

# **TT Interactive Data Mining Division**

## **Earthquake analysis and predictions**

**Presented by:**

Victor Chan

Kenneth Cheung

Brian Lee

Karen McVay

Thanh Truong

Linh Vu

CS 157B Section 1

Professor Lee

May 11, 2006

# Table of Contents

<u>Section</u>	<u>Name</u>	<u>Page</u>
<u>1.</u>	<u>Introduction</u>	
1.1.	Main ideas and goals	
<u>2.</u>	<u>Implementation</u>	
2.1.	Input and data structure	
2.2.	Algorithms	
2.4.	Graphical user interface	
<u>3.</u>	<u>Testing</u>	
3.1.	Results	
3.2.	Conclusion	
5.	Documentation and Code	

## **1. Introduction**

It is a coincidence that this year marks the 100<sup>th</sup> anniversary of San Francisco's great Earthquake of 1906. Like any other force of nature, earthquakes are an ever-present threat to the survival of humanity. It is highly difficult, if not impossible to know when an earthquake occurs before it happens. Researchers all over the world have studied earthquakes to a great extent in order to evaluate the devastation they caused and to hopefully use this data to understand more about them. This will increase people's preparedness for the next big one.

### **1.2. Main ideas and goals**

Our group, TT Interactive, is creating a computer system that is able to predict earthquakes. To do this, we are using the concept of data mining to gather data on earthquakes and examine them. Because there is so much data on earthquakes, we have decided to focus on the Bay Area. Thus, we are only using data located in this region for input.

We will enter as much data as we can, sort through it, and divide it into specific categories. The results we find will be used as a guide to determine future earthquakes.

## **2. Implementation**

### **2.1. Input and data structure**

We acquired our data from the Northern California Earthquake Data Center website (<http://www.ncedc.org>). In searching on the website, we used the latitude and longitude of the Bay Area as criteria in order to obtain results to enter into our data mining system. Because we focused on data from earthquakes localized within the Bay Area, we filtered the input so that we would only accept the recorded earthquakes from within the boundaries. The boundaries we chose were between 123 W (or -123) and 122.5 W (or -122.5) for the longitude and between 38.5 N and 37 N for latitude.

The data is originally stored in a text file. We extracted the data from the website and it resulted with 2 files. The first file, called data1.txt, contained 1052 entries. The second

data file, called data2.txt, contains 11086 entries. We combined the two files into data3.txt, which has a total of 12138 entries. This is the file that we read the data from. The individual data objects are stored in a data type we created called a GeoLocation. Each GeoLocation contains all the attributes for each item of earthquake data such as the location, date, and magnitude. When the data3.txt file is read, a class called DataReader opens the file, reads the data, creates a new GeoLocation for each record, and stores them in a collection called GeoGraph. The GeoGraph uses a built-in Java collection called ArrayList to store the GeoLocation objects. It is very easy to traverse through an ArrayList using a for-each loop, so accessing all the data would be easy.

## 2.2. Algorithm

### 2.2.1. Collection and filtering

#### **STEP 1: Collection of data and Representation of data in Java**

1. A sample data taken from <http://www.ncedc.org/ncedc/catalog-search.html> is as follows:

Date	Time	Lat	Lon	Depth	Mag	Magt	Nst	Gap	Clo	RMS	SRC
1966/07/01	09:41:21.87	35.9427	-120.4698	12.39	3.20	Mx	7	171	20	0.02	NCSN
1966/07/02	12:08:34.25	35.7867	-120.3265	8.99	3.70	Mx	8	86	3	0.04	NCSN
1966/07/02	12:16:14.95	35.7928	-120.3353	9.88	3.40	Mx	8	89	2	0.03	NCSN
1966/07/02	12:25:06.12	35.7970	-120.3282	9.09	3.10	Mx	8	101	3	0.08	NCSN
1966/07/05	18:54:54.41	35.9198	-120.4567	8.14	3.10	Mx	9	161	14	0.04	NCSN
1966/07/27	08:12:00.29	35.9088	-120.4383	8.19	3.00	Mx	10	158	12	0.02	NCSN
1966/08/03	12:39:05.79	35.8137	-120.3527	6.59	3.40	Mx	10	131	2	0.05	NCSN
1966/08/07	17:03:24.21	35.9345	-120.4577	9.98	3.00	Mx	12	152	15	0.07	NCSN
1966/08/19	22:51:20.16	35.9132	-120.4255	2.45	3.30	Mx	6	165	11	0.09	NCSN
1966/09/07	00:20:51.62	36.0165	-120.0088	9.54	3.40	Mx	13	262	29	0.08	NCSN
1968/01/12	22:19:10.36	36.6453	-121.2527	5.82	3.00	ML	16	150	2	0.13	NCSN

2. For the purpose of our project, each row represents a data point. A data point has only the following variables:
  - Date
  - Time
  - Longitude
  - Latitude
  - Magnitude
  - Ignore other variables
3. Special case: In case two different rows have the same longitude and latitude, we still consider them as two SEPARATE DATA POINTS.

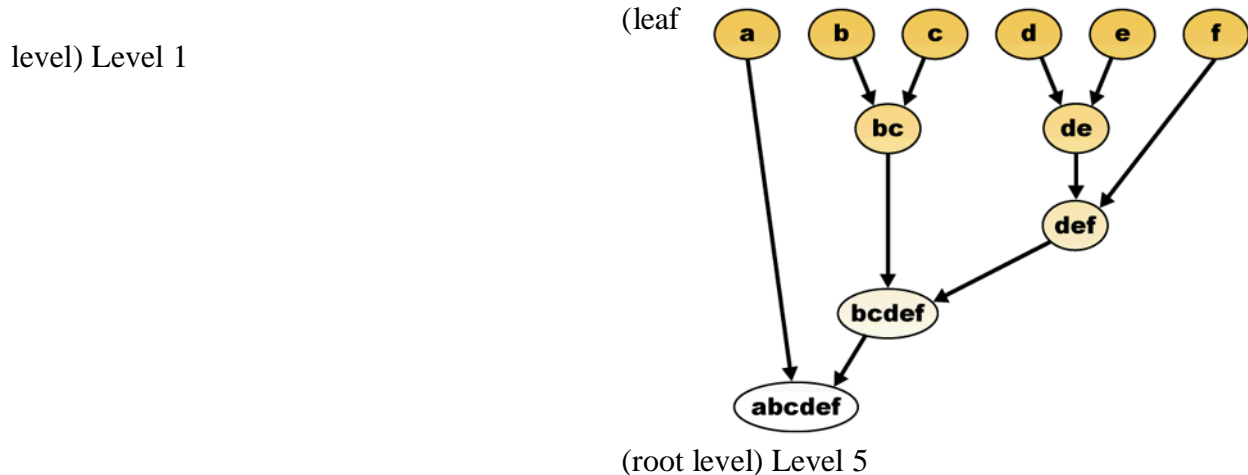
- I leave the specific region to Brian: Northern California, Bay Area, Santa Clara County, San Andreas fault line, etc. (How about the one Ken provided to us)

### **STEP 2: BUILD THE MINIMUM SPANNING TREE**

- For the purpose of the MST, I represent each data point (explained above) as a vertex ( $x$  = longitude,  $y$  = latitude), the physical distance between two data points as weight of the edge.
- Using Prim Algorithm to build MST (pick up any vertex and go from there. It's up to Brian to decide to which one)

### **STEP 3: GROUP VERTICES INTO CLUSTERS**

- I utilize a technique called "Agglomerative hierarchical clustering". It is basically a tree. The following picture illustrates how it works:



In the beginning, at the lowest level (leaf level), each vertex is a CLUSTER itself. So, {a}, {b}, {c}, {d}, {e}, {f} are level 1 clusters.

For level 2 clusters, the max physical distance between any two vertices within a level 2 cluster cannot be more than a pre-defined value (I propose 1 mile).

Special case 1: Let's say the distance between b and c is 1.5 (< 2 miles). At the same time, the distance between c and d is also 1.5 mile (< 2 miles). Assume that the distance between b and d is 2.4 miles (> 2 miles). Therefore, b, c, and d cannot be in the same cluster.

If the program sees b and c first, then b and c is in the same cluster and d is not. d can be alone or associated with other clusters (provided the max distance requirement is met).

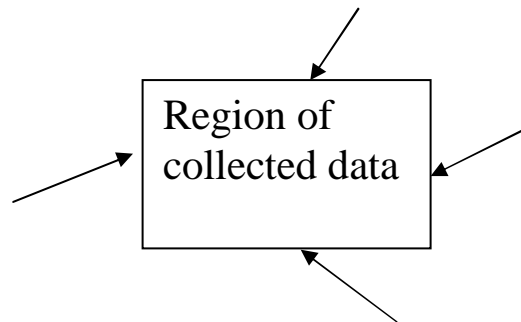
Special case 2: A cluster may have only one vertex

- Level 1 : leaf level
- Level 2 : 2 miles
- Level 3 : 4 miles
- Level 4 : 8 miles
- Level 5 : 16 miles
- Level 6 : 32 miles
- Level 7 : 64 miles
- Level 8: 128 miles (root level)

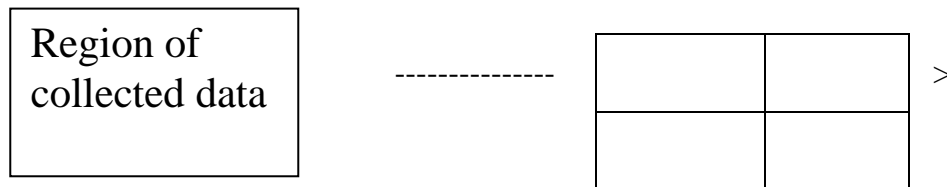
Special Note on root level: the root level cluster has ONLY ONE CLUSTER and this cluster contains every vertex in our data set.

**STEP 4: EARTHQUAKE PREDICTION**

1. The endpoints in the picture below are the data points that Ken provided to Brian (North, South, East, West). Therefore, all data points should fit into this rectangle. Note that just because data points fit in this rectangle, it doesn't mean the data points are spread out evenly.



2. Now, we need to draw an “earthquake map” of the mentioned above “rectangle”. We organize this rectangle into a “chessboard”:



Note this chessboard has exactly 4 cells and 9 PREDICTABLE EARTHQUAKE points. (This is just an example, not used for our real project.)

For the purpose of our project, I suggest each “cell” is 2 miles wide and 2 miles high. (Note: 2 miles is also max distance for level 2 in STEP 3).

3. Determine the magnitude for each PREDICTABLE EARTHQUAKE point.

Let's call "A" a PREDICTABLE EARTHQUAKE point in the "chessboard".

Level 1 (leaf) "A" belongs a level 1 cluster (implies one data point)

Level 2 "A" belongs a level 2 cluster

Level 3 "A" belongs a level 3 cluster

Level 4 "A" belongs a level 4 cluster

Level 5 "A" belongs a level 5 cluster

Level 6 "A" belongs a level 6 cluster

Level 7 "A" belongs a level 7 cluster

~~Level 8 (root)~~ (don't use this because all data points belong to this cluster anyway no difference).

Note "A" cannot belong to two or more clusters at the same level; it can only belong to exactly one cluster at any level.

## 2.2.2. How does the algorithm work?

### Step 1: Filter data points

- We get 73 data points for the coverage area (Bay Area):

South 37 North 38.5 West 122.5 East 123

- The coverage area looks like a rectangle. Therefore, we can logically divide the coverage area into a "chessboard" with the following specifications:

width of this chessboard : 4 cells

length of this chessboard : 15 cells

total cells on this chessboard : 4 cells X 15 cells = 60 cells

- Next, we analyze the distributions of the 73 data points among these 60 cells. Amazingly, one cell (South 37.6 North 37.7 West 122.5 East 122.625) stands out. Of the 73 data points spreading these 60 cells, 31 points are located with cell !!!

- As a result, we focus on this one cell and its 31 data points only. Since this cell is relatively small (width 5.8 miles x length 6.9 miles), we consider this cell as one geometric point.

- For many other reasons, we reduce the set of these 31 data points to 17 data points. These data points must be sorted by time (oldest to latest)

### Step 2: Derive magnitude function $m(t)$ (min 0 max 10) with respect to time $t$ (day).

- The 17 data points (mentioned above) date between 1957 and 1997.

- Using “Natural Cubic Spline”, we can derive function  $m(t)$  as a 3<sup>rd</sup> degree polynomial:  $m(t) = A_i + B_i(t - t_i) + C_i(t - t_i)^2 + D_i(t - t_i)^3$

with:  $t_i$  : time (day) that each data point occurred. This time must be sorted.  
 $t_0 < t_1 < t_2 < t_3 < \dots < t_{15} < t_{16}$

$t$  : time (day) variable time of the function  $m(t)$ . We use this variable  $t$  to graph the 3<sup>rd</sup> degree polynomial  $m(t)$  at equally spaced points between years 1956 and 2016

$A_i, B_i, C_i, D_i$  are constant coefficients of the polynomial  $m(t)$ .

Note that the polynomial  $m(t)$  has different coefficients  $A_i, B_i, C_i, D_i, t_i$  depending on where  $t$  value is.

- Solve for coefficients  $A_i, B_i, C_i, D_i$  using certain basic conditions.
- Once  $A_i, B_i, C_i, D_i$  are known, we can evaluate function  $m(t)$  at time  $t$  in the future ( $t$  can be years 2007, 2008, ..., 2016) to predict earthquakes.

## 2.4. Graphical User Interface

Our Graphical User interface consists of a display of a California map. To get started, a user clicks on the “Browse for Data” button. This will bring up an open file menu where the user will look for a file in the file system that contains the earthquake data. For our project, we have 3 files to choose from (data1.txt, data2.txt, data3.txt). Once the user chooses an appropriate file, then he or she will click on the “Process Request” button. At that point, the JFrame paints on the map the data points collected in the file in red. It then will display predicted points in blue

## 3. Testing

### 3.1. Results

In our focus in examining the data in the Bay area, we found that one section of the map have the most frequent earthquakes.

We found that the area has 31 data points, but there are only 17 distinct days. Some days had more than 1 quake. Using that data, we have a table.

The following is a table of predicted results

Years after 2006	Predicted magnitude
0	3.0461665274845453
1	3.267072650080894
2	4.02043670627544
3	3.191180541462603
4	3.5376649288593693
5	1.6831067267878232
6	3.8116777592589934
7	4.121069386949197
8	4.093469740889404
9	3.8406060416298895
10	3.4735161076442145
11	3.103237757405943
12	2.840808809388636
13	2.7972670820658543
14	3.0836503939111637
15	1.6183288113298469
16	1.3711027632545818
17	1.2722149334461186
18	1.3620699971607397
19	1.681072629654711
20	2.269627506184303
21	3.057471063107378
22	2.6029879676918712
23	2.1268087954242896
24	1.6423354430869905
25	1.1629698074623311
26	0.7021137853326678
27	0.27316927348035747

28	-0.11046183131224296
29	-0.43537763226277737
30	-0.688176232588888
31	-0.855455735508218
32	-0.9238142442384114
33	-0.879849861997112
34	-0.7101606920019607
35	-0.4013448374706039
36	0.05999959837931801
37	0.6872745123301609
38	1.493881801164278
39	2.4932233616640342
40	-43430.99830950997
41	2.55420392954236
42	26.55220558325543
43	146.0034566257472
44	433.5573646256441
45	961.8633371515729
46	1803.5707817721595
47	3031.3291060560305
48	4717.787717571813
49	6935.5960238881335
50	9757.403432573617
51	13255.859351196892
52	17503.613187326584
53	22573.314348531323
54	28537.612242379728
55	35469.15627644042
56	43440.59585828205
57	52524.58039547322
58	62793.75929558258

### 3.2. Conclusion

Our results may be accurate for up to 27 years. After that year, the predicted magnitudes become negative and thus not useful. Later years will go far into extreme positive. This is just a graph with 17 points so may need more points to be more accurate. The graph that we used may need more points to be more accurate.