

## Project 1: Questions 4 and 5

### Reducing the Graph

The dataset consisted of a very large graph (1,965,206 vertices, 2,766,607 edges) representing road networks in California. Since the purpose of this project is to demonstrate proficiency in graph analytics, we reduced the graph to a tractable size.

Reducing the graph was necessary, as the graph exceeded the computing power that we had access to. More importantly, many of the igraph functions are single-threaded.

For example, we began by running `IGraph::Betweenness()` on the original graph. After ~12 hours, we ran the [top](#) Unix utility to examine computer resources. Overall CPU was at a normal level, however, a single CPU core was at 100%. Given the size of the road graph, we concluded that we needed to substantially reduce the graph in order to analyze it. The non-parallel nature of the graph libraries made processing the original graph impractical- even with more computer horsepower.

Minimum degree	# of vertices
1	1,965,206
2	1,965,206
3	1,644,179
4	1,644,179
5	1,439,425
6	1,439,425
7	468,149
8	468,149
9	13,941
10	13,941

11	2,094
12	2,094
13	177
14	177
15	34
16	34
17	4
18	4
19	3
20	3

The above table was calculated with the following logic:

```

verticeCounts <- matrix(nrow=20,ncol=2)
originalGraphSize <- vcount(roadGraph)
roadGraphDegree <- degree(roadGraph)
for(minimumDegree in 1:20){
  verticeCounts[minimumDegree,1] <- minimumDegree
  verticeCounts[minimumDegree,2] <- length(which(roadGraphDegree >= minimumDegree))
}
print(verticeCounts)

```

Here, we can see that there are zero nodes of degree 1 in the graph. The graph still remains quite large, all the way down to degree size  $\geq 8$  ( 468,149 nodes). Since this leaves a very large graph, we removed all nodes of degree  $\leq 8$ .

```

roadGraph <- delete_vertices(roadGraph, which(roadGraphDegree <= 8))

```

This left a graph containing 13,941, and finally enabled us proceed with analyzing the graph as requested in the assignment.

#### Question 4

Note that all functions here employ the variable “myNetwork” as defined by:



```
> vertex_attr(myNetwork)
```

```
$name
 [1] "0"      "1"      "2"      "469"    "6"      "385"    "3"      "4"      "419"    "422"    "5"
[12] "98"     "420"    "7"      "8"      "9"      "79"     "33"     "10"     "84"     "11"     "110"
[23] "12"     "13"     "108"    "95"     "14"     "94"     "15"     "16"     "77"     "17"     "18"
[34] "3254"   "19"     "20"     "23"     "21"     "22"     "24"     "25"     "26"     "27"     "28"
[45] "29"     "30"     "3255"   "31"     "3247"   "3253"   "32"     "3246"   "2203"   "34"     "35"
[56] "36"     "50"     "1199"   "37"     "35885"  "1645159" "38"     "1641586" "39"     "40"     "1641587"
[67] "41"     "1641577" "42"     "1641355" "43"     "1639779" "44"     "45"     "1542024" "46"     "1154"
[78] "1538392" "47"     "27108"  "27325"  "48"     "49"     "27343"  "184"     "185"     "52"     "53"
[89] "54"     "4152"   "223"    "225"    "4120"   "55"     "56"     "57"     "1068"   "1099"   "76"
[100] "1069"   "58"     "1072"   "1071"   "59"     "1089"   "32419"  "60"     "61"     "62"     "63"
```

Function #5: [rnorm](#)

```
> rnorm(myNetwork)
```

```
[1] 0.39162224 2.40061059 -2.06306922 -0.15905611 1.14790781 -0.65093083 -0.08876664 0.33391912 0.09497353
[10] 0.62408066
```

Function #6: [is\\_connected](#)

```
> is_connected(myNetwork)
```

```
[1] FALSE
```

Function #7: [page\\_rank](#)

```
> page_rank(myNetwork)
```

```
$vector
      0      1      2      469      6      385      3      4      419
5.277895e-07 5.314830e-07 3.625435e-07 5.180956e-07 3.696652e-07 5.241425e-07 6.431780e-07 6.364789e-07 5.011867e-07
      422      5      98      420      7      8      9      79      33
6.377150e-07 5.038252e-07 6.359429e-07 5.138379e-07 5.384355e-07 4.012292e-07 5.112603e-07 5.178641e-07 6.082749e-07
      10      84      11      110      12      13      108      95      14
6.377762e-07 6.910508e-07 4.778386e-07 6.322602e-07 6.194260e-07 6.448131e-07 6.167511e-07 4.928051e-07 7.122870e-07
      94      15      16      77      17      18      3254      19      20
5.148286e-07 2.276889e-07 3.728068e-07 5.186749e-07 6.829081e-07 5.471579e-07 6.877709e-07 6.337438e-07 7.440116e-07
      23      21      22      24      25      26      27      28      29
6.761810e-07 2.871312e-07 2.871312e-07 2.679125e-07 6.796472e-07 2.688946e-07 6.464977e-07 2.595022e-07 5.542101e-07
      30      3255      31      3247      3253      32      3246      2203      34
7.057978e-07 6.810629e-07 5.140427e-07 6.130933e-07 7.204165e-07 5.295990e-07 6.479046e-07 6.366170e-07 2.486724e-07
```

Function #8: [as.edgelist.sna](#)

```
> as.edgelist.sna(small_Network)
```

```
      [,1] [,2] [,3]
[1,]    2    1    1
[2,]    1    2    1
[3,]    3    2    1
[4,]    2    3    1
[5,]    4    3    1
[6,]    3    4    1
[7,]    5    4    1
[8,]    4    5    1
[9,]    6    5    1
[10,]    5    6    1
attr(,"n")
[1] 6
attr(,"vnames")
[1] "254220" "254229" "254236" "254238" "254246" "
```

Function #9: [gden](#)

```
> gden(small_Network)
```

```
[1] 0.3333333
      0.3333333
```

Function #10: [is\\_connected](#)

```
> is.connected(small_Network)
Node 1, Reach 6, Total 6
Node 2, Reach 6, Total 12
Node 3, Reach 6, Total 18
Node 4, Reach 6, Total 24
Node 5, Reach 6, Total 30
Node 6, Reach 6, Total 36
[1] TRUE
```

**Question 5**(“Explore other functions in the igraph package – at least 15 of them not shown in the lecture notes.”)

Note that all functions here employ the variable “roadGraph,” as defined by:

```
#Read in table
print("Reading table")
edges <- read.table('roadNet-CA.txt')

#Convert to matrix
edgeMatrix <- as.matrix(edges)

#Extract vectors from matrix
v1 <- edgeMatrix[,1]
v2 <- edgeMatrix[,2]

print("Building relations from table")
#Build relations
relations <- data.frame(from=v1,to=v2)

#Construct graph
print("Constructing graph from relations")
roadGraph <- graph.data.frame(relations,directed=TRUE)
```

Function #1: [any\\_multiple](#)

```
> any_multiple(roadGraph)
[1] FALSE
```

Function #2: [are\\_adjacent](#)

```
> roadGraphNode1 <- V(roadGraph)[0]
> roadGraphNode2 <- V(roadGraph)[1]
> are_adjacent(roadGraph, roadGraphNode1, roadGraphNode2)
[1] FALSE
```

Function #3: [articulation\\_points](#)



```
> articulation_points(roadGraph)
+ 296/13941 vertices, named, from be27da5:
[1] 5771 8123 6030 6001 6442 6488 6487 6385 6478 6460 6384 6995 8189 8197
[15] 13456 13434 25078 30009 66649 83915 115326 107733 109792 109791 135335 116085 116651 111826
[29] 114741 115586 118527 122744 123093 125702 1181851 177505 186217 226500 218986 225120 225118 225022
[43] 225073 225197 226634 226631 293178 294541 294544 294545 293763 292365 293754 294195 295866 389026
[57] 364035 408484 436142 453260 504171 504175 504185 498390 534751 573435 500859 504777 534232 506206
[71] 505902 505897 534840 534844 534874 507518 508141 509123 512565 512773 534950 554587 542737 542632
[85] 542517 542516 544200 545463 548067 548349 548866 551537 551966 629367 658786 658747 556874 560745
[99] 560917 561026 561018 561365 562527 562909 563946 564691 565440 566181 574367 617407 580482 659694
[113] 585336 583951 584802 585415 585571 586685 586684 586683 587947 587958 587959 587851 589179 596884
[127] 601015 602340 602854 603508 603897 617103 617104 632082 667717 641959 643655 643677 643135 667816
+ ... omitted several vertices
```

Function #4: [layout\\_as\\_star](#)

```
> layout_as_star(roadGraph)
      [,1]      [,2]
[1,] 0.000000e+00 0.000000e+00
[2,] 1.000000e+00 0.000000e+00
[3,] 9.999999e-01 4.507306e-04
[4,] 9.999996e-01 9.014612e-04
[5,] 9.999991e-01 1.352192e-03
[6,] 9.999984e-01 1.802922e-03
[7,] 9.999975e-01 2.253651e-03
[8,] 9.999963e-01 2.704381e-03
[9,] 9.999950e-01 3.155109e-03
[10,] 9.999935e-01 3.605837e-03
[11,] 9.999918e-01 4.056565e-03
[12,] 9.999898e-01 4.507291e-03
[13,] 9.999877e-01 4.958017e-03
[14,] 9.999854e-01 5.408741e-03
[15,] 9.999828e-01 5.859465e-03
[16,] 9.999801e-01 6.310187e-03
[17,] 9.999771e-01 6.760908e-03
[18,] 9.999740e-01 7.211628e-03
[19,] 9.999706e-01 7.662346e-03
[20,] 9.999671e-01 8.113063e-03
```

Function #5: [authority\\_score](#)

```
> authority_score(roadGraph)
$vector
      1068      136      6790      6738      6713      125      222      239      325
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16
      346      3339      457      6793      458      7154      501      516      8153
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16
      585      3484      723      727      745      768      784      21744      946
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 1.408661e-15 1.408661e-15 8.028899e-16 8.028899e-16
      988      1022      7750      1153      1176      1352      1358      1252      19504
8.028899e-16 1.408661e-15 1.408661e-15 8.028899e-16 8.028899e-16 1.408661e-15 8.028899e-16 1.408661e-15 8.028899e-16
      1368      1410      1416      1444      1538      32300      1546      1558      1822
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16
      1850      1905      1917      24236      1972      2053      2082      6729      2109
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16
      2289      2508      2414      2516      2530      2619      2663      2887      2941
8.028899e-16 1.408661e-15 8.028899e-16 1.408661e-15 8.028899e-16 1.408661e-15 1.408661e-15 8.028899e-16 8.028899e-16
      2982      3003      5854      3369      3505      3512      3520      117563      3562
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 1.408661e-15 1.408661e-15 8.028899e-16
      4208      3630      3666      3716      3788      3796      3904      6040      3912
8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 8.028899e-16 1.408661e-15 1.408661e-15 8.028899e-16
```

#### Function #6: [cocitation](#)

```
> cocitation(roadGraph)
1068 136 6790 6738 6713 125 222 239 325 346 3339 457 6793 458 7154 501 516 8153 585 3484 723 727 745 768 784
21744 946 988 1022 7750 1153 1176 1352 1358 1252 19504 1368 1410 1416 1444 1538 32300 1546 1558 1822 1850 1905
1917 24236 1972 2053 2082 6729 2109 2289 2508 2414 2516 2530 2619 2663 2887 2941 2982 3003 5854 3369 3505 3512
3520 117563 3562 4208 3630 3666 3716 3788 3796 3904 6040 3912 3961 4025 5771 5923 4069 5774 4112 4089 4090 10984
4203 4343 4487 23525 4530 5657 4710 4741 4748 5049 5082 5177 5191 5251 5259 5250 5268 5301 33728 5622 21909 5313
5367 5370 5379 5464 5476 5495 5492 5580 5587 8082 5600 8119 8123 5608 6029 5724 5727 5750 5769 5796 5824 5866
6701 5869 5912 5965 5979 6030 6004 6001 6065 6134 6042 6091 6075 6093 6112 6290 6589 6356 6482 6442 6367 6394
6372 6391 6384 6385 6487 6452 6460 6483 8126 6419 6451 6478 6480 6484 6488 6554 6500 6578 6585 8409 6595 6617
6666 6672 9150 9050 7029 6804 6816 6920 6838 7062 6844 6995 6903 6905 6918 6996 7057 15013 9546 7079 9501 33646
7137 7181 7278 7314 7478 39754 7622 7900 7920 8557 122897 8069 8185 8109 8122 8197 8150 8189 8201 8218 8235 8303
8295 8294 8380 8323 8377 8403 8615 8421 8487 8659 8660 8821 8636 8708 8721 8719 8747 8881 8843 8921 9089 9096
9092 9094 9117 9177 9295 9594 9340 9341 10017 9430 9455 9580 9503 9767 9581 9632 9688 9699 9761 9869 9871 9973
9993 10024 10075 11181 10222 10596 10337 10278 11245 10305 10607 10327 10387 10379 10538 10391 10399 12953 10527
10640 10873 10968 11057 34116 11131 11346 11337 31426 11390 11901 11917 11942 12127 12031 12140 12213 12264 12268
12530 12539 12913 13158 12589 13006 12818 12809 12927 18617 12954 13112 13066 19551 13062 13140 13318 13259 13242
13522 13309 13456 13478 13397 13404 13434 13432 13433 13494 13495 13499 13938 17917 13862 14350 14001 13970 13701
14005 14131 14129 14417 14067 14415 14073 14286 14133 14097 14108 14120 14376 14428 14485 14464 14568 14771 14766
15059 15066 15177 15309 15296 15472 15473 15569 15684 15724 15834 15971 16279 16313 29456 16523 16635 16897 21935
16855 21598 17319 16967 17097 17211 17393 17556 18330 17930 17887 17914 17921 27245 32436 17945 18521 38565 18132
```

#### Function #7: [vertex\\_connectivity](#)

```
> vertex_connectivity(roadGraph)
[1] 0
```

For functions 8 - 15, the following reduction was made to the graph in order to make these computationally expensive functions able to complete in a reasonable amount of time:

```
> roadGraphDegree <- degree(roadGraph)
> roadGraph <- delete_vertices(roadGraph, which(roadGraphDegree <= 8))
```



### Function #8: farthest\_vertices

```
> farthest_vertices(roadGraph)
$vertices
+ 2/13941 vertices, named, from 24fe843:
[1] 6480 6484

$distance
[1] 6
```

### Function #9: constraint

```
> constraint(roadGraph)
```

1068	136	6790	6738	6713	125	222	239	325	346	3339	457
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6793	458	7154	501	516	8153	585	3484	723	727	745	768
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0000000
784	21744	946	988	1022	7750	1153	1176	1352	1358	1252	19504
1.0000000	NaN	NaN	NaN	1.0000000	1.0000000	NaN	NaN	1.0000000	NaN	1.0000000	NaN
1368	1410	1416	1444	1538	32300	1546	1558	1822	1850	1905	1917
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
24236	1972	2053	2082	6729	2109	2289	2508	2414	2516	2530	2619
NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0000000	NaN	1.0000000	NaN	1.0000000
2663	2887	2941	2982	3003	5854	3369	3505	3512	3520	117563	3562
1.0000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0000000	1.0000000	NaN
4208	3630	3666	3716	3788	3796	3904	6040	3912	3961	4025	5771
NaN	NaN	NaN	NaN	NaN	NaN	1.0000000	1.0000000	NaN	NaN	1.0000000	0.5000000
5923	4069	5774	4112	4089	4090	10984	4203	4343	4487	23525	4530
NaN	NaN	NaN	NaN	1.0000000	1.0000000	NaN	NaN	NaN	NaN	NaN	NaN
5657	4710	4741	4748	5049	5082	5177	5191	5251	5259	5250	5268
NaN	NaN	1.0000000	1.0000000	NaN	NaN	NaN	NaN	1.0000000	NaN	1.0000000	NaN

Function #10: count triangles

[illegible]

### Function #11: `count_motifs`

```
> count_motifs(roadGraph)
[1] 398
```

### Function #12: coreness





```

> mst(roadGraph)
IGRAPH fded007 DN-- 13941 1527 --
+ attr: name (v/c)
+ edges from fded007 (vertex names):
[1] 768 ->784 1022 ->7750 1352 ->1252 2508 ->2516 2619 ->2663 3520 ->117563 3904 ->6040 4025 ->5771
[9] 5771 ->5769 4089 ->4090 4741 ->4748 5251 ->5250 5476 ->5492 5600 ->8119 5600 ->8123 8123 ->8126
[17] 5608 ->6029 5724 ->5727 5796 ->5824 5866 ->6701 5979 ->6030 6030 ->6042 6001 ->6065 6001 ->6134
[25] 6356 ->6442 6442 ->6451 6367 ->6394 6384 ->6385 6384 ->6452 6385 ->6460 6385 ->6483 6385 ->6487
[33] 6487 ->6488 6460 ->6478 6478 ->6480 6488 ->6484 6666 ->6672 6816 ->6920 6844 ->6995 6995 ->6996
[41] 6903 ->6905 7920 ->8557 8109 ->8122 8197 ->8150 8197 ->8189 8189 ->8201 8295 ->8294 8323 ->8377
[49] 8403 ->8615 8708 ->8721 9096 ->9117 9092 ->9094 9295 ->9594 9430 ->9455 9503 ->9767 10391 ->10399
[57] 13318 ->13456 13242 ->13522 13456 ->13478 13434 ->13432 13434 ->13433 13494 ->13495 14129 ->14133 14417 ->14067
+ ... omitted several edges

```

Function #15: [strength](#)

```

> strength(roadGraph)
1068 136 6790 6738 6713 125 222 239 325 346 3339 457 6793 458 7154
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
501 516 8153 585 3484 723 727 745 768 784 21744 946 988 1022 7750
0 0 0 0 0 0 0 0 2 2 0 0 0 2 2
1153 1176 1352 1358 1252 19504 1368 1410 1416 1444 1538 32300 1546 1558 1822
0 0 2 0 2 0 0 0 0 0 0 0 0 0 0
1850 1905 1917 24236 1972 2053 2082 6729 2109 2289 2508 2414 2516 2530 2619
0 0 0 0 0 0 0 0 0 0 2 0 2 0 2
2663 2887 2941 2982 3003 5854 3369 3505 3512 3520 117563 3562 4208 3630 3666
2 0 0 0 0 0 0 0 0 2 2 0 0 0 0
3716 3788 3796 3904 6040 3912 3961 4025 5771 5923 4069 5774 4112 4089 4090
0 0 0 2 2 0 0 2 4 0 0 0 0 2 2
10984 4203 4343 4487 23525 4530 5657 4710 4741 4748 5049 5082 5177 5191 5251
0 0 0 0 0 0 0 0 2 2 0 0 0 0 2
5259 5250 5268 5301 33728 5622 21909 5313 5367 5370 5379 5464 5476 5495 5492
0 2 0 0 0 0 0 0 0 0 0 0 2 0 2
5580 5587 8082 5600 8119 8123 5608 6029 5724 5727 5750 5769 5796 5824 5866
0 0 0 4 4 6 2 2 2 2 0 2 2 2 2
6701 5869 5912 5965 5979 6030 6004 6001 6065 6134 6042 6091 6075 6093 6112

```