String** (yyyy-mm-dd)**; (1. Voting, 2. Longest, 3. Trust score)**

**gender: String** (“F” / ”M” / null)**; (1. Voting, 2. Longest, 3. Trust score)**

**(index) place\_of\_birth**: eg “Oklahoma City, Oklahoma, USA” **(1. Voting, 2. Longest, 3. Trust score)**

**nationality: (1. Voting, 2. Trust score)**

known\_credits: Integer

adult\_actor: Boolean

years\_active: String (“1990 - 2000” / “1995 - “ )

alias: Array of String

~~relative: Hash {name: relation}~~

biography: String

known\_for: Array of Hash

}//NULL: choose a non-null value from one database

//difference: union

Relationship:

{mid, aid, role}

每个人的actor的小dataset要有至少8个attributes，然后和在一起的actor的大dataset要有至少10个attribute

joint database是movie＋actor＋relationship

Mongodb->html

(most popular, highest rating, …, -> rating 最高

(most famous, ….,..)-> 演出最多电影