Efficiency in an Algorithm for Finding Cliques

Leanne

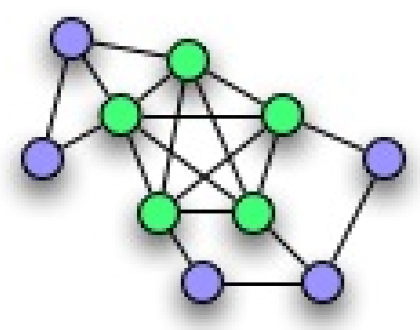
Department of Mathematics,
University of Massachusetts Dartmouth
Lsilvia2@umassd.edu

Abstract

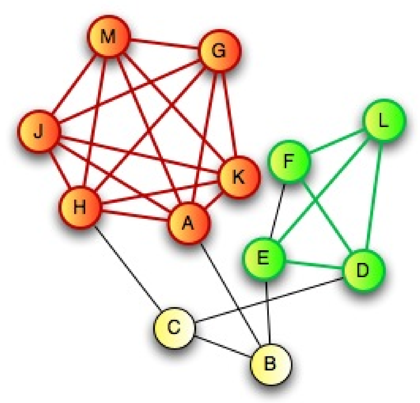
In an undirected graph, a clique is a subset of vertices such that every two vertices are connected by an edge; that is to say the clique itself is a complete graph found within the graph. The clique problem is classified as NP-Complete. As the starting graph increases in number of vertices and edges it becomes very difficult to find the answer, the maximum clique, in polynomial time.

What is a Clique?

A clique is a set of vertices in a graph, such that every vertex is connected to every other vertex in the graph. In other words, it is a (sub)complete graph.
In the below picture, the green points form a 5-clique.



A maximal clique is not a subset of any other clique. It is the maximum sized clique for those vertices.



Both Orange and Green are Maximal Cliques, while Orange is the Maximum Clique

The maximum clique is a maximal clique that has maximum cardinality; it is the clique with the largest number of vertices.

Protein structure prediction can be viewed as a problem of finding cliques in a graph whose vertices represent positions of molecules in the protein. By seeing a protein-protein interaction network as cliques, or clusters of proteins that interact with each other and don't interact as frequently outside their cluster.

Methods

Generating a Graph

This code generates a graph on two constraints which allows me to specify the number of points (vertices) the graph will contain and a maximum distance that if two points are within such distance are joined by an edge. The code randomly places the points before applying the connections.

```
t2 = SessionTime[];
r = .1;
numberofpoints = 100;
points = {{RandomReal[], RandomReal[]}};
n = 0;
lines = {};
v = {1};
Do[
 points = Append[points, {RandomReal[], RandomReal[]}];
 n++;
 v = Append[v, n];
 Do[
  f[i, n] = If[Norm[points[[i]] - points[[-1]]] < r, True, False],
  {i, 1, Length[points] - 1}];
 lines =
  Union[lines,
   DeleteCases[
    Table[If[Norm[points[[i]] - points[[-1]]] < r,
      Line[{points[[i]], points[[-1]]}]],
     {i, 1, Length[points] - 1}], Null]],
 {k, 1, numberofpoints}]
G = MakeGraph[v, f, Type -> Undirected];
Graphics[{{PointSize[0.02], Red, Point[points]},
  {PointSize[Large], Blue, lines}}]
end = SessionTime[] - t2
```

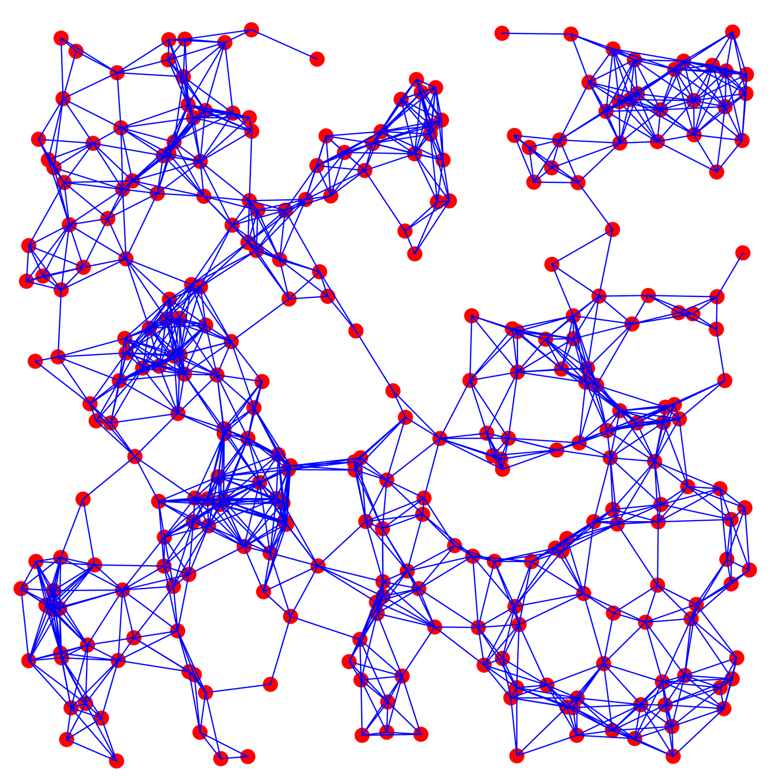
Finding the Maximum Cliques

The basis of my code, which is written in Mathematica, is manipulating and creating lists. I discovered Mathematica has its own command to find the maximum clique, MaximumClique[G]. However, I have started testing the speed of my code against that of mathematica's command and for the graphs that I have tested, my code runs significantly faster.

The code requires two parts of information from the graph being analyzed: a list of the degrees, and a list of the edges, ordered by the vertex.

```
start = SessionTime[];
DList = Degrees[G];
EList = EdgeList[G];
MyList = {};
Do[{H = EList[[1 ;; DList[[1]]]],
  EList = Complement[EList, H];,
  A = Flatten[H],
  B = Complement[A, {0}],
  DList = Delete[DList, 1],
  MyList = Append[MyList, B]}, {k, 1, Length[Degrees[G]]}]
Thing = Complement[Degrees[G], {0}] ;
Last[Thing];
Do[CliquesOf[k] = {}, {k, 1, Last[Thing] + 1}]
CliquesOf[2] = EdgeList[G];
Do[{Inter = MyList[[k3]],
  Stuff = {k3},
  A = Complement[Inter, Stuff],
  Do[{B = A[[k]],
    Inter2 = Intersection[Inter, MyList[[B]]],
    Etc = Sort[Flatten[{Stuff, B}]],
    x = Length[Etc],
    C1 = Complement[Inter2, Etc],
    If[Inter2 == Etc, {},
     {Do[{y = x + 1, CliquesOf[y] = Append[CliquesOf[y],
          Sort[Flatten[{Etc, C1[[k2]]}]]]}, {k2, 1, Length[C1]}]}]
   }, {k, 1, Length[A]}]}, {k3, 1, Length[Degrees[G]]}]
CliquesOf[3] = Complement[CliquesOf[3], {}];
Do[{If [CliquesOf[k4] == {}, {MaxCliq = CliquesOf[k4 - 1][[1]], Break[]},
   {Do[{Letters = CliquesOf[k4][[k]],
      Inter = Intersection[MyList[[Letters[[1]]]], MyList[[Letters[[2]]]]],
      Do[Inter = Intersection[Inter, MyList[[Letters[[k3]]]]],
       {k3, 3, Length[Letters]}],
      C2 = Complement[Inter, Letters],
      If[Inter == Letters, {},
       Do[{CliquesOf[k4 + 1] = Append[CliquesOf[k4 + 1],
          Sort[Flatten[{Letters, C2[[k2]]}]]],
         CliquesOf[k4 + 1] = Complement[CliquesOf[k4 + 1], {}]},
        {k2, 1, Length[C2]}]]}, {k, 1, Length[CliquesOf[k4]]}]}]},
 {k4, 3, Last[Thing] + 1}]
end = SessionTime[] - start
MaxCliq
```

Data



Random Graph of 300 points,connecting if within a distance of .1

For graphs that don't include high amounts of vertices with outrageous degrees, my code seems to run in polynomial time.

My Method vs. Mathematica's MaximumClique[G] command

My Code	Mathematica	Size of Max. Clique
0.05242	2.795514	3
0.052284	2.913246	3
0.052315	4.556635	3
0.059131	2.911586	3
0.056061	4.33452	4
0.061157	4.956026	4

Table 1: *100 points, .1 distance*

My code has yet to reach one second and Mathematica's has increased to over half a minute.

My Code	Mathematica	Size of Max. Clique
0.158542	27.006818	3
0.150284	33.837406	3
0.162106	27.534414	3
0.168909	47.2927	4
0.175829	35.271799	4
0.210005	42.222799	4

Table 2: *200 points, .1 distance*

Although my code has doubled in time, it has increased by a minuscule amount, while Mathematica's command is taking over four minutes.

My Code	Mathematica	Size of Max. Clique
0.382765	261.293623	4
0.367425	375.67082	4
0.384376	262.535695	4

Table 3: *300 points, .1 distance*

In the following tables, Mathematica's code becomes very inefficient to return the Maximum Clique.

My Code	Mathematica	Size of Max. Clique
0.806121	1810 unfinished	4
0.693635	no attempt	4
0.71146	1250 unfinished	4

Table 4: *400 points, .1 distance*

My Code	Mathematica	Size of Max. Clique
1.342606	no attempt	4
1.236563	no attempt	4
1.253054	no attempt	4

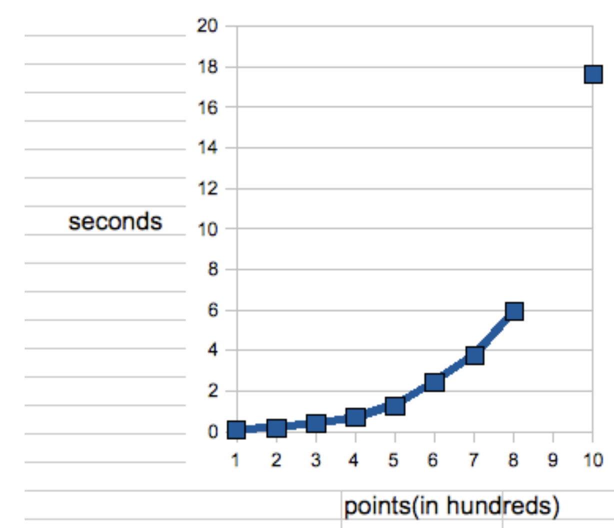
Table 5: *500 points, .1 distance*

My Code	Mathematica	Size of Max. Clique
1.99273	no attempt	4
2.262895	no attempt	4
2.623419	no attempt	4
2.314154	no attempt	5

Table 6: *600 points, .1 distance*

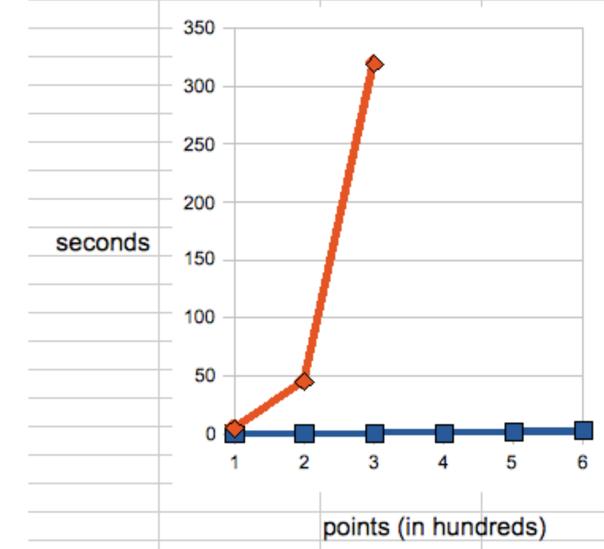
My Code	Mathematica	Size of Max. Clique
11.205336	no attempt	5
14.895239	no attempt	5
18.801664	no attempt	5
18.891935	no attempt	5
22.461103	no attempt	5 Degree Range: 10-49

Table 7: *1000 points, .1 distance*



This is a plot of the average time stamps for graphs ranging from 100-1000 points

When comparing my time stamps against that of Mathematica's command, mine appears to appear almost linear and non-increasing. This shows how much more dramatically Mathematica's command takes for the graphs that i tested.



When the graphs I tested had all degrees ranging very high, such as in the below table, neither code would produce an answer in a reasonable amount of time.

My Code	Mathematica	Size of Clique	Degree Range
1589.13 unfinished	no attempt	not found	24-91

Table 8: *2000 points, .1 distance*

My Code	Mathematica	Size of Clique	Degree Range
21.99	706.13 unfinished	4	3-32

Table 9: *2000 points, .05 distance*

For a graph of 3,000 points, containing degrees from 0-14, my code took only 11.68 seconds to find the maximum clique, 3. When the degree range is increased to a range of 5-42 it took approximately 112 seconds, finding a 4-clique. Mathematica's command would take an undetermined amount of time.

For 50 points with degrees ranging from 11-38 the code took 20 secs,a slim difference with mathematica's taking 24. Both codes found the same 10-clique.

My code seems to run fastest when the degrees stay reasonable. For 4,000 points, degrees of 0-6 took 5.37 seconds, mathematica's command was still unfinished at 2234. On a graph of 100 points, with degrees 4-25, .8 seconds (mathematica: 14.34). Degrees 12-46 took 114 seconds (mathematica: 191).

Future Objectives

Refining my code, targeting more specific graphs that correlate to finding cliques. Creating a code to find all maximal cliques.

SIAM AN10, Society for Industrial and Applied Mathematics Annual Meeting, 12 - 16 July 2010, Pittsburgh, Pennsylvania