Ramsey Numbers: Improving the Bounds of R(5,5)

Curtis J. Kunkel
Department of Mathematics
University of Minnesota, Morris
Morris, MN 56267, kunk0003@mrs.umn.edu

Faculty Advisor: Prof. Peh Ng
Department of Mathematics
University of Minnesota, Morris
Morris, MN 56267, pehng@mrs.umn.edu

Abstract:

Ramsey number R(m,n) = r is the smallest integer r such that a graph of r vertices has either a complete subgraph (clique) of size m or its complement has a complete subgraph of size n. Currently, the exact value of R(5,5) is unknown, however the best known lower and upper bounds are 43 <= R(5,5) <= 49. In this paper we will discuss a method that we use to construct a better lower bound, namely, by way of genetic programming (GP)/ genetic algorithm (GA). This method involves standard genetic algorithms mutations and crossovers as recombination techniques as well as using algorithms and heuristics to find maximum clique of a graph. We implemented this method on a particular genetic algorithm software called Sutherland. Results will include the “best” graphs found using this technique over multiple runs, statistical data as to the likelihood of increasing the current best-known lower bound of R(5,5) if not strictly the lower bound.

1. Introduction to Problem

Our project deals with the concept of Ramsey Numbers. The Ramsey Number Problem is also referred to as the Party Problem or a form of the Maximum Clique Problem. The basic idea of a Ramsey Number can be described using the following example:

Example 1:

A Ramsey Number, R(m,n) = r, is the smallest number of people r at a party in which either everyone in a group of people of size m know everyone else in that group or everyone in a group of people of size n does not know anyone else in that group.

When we think of this in a little more mathematical way, it can be defined by the following:

Definition 1:

A Ramsey Number, R(m,n) = r, is the smallest size of a graph r such that every graph of that size has either a clique of size m or an independent set of size n.

An equivalent definition of a Ramsey Number is also given:

Definition 2:

A Ramsey Number, R(m,n) = r, is the smallest size of a graph r such that every graph of that size has either a clique of size m in the graph or a clique of size n in its complement.

Our main goal of this project has been the optimization of r = R(5,5), i.e. finding the smallest integer r such that all graphs, G, of size r include a clique of size 5 in either G or the complement of G. Currently, R(5,5) is between 43 and 49 inclusive. We are going to focus on the current lower bound, 43.

In the following sections of this paper, the equivalence of both definitions of a Ramsey Number that were previously mentioned will be explored more fully. A general overview of Ramsey Theory will also be explored, along with a look at how to theoretically solve a Ramsey Number problem. After this theory is explored, the implementation of what was actually created will be described along with the methods used. The final sections of the paper contain results and their analysis, along with some conclusions that can be drawn from the results.

2. Mathematical Foundation

This section will give a brief description of the main graph theory concepts used in Ramsey Numbers. We will discuss a few theories related to Ramsey Numbers and in particular, R(5,5).

Graph Theory Foundation

It is clear from definition 1 that we need to define a graph, a clique, and an independent set before we can fully understand the definition of a Ramsey Number. Therefore, these definitions are given:

Definition 3:

A graph, denoted G = (V,E), consists of a set of vertices, V, and a set of edges, E, where the edges are lines that connect pairs of vertices. Figure 1 shows examples of graphs.

Definition 4:

A clique of a graph is a subset of vertices such that there exists an edge between all possible pairs of vertices in that subset of vertices. Figure 1 (b) is an example of a clique of size 3 in a graph (the clique is in bold).

Definition 5:

An independent set of a graph is a subset of vertices such that there exists no edges between any pair of vertices in that subset of vertices. Figure 1 (c) is an example of an independent set of size 3 of the graph in Figure 1 (b).

Definition 6:

A complete graph is a graph such that all pairs of vertices have an edge that connects both vertices. Complete graphs are notated K_i, where i is the number of vertices. Figure 1 (a) is an example of the complete graph K_5.

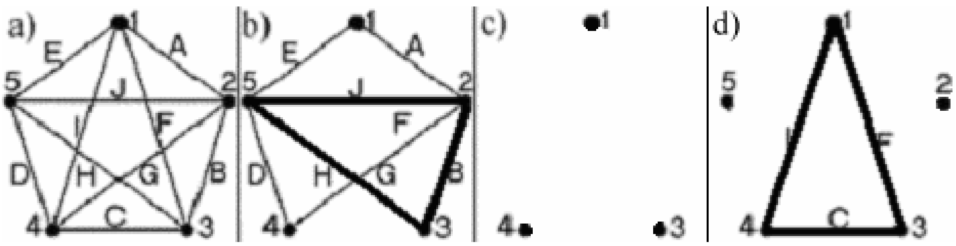
Definition 7:

The complement of graph G = (V, E), denoted $\bar{G} = (V, \bar{E})$, is a graph with the same set of vertices V such that $\bar{E} = \{(u,v) : u,v \in V \text{ and } (u,v) \notin E\}$. For instance, figure 1 (b) and Figure 1 (d) is an example of a graph and its complement.

Result of Definition 7:

Given G and its complement $\overline{G}$, then $G \approx \overline{\overline{G}}$.

Figure 1: Graphs



Now that we have some basic structures of graph theory defined, we can proceed to show the equivalence of definition 1 and definition 2. A Ramsey Number, R(*m*,*n*) = *r*, is the smallest size of a graph G such that every graph of that size has either a complete subgraph of size *m* in its graph or a complete subgraph of size *n* in a graph consisting of its deleted edges.

The relationship between a graph G and its complement $\overline{G}$ as it relates to cliques and independent sets is as follows: if a subgraph forms a clique in G, then the vertices in this subgraph induces an independent set in the complement of G, $\overline{G}$. This makes our case, R(5,5), a bit easier to look at. R(5,5) is therefore defined as the smallest size of a graph G such that there exists a clique of size 5 in its graph or its complement.

Ramsey Theory Foundation

The old joke that is often associated with Ramsey Numbers is that if an alien spaceship were to come to Earth and demand that we tell them the answer to R(5,5) or they will kill us all, it would take all the computing power in the world to find the answer for the aliens. If he asks for R(6,6), we should try to kill them. This shows how difficult this problem is. Many people have been working on R(5,5) for decades and have made some remarkable headway. The following few theorems gives only a small insight into some of their hard work. Endös and Szekeres are given credit for proving theorem 1. [4]

Theorem 1: $R(m,n) \le \binom{m+n-2}{m-1}$

Proof:

We proceed by induction on *k*, where *k* = *m* + *n*. There is equality when *m* = 1 and *m* = 2, and for every value of *n*. It is also true for *n* = 1 and *n* = 2, independent of the value of *m*.

Therefore, the result is true for all values of k with $2 \leq k \leq 5$ and we proceed through the remainder of the proof assuming that $m \geq 3$ and $n \geq 3$.

Assume that R(s,t) exists for all positive integers s and t with $s + t < k$ where $k \geq 6$, and that

$$R(s,t) \leq \binom{s+t-2}{s-1}$$

Let m and n be integers such that $m + n = k$, where $m \geq 3$ and $n \geq 3$. By the inductive hypothesis, the Ramsey numbers R(m-1,n) and R(m,n-1) exist and further,

$$R(m\text{-}1,n) \leq \binom{m+n-3}{m-2} \quad \text{and } R(m,n\text{-}1) \leq \binom{m+n-3}{m-1}$$

Since

$$\binom{m+n-3}{m-2} + \binom{m+n-3}{m-1} = \binom{m+n-2}{m-1}$$

,it follows that $R(m\text{-}1,n) + R(m,n\text{-}1) \leq \binom{m+n-2}{m-1}$ (*)

Now let p = R(m-1,n) + R(m,n-1) and suppose that each edge of K_p is arbitrarily colored red or blue, analogous to exists or does not exist. We show that there is either a red K_m or a blue K_n. Let v be a vertex of K_p. Then the degree of $v = p - 1$ = R(m-1,n) + R(m,n-1) – 1. We consider two cases:

Case 1. Assume that v is incident with at least R(m-1,n) red edges. Let S denote the set of vertices of K_p that are joined to v by red edges. Thus $|S| \geq$ R(m-1,n) and (S) is a complete graph of order at least R(m-1,n) whose edges are colored red or blue. Therefore, (S) contains either a red K_{m-1} or a blue K_n. If (S) contains a blue K_n, so does K_p. Suppose that (S) contains a red K_{m-1}. Then (S U {v}) contains a red K_m. Hence, in this case, K_p contains either a red K_m or a blue K_n.

Case 2. Assume that v is incident with at most R(m-1,n) –1 red edges. Then v is incident with at least R(m,n-1) blue edges. Let T denote the set of vertices of K_p that are joined to v by blue edges. Therefore, $|T| \geq$ R(m,n-1) and (T) is a complete graph of order at least R(m,n-1) whose edges are colored red or blue. Hence, (T) contains either a red K_m or a blue K_{n-1}. If (T) contains a red K_m, then K_p does as well. Suppose that (T) contains a blue K_{n-1}. Then (T U {v}) contains a blue K_n. In this case as well, then K_p contains a red K_m or a blue K_n.

This shows that R(m,n) ≤ R(m-1,n) + R(m,n-1), which when combined with (*), gives the desired result. □

For our case, this theorem yields the result that R(5,5) ≤ 70. This of course is far more than the current upper bound, which is 49. The corollary to theorem 1, however gives us a little closer approximation to the current upper bound. [4]

Corollary to Theorem 1: R(m,n) ≤ R(m-1,n) + R(m,n-1); if R(m-1,n) and R(m,n-1) are both even, then the inequality is strict.

Result: R(m,n) = R(n,m)

The previous corollary and result used together yield a new upper bound to R(5,5) as long as we know R(4,5), which we know is 25. [7] Therefore, our new upper bound to R(5,5) ≤ 50. Stanislaw Radziszowski and Brendan McKay in their 1992 publication *A New Upper Bound For The Ramsey Number R(5,5)* discuss the method in which they moved the upper bound from 50 to 49. [5] Our goal of this project is to improve the lower bound of R(5,5) from 43 to 44.

3. Approach to Finding Lower Bounds

The lower bound of R(5,5) has been improved by the process of presenting counter examples (i.e. a graph of size *r* such that no clique or independent set of size 5 exist in that graph). The most recent transition of the lower bound (from 42 to 43) came about through the use of Genetic Algorithms to search some of the graphs of size 42 for a counter example. They were successful in doing so and thus, the lower bound was shifted to 43. [7] This is the same approach we used in our project. There are many different ways in which to go about this, all of which come down to finding a counter example that disproves the fact that 43 is the lower bound.

Enumeration of all Possibilities

The first and most obvious approach is to enumerate all possible graphs of size 43 and check if all of these graphs have a clique or independent set of size 5. Before beginning to write out all possible graphs of size 43, however, lets first look at how many graphs exist.

There are 903 possible edges in a graph with 43 vertices and each possible edge has two possibilities, either an edge exists or it doesn't. Therefore there are 2^{903} different possible graphs. This works out to be about 6.76×10^{271}, which are far too many graphs to enumerate. Therefore this is not a feasible method.

Genetic Algorithms

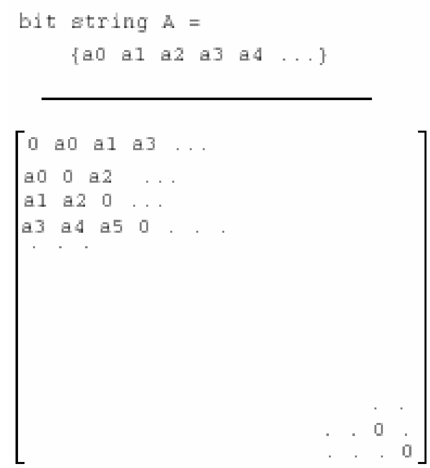
The method we are going to use when it comes to searching for a counter example is Genetic Algorithms (GA). Genetic algorithms are best described as a better than random, generational search algorithm. It stems from Evolutionary Computation (EC) and its process is elaborated upon in terms of R(5,5) in the following explanation.

Step 1

The first step is to select a random sample of individuals from the population. In this case, the population is all the different graphs of size 43 (approximately 6.76×10^{271}). From this population, a sample of 250 individuals was selected. Each individual represents a graph in the following manner.

Since bit strings are the key data structure used in genetic algorithms, each graph is represented by a string of 0s and 1s (0 represents false and 1 represents true). In this case, a 1 represents the existence of an edge, whereas a 0 represents the lack of an edge. The bit string is parsed into an incidence matrix as follows:

Figure 2: Parsing of Bit Strings to Graphs



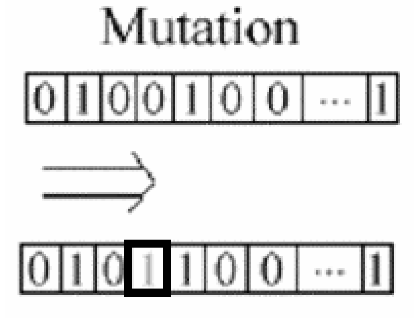
Step 2

Now that we have 250 individuals, we must conduct tournaments to see which graphs are better than others. In order to conduct the tournaments, we must have a way of ranking or ordering all of the individuals. This is called the fitness function. In our case, the fitness function is the number of cliques of size 5 in the graph added to the number of cliques of size 5 in the graphs complement. The possible fitness values for this fitness function range from 0 (hopefully) to 962598. A portion of the java code for this fitness function is given in Appendix A.

Given that each individual has a fitness value, we can now hold tournaments to determine what subset of the individuals in our sample are the better ones. In our case, smaller numbers are better since we are looking for a graph whose fitness function yields a 0. The better individuals are now granted permission to produce offspring through point mutation and/or standard crossover. These are relatively standard recombination techniques.

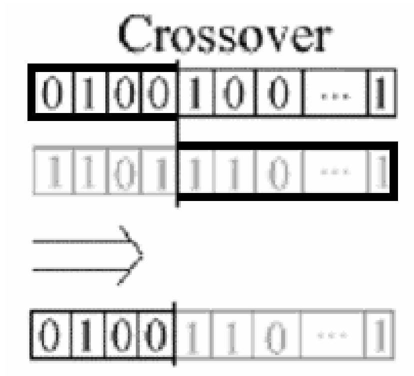
Point mutation involves the random switching of bits in the bit string. This equates to the removal or addition of an edge to a graph, thus creating a new graph. Figure 3 shows an example of point mutation.

Figure 3: Mutation Recombination Technique



Standard crossover involves taking two individuals and merging them together. This merge takes place at a random point in both individuals, where the new graph consists of everything prior to the chosen point from the first individual and everything after the chosen point from the second individual. An example of this is given below:

Figure 4: Crossover Recombination Technique



Step 3

Now that new offspring are formed, they replace the poorer individuals from the tournaments. This is the end of one generation and the beginning of the next.

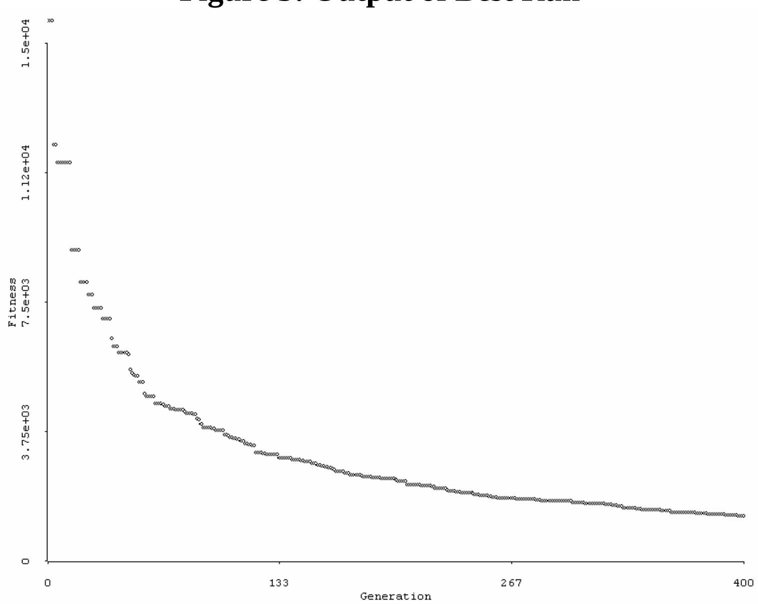
Step 2 and 3 are repeated for as long as is chosen by the user. In our case, 400 generations was chosen because after about 400 generations the variance in the population got so small that there was hardly any progress being made.

For more information regarding Evolutionary Computation and Genetic Algorithms see *Genetic Programming~ An Introduction* by Banzhaf, Nordin, Keller, and Francone. [1]

4. Results and Analysis

Over a period of 2 weeks, the genetic algorithm method described above was allowed to run on 15 different machines in the Computer Science Teaching Lab at the University of Minnesota, Morris (Sci 2610). After the two-week period, there were 230 complete runs where data could be extracted and analysis of the run could take place. Of these 230 different runs, each run taking about 3 days to complete, 10 had a best fitness between 1300 and 1400 while the others were larger than 1400 but smaller than 2000. The best fitness value of a graph was 1300 and was taken from the 13th run on the machine labeled Cochrane. The following is a graph of the fitness of that run with generation on the x-axis.

Figure 5: Output of Best Run



All runs have a curve similar to this one, and that is why it is uncertain whether or not extending this graph to an infinite number of generations will improve the fitness to 0 or if it will plateau asymptotically at a larger value. The best graph we found is given by the chromosome below (parsing via the method mentioned previously (Figure 2) will yield the appropriate incidence matrix of the graph):

Figure 6: Chromosome of Best Individual

01100111000001110010000111011000000000101001100100101111101110101001111111101000110010000100110011111100101011
11100000101001110110111010010101100110011000110110100000000110000101101000111110011101000101010110011100 10001
1100001111010001010101111001000001011101111011010110100000011000110001101001110110110111001011001101011110100
0111110110100011000101000101001010111101011000100011001011111011011000010110000101001001010101000101110110111
1100111011000010001000111110111010010101011110101011100110100011110001101110100110110010010110011000000000100
110010111111100010010001011010110011010101001110010100111001001011100100100100100010110011110000100101010111
1101011010000001111001111101100011111001010101000010011001110100100011100101011000011100010101110011101111000
1010001100010101001001111001011110110101000011000100101010110001000101011101010101111010011100001101101 01001
000010011110111001100101111011100001010

5. Conclusion

It is clear that this project did not improve the lower bound of R(5,5). However, the genetic algorithm method does yield interesting results and therefore might in fact produce a counter example. The method we implemented could be improved by changing the number of generations, the population size, the recombination techniques, or any other of many available ‘knobs’ that can be tweaked. This method does not guarantee the existence of a counter example, but instead only allows us a different way of looking at the problem. With other techniques and advanced technology, the future is looking bright for finding a solution to this problem, or at the very least improving the lower bound of R(5,5) to 44.

References

[1] Banzhaf, Nordin, Keller, Francone, *Genetic Programming~ An Introduction*, Morgan Kaufmann Publishers, Inc., San Francisco, CA (1998).

[2] Bomze, Budinich, Pardalos, Pelillo, *The Maximum Clique Problem*, (1999).

[3] Bondy & Murty,*Graph Theory With Applications*, The Macmillan Press Ltd, Great Britain (1976).

[4] Graham, Rothschild, Spencer, *Ramsey Theory*, John Wiley & Sons, New York, NY (1990).

[5] McKay, Brendan D., Radziszowski, Stanislaw P., *A New Upper Bound For The Ramsey Number R(5,5)*, (1992).

[6] Micron PC, http://support.buympc.com/index.html, 2003.

[7] Radziszowski, Stanislaw P., *Small Ramsey Numbers (Revision #9)*, (2002).

[8] Weisstein. *Eric Weisstein’s World of Mathematics: A Wolfram Web Resource*, http://mathworld.wolfram.com/ (2002).

[9] West, Douglas B., *Introduction to Graph Theory*, Prentice-Hall, Inc., Upper Saddle River, NJ (1996).

Appendix A: Number of Cliques of Size 5 Algorithm

```
/**
 * This method is an exact algorithm that returns the number of
 * cliques of size 5 in this graph.
 **/
public int numCliquesSize5(Graph g) {
int[] temp = new int[size];

for(int i = 0 ; i < g.getSize() ; i++) {
    temp[i] = g.getDegree(i);
}

Graph tempG = g;

int[] temp2 = new int[0];

for(int i = 0 ; i < g.getSize() ; i++) {
    if(temp[i] >= 4) {
       temp2 = add(temp2,i);
    } else {
       tempG = remove(tempG,i);
    }
}

int out = 0;

boolean[] intersect = new boolean[tempG.getSize()];
intersect = initAll(intersect);
boolean[] intersect1 = new boolean[tempG.getSize()];
boolean[] intersect2 = new boolean[tempG.getSize()];
boolean[] intersect3 = new boolean[tempG.getSize()];
boolean[] intersect4 = new boolean[tempG.getSize()];
boolean[] intersect5 = new boolean[tempG.getSize()];

for(int x1 = 0 ; x1 < (temp2.length - 4) ; x1++) {
   intersect1 = intersection(intersect, connectedList(tempG, temp2[x1]));

   for(int x2 = (x1 + 1) ; x2 < (temp2.length - 3) ; x2++) {
   intersect2 = intersection(intersect1, connectedList(tempG, temp2[x2]));
      for(int x3 = (x2 + 1) ; x3 < (temp2.length - 2) ; x3++) {
      intersect3 = intersection(intersect2, connectedList(tempG, temp2[x3]));
         for(int x4 = (x3 + 1) ; x4 < (temp2.length - 1) ; x4++) {
         intersect4 = intersection(intersect3, connectedList(tempG, temp2[x4]));
            for(int x5 = (x4 + 1) ; x5 < temp2.length ; x5++) {
            intersect5 = intersection(intersect4, connectedList(tempG, temp2[x5]));

              if(numTrue(intersect5) > 4) {
                 out++;
                  }

               }
            }
        }
    }
}

return out;
}
```

Appendix B: Computer Specifications [6]

Category	Description	QTY	Part Number
Processor	Intel Pentium IIII 1.6GHz 400FSB 256K 478Pin	1	CPU002049-01
Memory	***512MB 64X64 133MHZ SDRAM	1	MOD001966-00
Hard Drive	Seagate U6 20GB IDE 5400RPM	1	HDI001793-00
CD-ROM Drive	Lite-On 52X IDE CD-ROM LTN526	1	CDI001263-01
Video Card	Visiontek Vanta 16MB Video Card	1	VCD001472-01
Misc I/O	Microsoft 104-Key Elite Natural Keyboard	1	KBR001062-03
Misc I/O	Microsoft Intellimouse Optical USB and PS/2 Compatible	1	MOU001072-01
Case	Odyssey Client Pro PA700 front foot	1	CSE001587-00
Case	Odyssey ATX PA700 MiniTower w/o Power Supply	1	CSE001638-00
Power Supply	Delta 300W Power Supply	1	PWS001108-00
Operating System	Microsoft Windows 98 SE Recovery Media Kit	1	OSS001318-00
Software	Havre - Driver CD	1	SFD001106-02
Software	Microsoft Intellimouse Driver 3.5 in. Floppy (English)	1	SFO001446-07
Software	Microsoft.Internet Explorer 5.5	1	SFO002447-00
Warranty	ASSY DESKTOP THREE-YEAR LIMITED PARTS WARRANTY AND TECHNICAL SUPPORT POLICY	1	WAR001049-00
Case	Odyssey PA700 ATX MT Bezel Assembly	1	BZL001057-03
Motherboard	Havre - Intel 845 Brookdale Motherboard (Millennia® Max XR and ClientPro® CR)	1	MBD001151-03
Software	Nvidia Video Driver CD	1	MED001417-05