CSCI E-124 - Minimum Spanning Trees in Random Graphs

By: Tofik Mussa

I was tasked with collecting data points to determine a formula for the average weight of a minimum spanning tree for a graph with randomly generated weighted edges in dimensions 0, and 2 through 4. The purpose of the assignment was to experience the challenges with having to fully implement algorithms in a language of choice and to study the behavior/patterns of minimum spanning trees in a randomly generated graph.

I chose Java since that is one of the recommended options and I am familiar with the language. I ran into several challenges while implementing the program. I initially had a bug in my merge sort implementation and the weights were not really sorted which slowed down my algorithm and consumed a lot of memory eventually leading my program to crash. I then had a threshold function for throwing out edges which was growing proportionally to the number of vertices which was completely wrong. The maximum weight that will be part of a minimum spanning tree in a graph with large number of vertices should be small because there are more edges to throw out in a complete graph and the Kruskal's algorithm will pick smaller weight edges over them. My algorithm vastly improved when I correctly sorted the edges and fitted an inverse polynomial function to determine max weight. Another major challenge I faced was with memory management. I tried to increase the heap space size of my IDE and also increasing the page file size of my computer to no avail. I had to have a tighter upper bound for maximum weight than an identical algorithm in a language where reclaiming memory is possible. A tight bound that gradually shrunk as the number of vertices ensured that I consumed less space and have the program run in a reasonable amount of time. I recorded the number of seconds in each trial to track whether my threshold is too big and to experimentally estimate the runtime of my algorithm as a function of number of vertices and dimension.

Through too many trials, I was able to notice a repeatable pattern. The average weights stayed consistent across several trials. I compared the average weights with and without throwing out edges for smaller number of vertices and the average weights were very similar in both cases. This proved to me that I wasn't throwing out edges that would have otherwise been part of the minimum spanning tree and the edges I was throwing out had negligible effect on the average weight of the minimum spanning tree.

I used Kruskal's algorithm because it lent itself very suitable with my strategy of throwing out edges. Prim's algorithm requires that the edges to be picked should be adjacent to the spanning tree under construction, however, Kruskal's algorithm has no such restriction allowing me to pick the smallest edges regardless of their vicinity to the tree under construction. I increased the cache size of my computer as said before but trimming down the threshold had a more profound effect than the computational power of my computer. I thought about coming up with a multi-threaded program but decided but it wasn't worth the complexity. The random number generator was very repeatable for large number of trials adhering to the central limit theorem but diverged when the number of trials was low.

I first explored determining the maximum weight of a minimum spanning tree for a complete graph with small number of edges and the results I have gotten are shown below by using a graphing tool for fitting. I had surmised before the experiment that the max weight can possibly be 1 for the 0-dimensional case, √2 for the 2D case and √3 for the 3D case because it is just a Euclidean distance with unit dimensions in the maximum case.

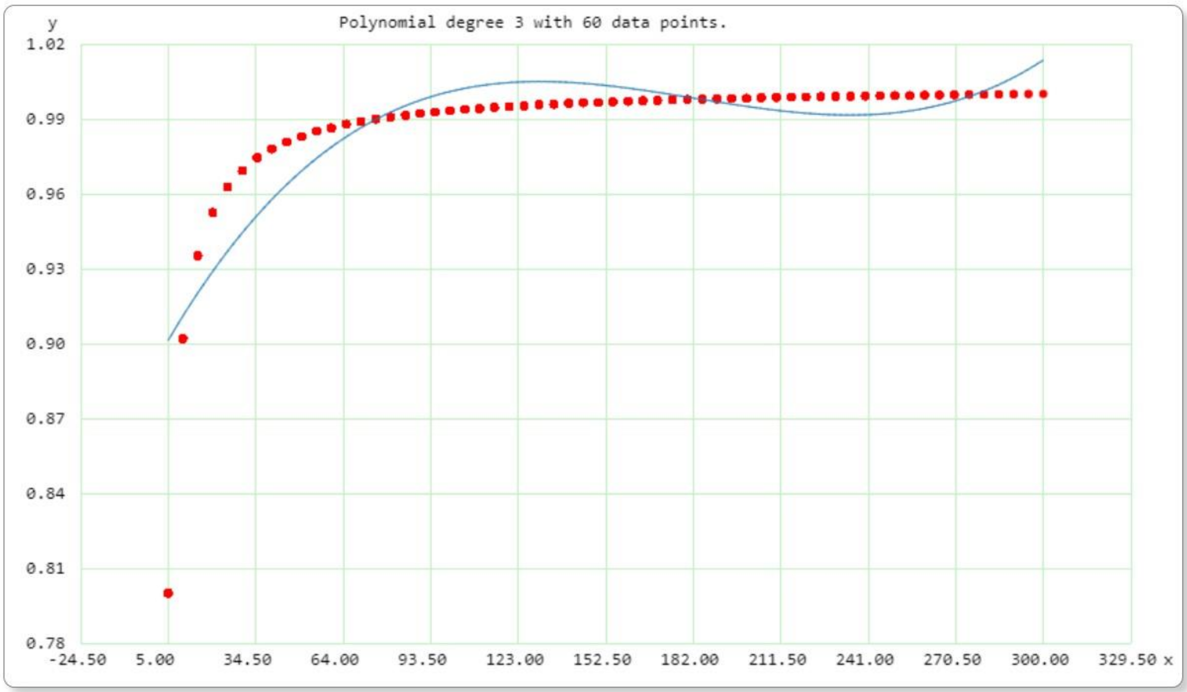

Figure 1 – 0-dimensional case max-weight graph

numVertices,maxWeight

5,0.8001866912773067

10,0.9003767994534748

15,0.933035741893147

20,0.9499656352983522

25,0.9599904131481558

30,0.9664238573477442

35,0.9715843185482775

40,0.9749634826641239

45,0.9777327246709759

50,0.9798062609011049

55,0.9819869661870678

60,0.9831964370371777

65,0.9847733752901662

70,0.9857669564062586

75,0.9867478570653213

80,0.9874912270084436

85,0.9882197063924667

90,0.9889629732106295

95,0.9895192185334941

100,0.99001277433378

105,0.990539208693413

110,0.9907949552953622

115,0.9913451396617504

120,0.9916930767515382

125,0.9919966496518662

130,0.9924211126569517

135,0.9926022768002698

140,0.9929363461234331

145,0.9931585050368292

150,0.9932730770545651

155,0.9935180129201144

160,0.9937006111574476

165,0.9939330527254672

170,0.9940549408309076

175,0.9943110281694624

180,0.9944650851837301

185,0.9945610963609568

190,0.9947227768754825

195,0.9948816277824654

200,0.9949532792921753

205,0.9951357984812352

210,0.9952774540426144

215,0.9953725541735173

220,0.9954625267106892

225,0.9955582425781416

230,0.9956527199772487

235,0.9957610083174457

240,0.9958206291398679

245,0.9959267885191906

250,0.9960245673258509

255,0.9960157283461711

260,0.9961518380953319

265,0.9962046109766503

270,0.9962962358569193

275,0.996384604339036

280,0.9963992670977591

285,0.9965016847696466

290,0.9965850599890141

295,0.996621335854151

300,0.9966511823299534

Data Point 1 – 0-dimensional case max-weight data points

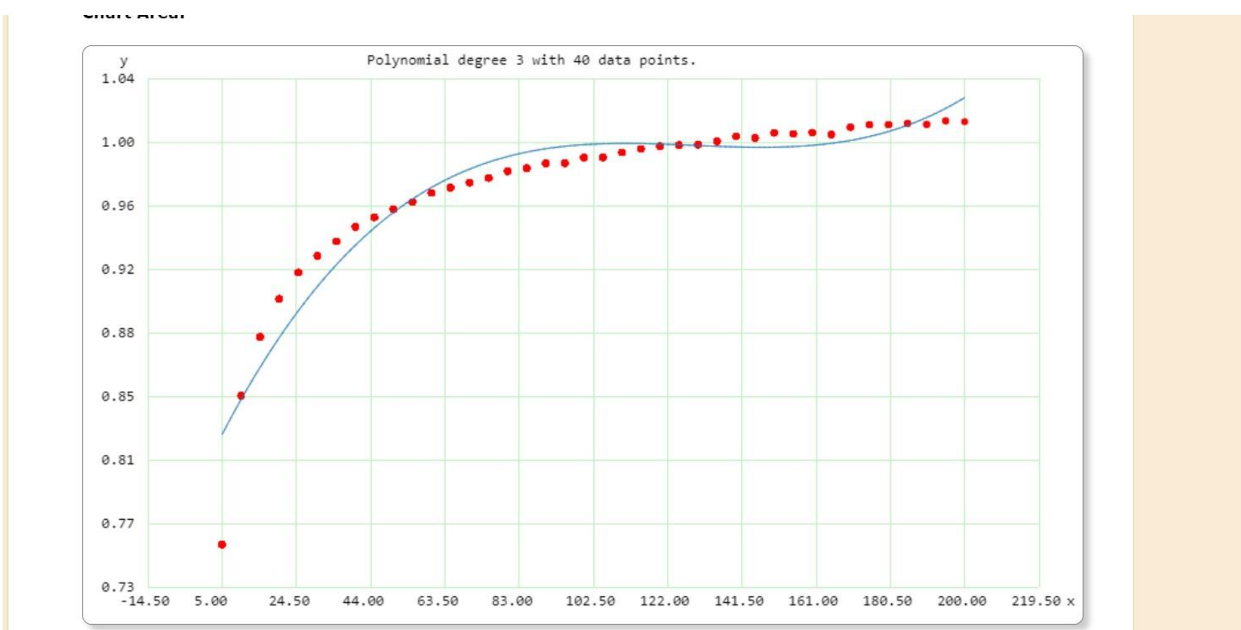

Figure 2 – 2-dimensional case max-weight graph

numVertices,maxWeight

5,0.7573168437756598

10,0.8465899393871427

15,0.8816382578074932

20,0.9044229304060339

25,0.920296268953383

30,0.930121296042204

35,0.9389509135186672

40,0.9476240210473538

45,0.953260115134716

50,0.9581156752437353

55,0.9624559050887823

60,0.9678928285986185

65,0.9710306810885668

70,0.9740322371870279

75,0.976809416809678

80,0.9808932523995638

85,0.9826608639210462

90,0.9855941891998052

95,0.9857240170836449

100,0.9891327050358057

105,0.9891603808909655

110,0.9920732222825289

115,0.9942988494902849

120,0.99583669090271

125,0.9965241845786571

130,0.9967792299240827

135,0.9988471626192331

140,1.0017498819380999

145,1.0008012814879417

150,1.003819194895029

155,1.0033063169747591

160,1.0040255785226821

165,1.002841364967823

170,1.0072297242850066

175,1.0086960266530514

180,1.0087743903249502

185,1.0095520547062158

190,1.0089411427676678

195,1.011076569122076

200,1.01053427862823

Data Point 2 – 2-dimensional case max-weight data points

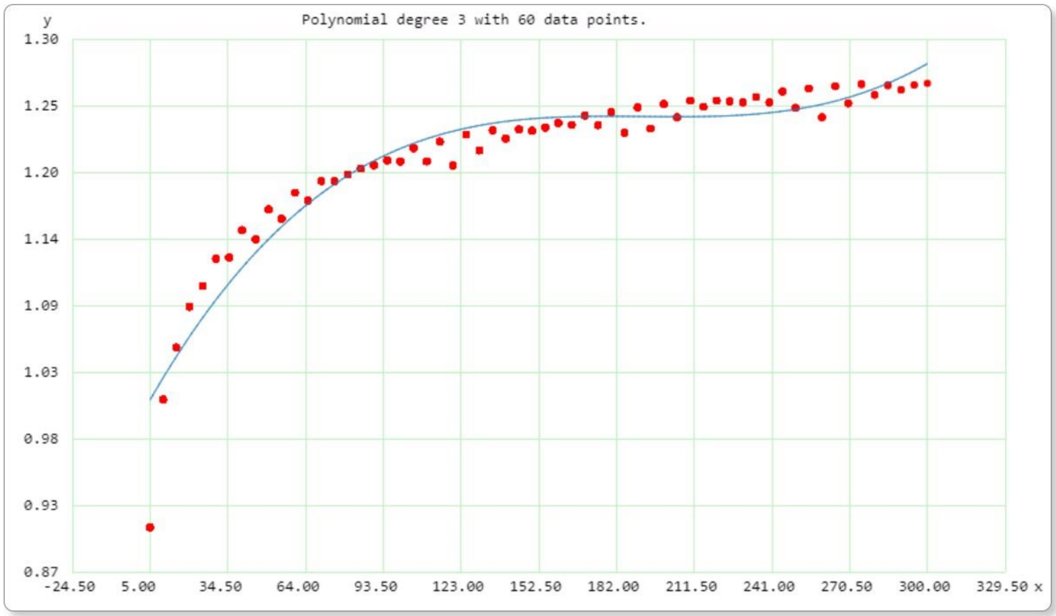

Figure 3 – 3-dimensional case max-weight graph

numVertices,maxWeight

5,0.9094209999817704

10,1.0124473600415007

15,1.0543248906646288

20,1.0871595928007207

25,1.1038694974950198

30,1.1256954361379106

35,1.126848117290502

40,1.1489345480703173

45,1.1414847157458154

50,1.1657012701348104

55,1.1580383955524816

60,1.1790002824847738

65,1.1726860837567252

70,1.18848722670662

75,1.1883521041642098

80,1.1936897182962682

85,1.1984550649842458

90,1.2011255459995351

95,1.2050180619900765

100,1.2039732253092326

105,1.2147937233365478

110,1.204203228281799

115,1.2202254210173715

120,1.2008017974042215

125,1.2259076818922057

130,1.2130125086684376

135,1.2293986491013285

140,1.2226459602218105

145,1.2301450858393341

150,1.22904715940099

155,1.2314484243147426

160,1.2351933353878681

165,1.2335715845853583

170,1.241172500550278

175,1.233423976014941

180,1.2440890606260344

185,1.227280194024947

190,1.247676072449164

195,1.2306195776285762

200,1.2505597888047157

205,1.2399410251432377

210,1.25305509208509

215,1.2481822794791249

220,1.2531916504270992

225,1.2525654080271185

230,1.2518764403487546

235,1.2561244406626284

240,1.2518765733360364

245,1.2605603745723688

250,1.2474709566808297

255,1.2630472054562183

260,1.2398400391700382

265,1.2647245303354973

270,1.2509688654147457

275,1.26648761184782

280,1.2578032579390093

285,1.2655266562813825

290,1.2619106456472238

295,1.2657850678428075

300,1.2671479195237283

Data Point 3 – 3-dimensional case max-weight data points

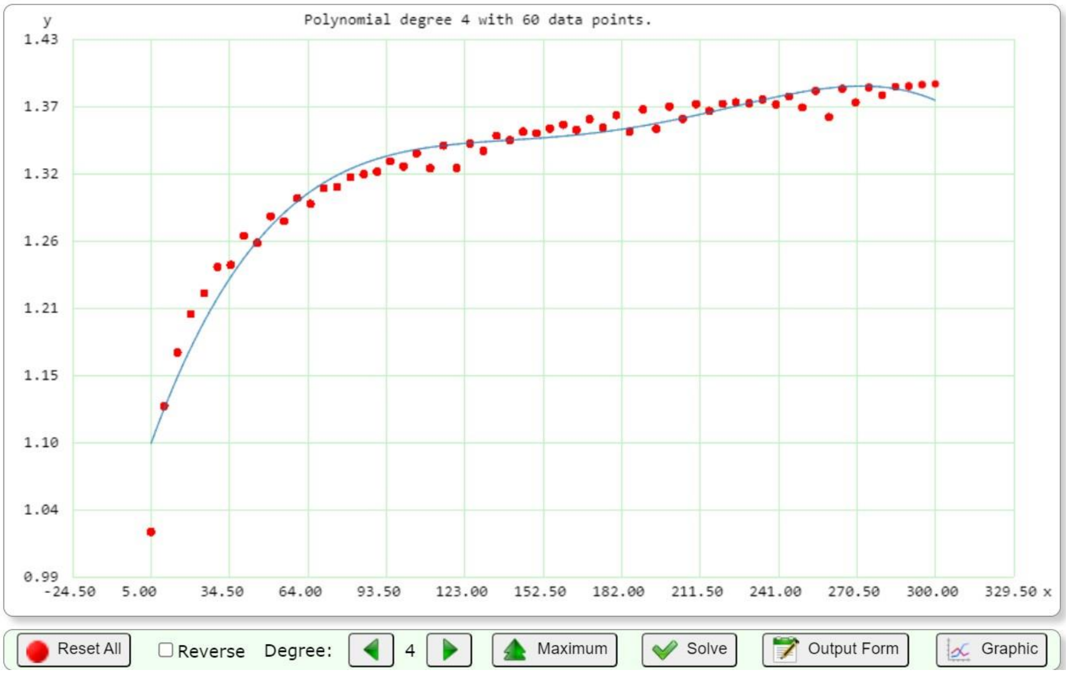

Figure 4 – 4-dimensional case max-weight graph

numVertices,maxWeight

5,0.8001866912773067

10,0.9003767994534748

15,0.933035741893147

20,0.9499656352983522

25,0.9599904131481558

30,0.9664238573477442

35,0.9715843185482775

40,0.9749634826641239

45,0.9777327246709759

50,0.9798062609011049

55,0.9819869661870678

60,0.9831964370371777

65,0.9847733752901662

70,0.9857669564062586

75,0.9867478570653213

80,0.9874912270084436

85,0.9882197063924667

90,0.9889629732106295

95,0.9895192185334941

100,0.99001277433378

105,0.990539208693413

110,0.9907949552953622

115,0.9913451396617504

120,0.9916930767515382

125,0.9919966496518662

130,0.9924211126569517

135,0.9926022768002698

140,0.9929363461234331

145,0.9931585050368292

150,0.9932730770545651

155,0.9935180129201144

160,0.9937006111574476

165,0.9939330527254672

170,0.9940549408309076

175,0.9943110281694624

180,0.9944650851837301

185,0.9945610963609568

190,0.9947227768754825

195,0.9948816277824654

200,0.9949532792921753

205,0.9951357984812352

210,0.9952774540426144

215,0.9953725541735173

220,0.9954625267106892

225,0.9955582425781416

230,0.9956527199772487

235,0.9957610083174457

240,0.9958206291398679

245,0.9959267885191906

250,0.9960245673258509

255,0.9960157283461711

260,0.9961518380953319

265,0.9962046109766503

270,0.9962962358569193

275,0.996384604339036

280,0.9963992670977591

285,0.9965016847696466

290,0.9965850599890141

295,0.996621335854151

300,0.9966511823299534

Data Point 4– 4-dimensional case max-weight data points

After determining max weight for all 4 required dimensions, I proceeded with coming up with the function to throw out edges. I trimmed down my throw out function running several experiments and I was finally happy with the results below for average weights.

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

0.9077525615692139,16,5,0,0

1.279867684841156,32,5,0,0

1.079302680492401,64,5,0,0

1.1591216564178466,128,5,0,0

1.1649361729621888,256,5,0,0

1.2132687330245973,512,5,0,0

1.212718117237091,1024,5,0,0

1.198108720779419,2048,5,0,0

1.2029218792915344,4096,5,0,1

1.1887968182563782,8192,5,0,6

1.203907024860382,16384,5,0,23

1.2010000228881836,32768,5,0,95

1.2003456950187683,65536,5,0,383

1.1986775279045105,131072,5,0,1570

Data Point 5– 0-dimensional case average weight data points

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

1.317669117450714,16,5,0,0

1.0515082240104676,32,5,0,0

1.3972672700881958,64,5,0,0

1.2550620794296266,128,5,0,0

1.2347891211509705,256,5,0,0

1.1894835472106933,512,5,0,0

1.181292676925659,1024,5,0,0

1.2221044540405273,2048,5,0,0

1.2084458231925965,4096,5,0,1

1.204325258731842,8192,5,0,6

1.1964842081069946,16384,5,0,23

1.2006375312805175,32768,5,0,95

1.198876941204071,65536,5,0,385

1.1965404629707337,131072,5,0,1569

Data Point 6– 0-dimensional case average weight data points – second run

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

2.8640124574303627,16,5,2,0

4.036417213780806,32,5,2,0

5.465721874544397,64,5,2,0

7.7440715715754775,128,5,2,0

10.648013108409941,256,5,2,0

14.783850453031482,512,5,2,0

21.13774647199316,1024,5,2,1

29.059839572268537,2048,5,2,0

41.757622344205444,4096,5,2,0

58.977279675464885,8192,5,2,0

83.32516839902055,16384,5,2,3

117.56383692590771,32768,5,2,13

166.01494043814964,65536,5,2,66

234.48801565180298,131072,5,2,364

331.6316216765754,262144,5,2,997

Data Point 7– 2-dimensional case average weight data points

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

2.535987174510956,16,5,2,0

3.7888180065900086,32,5,2,0

5.29129505767487,64,5,2,0

7.774453801102936,128,5,2,0

10.668283758754843,256,5,2,0

14.937799415015615,512,5,2,0

21.057418422435877,1024,5,2,0

29.650448589908773,2048,5,2,0

41.78933632510598,4096,5,2,0

59.02053751886406,8192,5,2,0

83.31463324092928,16384,5,2,3

117.63271960722977,32768,5,2,14

165.94013368060072,65536,5,2,72

234.60718925522488,131072,5,2,383

331.63165489965754,262144,5,2,997

Data Point 8– 2-dimensional case average weight data points – second run

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

3.9379806905984878,16,5,3,0

7.0748623128980395,32,5,3,0

11.275876444391907,64,5,3,0

17.460271099582314,128,5,3,0

27.05195173965767,256,5,3,0

43.292936361860484,512,5,3,0

68.41637505339459,1024,5,3,0

107.60926598482766,2048,5,3,0

105.28322581606918,4096,5,3,0

251.8443615792552,8192,5,3,0

421.01424381630494,16384,5,3,3

669.058238634083,32768,5,3,15

1058.4925859335926,65536,5,3,63

1677.4240568766807,131072,5,3,262

2657.8024463733864,262144,5,3,1171

Data Point 9– 3-dimensional case average weight data points

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

3.945302838087082,16,5,3,0

6.974357525259256,32,5,3,0

11.419268963485957,64,5,3,0

17.556969568505885,128,5,3,0

27.051677347905933,256,5,3,0

43.231725483387706,512,5,3,0

68.21574403573759,1024,5,3,0

107.36205035708845,2048,5,3,0

104.2279126307927,4096,5,3,0

251.4779570076149,8192,5,3,0

420.85270637853534,16384,5,3,3

667.4965836753603,32768,5,3,16

1059.5896649725328,65536,5,3,64

1677.2849300887494,131072,5,3,264

2658.4136866184244,262144,5,3,1197

Data Point 10– 3-dimensional case average weight data points – second run

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

4.520382408797741,16,5,4,0

10.226742908358574,32,5,4,0

16.439463831484318,64,5,4,0

28.939876575022936,128,5,4,0

46.5976966522634,256,5,4,0

78.19732644706964,512,5,4,0

129.66116686780006,1024,5,4,0

215.6711673144251,2048,5,4,0

361.243078167364,4096,5,4,0

603.6596801473759,8192,5,4,3

1009.6207018316724,16384,5,4,23

1556.8886403066106,32768,5,4,17

2809.5797778014094,65536,5,4,71

4740.292861791258,131072,5,4,300

7949.762444243347,262144,5,4,1551

Data Point 11– 4-dimensional case average weight data points

aveWeight,numVertices,numTrials,dimension,timeTaken in seconds

5.852017280459404,16,5,4,0

10.041743900626898,32,5,4,0

16.916150530427693,64,5,4,0

28.789752888679505,128,5,4,0

46.53302054516971,256,5,4,0

78.14827242605388,512,5,4,0

130.78476838953793,1024,5,4,0

217.0634129591286,2048,5,4,0

359.70045942869035,4096,5,4,0

603.3072808695026,8192,5,4,3

1007.5033384215087,16384,5,4,29

1554.5492929211352,32768,5,4,18

2811.3532751438443,65536,5,4,69

4740.102171613113,131072,5,4,285

7952.562185400328,262144,5,4,1363

Data Point 11– 4-dimensional case average weight data points – second run

Conclusion

It was easy to observe that the 0-dimensional case approached to 1.2 and is convergent. The higher dimensions were less obvious when picking different seed values for the pseudorandom number generator. Here is my analysis.

Expected size [edit]

The expected size of the EMST for large numbers of points has been determined by J. Michael Steele.[14] If f is the density of the probability function for picking points, then for large n and $d \neq 1$ the size of the EMST is approximately

$$c(d)n^{\frac{d-1}{d}} \int_{\mathbb{R}^d} f(x)^{\frac{d-1}{d}}\, dx$$

where $c(d)$ is a constant depending only on the dimension d. The exact value of the constants are unknown but can be estimated from empirical evidence.

Credit:

http://www-stat.wharton.upenn.edu/~steele/Publications/PDF/MSTfGwREL.pdf

According to Steele's equation and backed by my experimentation, the function for the average weight of a randomly generated graph is within a constant factor of n $^{(d-1)/d}$ where n stands for the number of vertices and d is the dimension.

As for the runtime, it is dominated by the time it takes to sort the edges. Using the path compression and union by rank heuristics covered in class, each disjoint set operation run in constant amortized time for all practical purposes, O((m+n) log *n) time for all the find/union operations. The O(nlogn) time to sort all the edges is the dominant factor and is the overall complexity.

How to run the program

You need to have Java/JDK installed in your machine. Any version above 8 will do. You then switch directory to pa1\src\edu\harvard\extension and run the following commands

- cd “pa1\src\edu\harvard\extension”
- javac *.java
- java RandMST 0 262144 5 2 – this is one example of a test