CS615: Group 4
Representative Skylines with Noisy Comparisons

Pranshu Gupta
13493
pranshug@iitk.ac.in
Dept. of CSE

Rahul Tudu
13538
trahul@iitk.ac.in
Dept. of CSE

Indian Institute of Technology, Kanpur

Final report
21st November, 2016

Abstract

Most systems focus on efficiently finding out skylines and the user is presented with the entire set. Now, if the skyline set it large, it becomes the job of the user to manually inspect the points and pick out the most informative ones. These are the representative skyline points. Our system assumes a computation model where the data items can only be compared through noisy comparisons. In this model, for each comparison we have a probability with which the first item will dominate second. A generic system won't work with such model. We intend to implement a system which finds out representative skylines on data with noisy comparison model

1 Problem Statement

The objective of the implemented system is to find out Representative Skylines on data with Noisy Comparisons.

Input	Description
D	Number of Dimensions
D_i	Number of possible values in dimension i
M_i	Dominance probability matrix for values in dimension i. $\mathrm{M}_i[p,q] = Prob(p \succ q)$
e	Error tolerance for a single comparison
N	Number of samples in the data
$Data$	Samples for which the skyline has to be computed

Output	Description
R	Skyline Set in Representative Order

Notation	Description
S	Skyline Set
W_i	A sorted instance of the values in dimension i as per M_i
LD	Data labeled according to W
DS	Dominance sets for the skyline points
J	Jaccard distance matrix for the skyline set S

1.1 Related Material

The work by Groz and Milo [1] talks about finding skylines for data with noisy comparisons based on the model described in the paper by Feiget et al. [2]. A procedure to find out representative skylines on absolute data is described in the paper by Valkanas et al. [3]. We combine these two works [1], [3] to find out representative skylines for data with noisy comparisons.

2 Approach

Finding representative skylines with noisy comparisons involves the sorting the possible values in the dimension according to the noisy comparison model. Our system does merge sort with a custom comparison operator. This operator takes two points p and q and the uses the dominance probability matrix for the corresponding dimension, and does repetitive comparisons with each comparison giving the result according to the value in M_i[p,q]. The result given by the operator is decided by the majority vote of the repetitive comparisons. So, if majority says $p \succ q$ then the operator says $p \succ q$. [2]

Algorithm 1 RSNC

Input: D, D_i, M_i, e, N, $Data$
Output: Result set R

1: Foeach i: $W_i \leftarrow D_i$.Sort(M_i, e)
2: $LD \leftarrow Data$.Label(W)
3: $S \leftarrow LD$.SFS()
4: Foreach p in S, $DS[p] \leftarrow p$.DominanceSet(LD)
5: Foreach (p,q): J[p,q] ←JaccardDist(p,q)
6: Find the Respresentative order of S using the jaccard distance based diversification algorithm and store the result in R
7: Return R

After the sorting of the possible values in each dimension is done, the attributes of the data points are labeled accordingly. Say, if a value p in dimension d comes at position i in the sorted order then all the instances of p in dimension d the data points will be labeled as i.

Now, the attributes so labeled are assumed to represent the numeric values of the data points and thus, we can utilize any skyline finding algorithm on the labeled data. Our system uses SFS algorithm for this purpose. [1]

After we have found the skyline set, we find the dominance sets of each skyline point. These dominance sets are used to find out the pairwise Jaccard distances of the skyline points. Jaccard distance is defined as the ratio of sizes of the intersection to the union of the dominance sets of the concerned points.

Finally, to find the representative order of the skylines we follow the Jaccard Distance based diversification algorithm. The skyline point with largest domination is chosen as the initial point in cluster A. Then in every iteration, we choose the next skyline point q such that minimum Jaccard distance of q to any p in the cluster A is maximized. We already have computed jaccard distances so we don't use min-hashing for estimation. Ties are broken using the domination score. [3]

The main algorithm followed by our system is RSNC (see Algorithm 1)

3 Results

We have tested our implementation for different use cases. For example, how does the size of skyline set change when we increase the number of samples in the data. How much time does the system take to compute the result with varying sample sizes. Please see Figure 1 and Figure 2 for the results. Note that both of these plots were obtained on varying the sample size while keeping the comparison model constant (with 5 independent dimensions, each having 20 possible values) and error tolerance (e) equal to 0.1 with 100 runs for each sample size (for sizes above 10000, 10 runs were done).

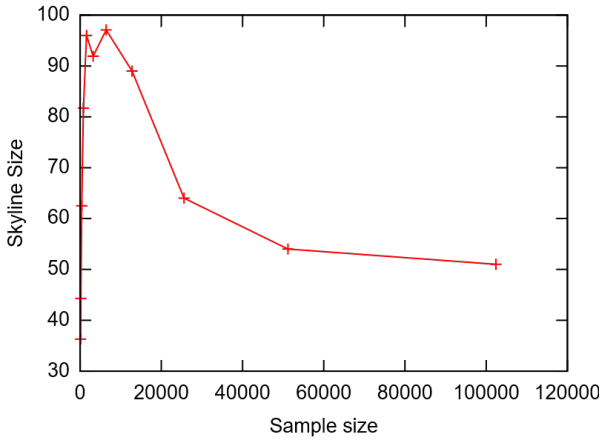

Figure 1: Average Number of Skylines vs Sample Size

We can see in Figure 1 that the number of skylines first increases as we increase the number of samples but for very large number of samples it starts decreasing and then becomes stable.

This is because the domain of possible values in each dimension is finite and as we go on increasing the number of samples, the possibility of a dominating point being there in the data first increases and for very large set most of the dominating point are there so the number of skylines flattens out. Initially, the number of skylines does increase when the data size is smaller as in that case most of the points are of similar calibre and hence, incomparable.

So, now we increase the number of possible values in each dimension while keeping the sample size constant. The following table shows that number of skylines increases with increase in the domain size of the dimensions. This in turn supports our claim about the no. of skylines w.r.t the sample size.

Samples	Domain Size per Dim.	No. of Skylines
25600	10	1
25600	20	63
25600	30	110
25600	40	161
25600	50	235
25600	60	236
25600	100	247

We also analyzed how the results changed on varying the number of dimensions while keeping the number of

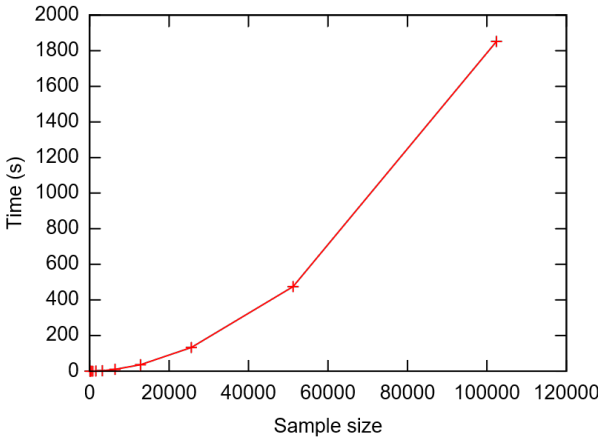

Figure 2: Time taken vs Sample Size

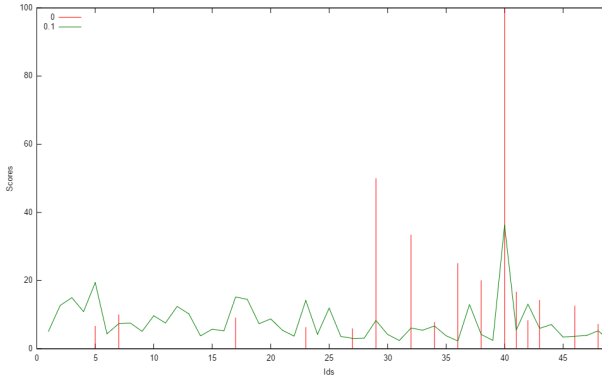

Figure 3: Score of Each ID

samples constant. The following table shows the results when the number of samples (N) was 12800, e = 0 and domain size for each dimension D_i = 20. The number of skylines increases with increasing number of dimensions. This is what we call the curse of dimensionality.

Samples	Dimensions	Skylines	Time(s)
12800	5	59	30.86
12800	6	259	45.96
12800	7	823	190.67
12800	8	1504	1212.11
12800	9	2776	12323.2

The total time taken to compute the result increases with the number of samples as expected.

There is one more important parameter in our implementation, the error tolerance of the comparison model (e). We also tested how the results differ on changing the value of e.

On increasing the tolerance of the comparison operator the skyline many other points which were not in the true skyline set (e = 0) also creep into the result set. Also, the representative ordering does not seem to be preserved. As we can see in Figure 3 which plots the scores of each skyline point in with e = 0 and e = 0.1.

Score of a point is defined as the sum of the reciprocal of representative ranks of the point in multiple runs of program. So, better ranks will lead to higher score. Rank is the index of the skyline point in the representative order.

We can see that the score has been distributed across points with e = 0.1. One of the reason can be that we are applying opposite sided tolerance in each comparison. That is, if dominance probability is greater than 0.5 then we subtract e from it while doing the comparison, otherwise we add e. So, we are making domination more difficult. That is why other points are also becoming skylines.

However, if we keep e = 0 the skyline set remains constant as well as the representative order is preserved in all the runs.

4 Conclusions

We have successfully implemented and analyzed a system that can give representative skylines on data with noisy comparisons. Suggested future improvements and possible extensions of this work are as follows:

- Optimize the representative order of skylines finding part. The skyline diversification algorithm. Can use the jaccard distance estimation technique.
- Enable the program for distributed data-sets
- As of now, all the dimensions are assumed to have noisy comparison model. We can add the ability to allow some of the dimensions to have absolute comparisons.
- Add ability to read all kinds of input attributes strings, numbers etc. for noisy comparison model.
- Add the ability to take the comparison model as a function that can be used to compute the dominance probability (if applicable) and thus avoid storing the dominance probability matrices for each dimension. This will considerably reduce the memory usage. This will also enable us to take inputs which have dimensions with infinite sized domain.
- Implement other methods to find representative skylines and enable the user to choose one of them at run-time

5 Appendix

Sample	Avg. No. of Skylines	Time Taken (sec)
100	36.3	0.0775685
200	44.3	0.092131
400	62.5	0.133191
800	81.7	0.296066
1600	96	0.891572
3200	91.9	2.89024
6400	97.1	10.165
12800	89	35.932
25600	64	132.972
51200	54	474.988
102400	51	1852.29

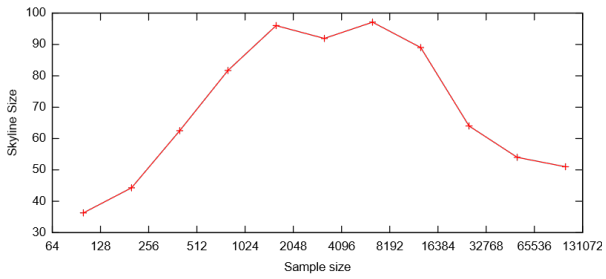

Figure 4: Avg. no. of Skylines vs Sample Size (log scale)

6 Implementation Details

We have implemented the entire system in C++11. We have strictly followed the Object Oriented Model of programming. The source code is publicly available on Github under GNU GPL v3.0. You can find the repository at https://github.com/Pranshu258/RSNC

7 Usage

- Prepare the Input files
  - input/dims : write the number of dimensions in the first line followed by D lines such that i'th next line has the domain size of i'th dimension
  - input/samples : write the number of samples in the data
  - input/tolerance : the error tolerance of comparison operator. Choose 0 for best results.
  - input/data : D lines, line i has the i'th dimension attribute value for all the items in the data sepearted by spaces
  - input/model : D matrices, matrix i defining the comparison model for i'th dimension. matrix[i][p,q] will contain the probability of p dominating q in dimension i.
- Run the program
  - If you already have created all the input files then issue the following command
    $ bin/main f f
  - If you have only created the input files for dims, samples and tolerance and want the program to generate the data and model files for you, issue the following command.
    $ bin/main t t
  - If you have created the input files for dims, samples tolerance and model and want the program to generate the data file for you, issue the following command
    $ bin/main f t
  - If you have created the input files for dims, samples tolerance and data and want the program to generate the model file for you, issue the following command
    $ bin/main t f
  - The program will print the ids of the skyline points in their representative order, along with the number of Skylines and the total time spent in doing the computations.
- Please note that the attribute values in the data file must agree with the number of dimensions and the domain size of each dimension described in the corresponding input files

8 Division of Labor

Std 1 - Pranshu Gupta: 50 and Std 2 - Rahul Tudu: 50

References

[1] Benoit Groz and Tova Milo. *Skyline Queries with Noisy Comparisons*. ACM SIGMOD/PODS, May 2015, Melbourne, Australia.

[2] Uriel Feiget, Prabhakar Raghavant, David Peleg and Eli Upfal *Computing with Noisy Information*. Society for Industrial and Applied Mathematics, 1994

[3] George Valkanas, Apostolos N. Papadopoulos and Dimitrios Gunopulos *SkyDiver: A Framework for Skyline Diversification*.