Programming Assignment 2

Wedding Seating Arrangement

1>Task 1: SAT Encoding

1. Encoding the constraint that a person can sit in only one place.
   Consider a person A and there exists 3 tables; 1, 2, and 3, then the CNF to encode the above constraint would be
   (A1 ∨ A2 ∨A3) ∧ (¬A1 ∨ ¬A2) ∧ (¬A1 ∨ ¬A3) ∧(¬A2 ∨ ¬A3)

   Extending this to "M" tables and "N" number of people we get.

   $$\bigwedge_{i\in M}\left(\bigvee_{j\in N} x_{i,j} \wedge \bigwedge_{j\in N}\left(\bar{x}_{i,j} \bigvee_{k\in N_j} \bar{x}_{ik}\right)\right)$$

2. Encoding the constraint that if the relationship between given two people is friends then they always sit in the same table.

   Consider 2 people A and B, and consider there exists 3 tables, then the following CNF encodes A and B are friends
   (¬A1 v B1) ∧ (¬A2 v B2) ∧ (¬A3 v B3)

   Extending this to "M" tables and "N" number of people we get. *Where <i,j>εF(i,j) i.e, for F friend relationships*

   $$\bigwedge_{\substack{<i,j>\in M \\ j>i}} \bigwedge_{k\in N} \left(\bar{x}_{ij} \bigvee x_{jk}\right)$$

3. Encoding the constraint that if the relationship between two people is that of enemy then they never sit in the same table.

   Consider 2 people A and B, and consider there exists

3 tables; 1, 2, and 3. Then the following CNF encodes A and B are enemies

(¬A1 v ¬B1) ∧ (¬A2 v ¬B2) ∧ (¬A3 v ¬B3)

Extending this to "M" tables and "N" number of people we get. Where ***E(i,j)*** *i.e, for E enemy relationships*

$$\bigwedge_{\substack{<i,j>\in M \\ j>i}} \bigwedge_{k\in N} \left(\bar{X}_{ij} \vee \bar{x}_{jk}\right)$$

The number of clauses in total to encode if there "M" people and "N" tables, out of which "F" number of friends and "E" number of enemies exist the would be:

1. For the first constraint for M people we get
   $M + (M * {}^{N}C_{2}) = M(1+ (N(N-1)/2))$ clauses.
2. For the second constraint there are F friends
   For a relation the number of clauses are N, then for F relations the number of clauses will be F*N.
3. For the 3rd constraint there are E enemies
   For a relation the number of clauses are N, then for E relations then number of clauses are N*E.
   So the total would be
   M(1 + N*((N+1)/2)) + F*N + E*N clauses.

2>Task 2: Instance Generator

An Instance generator has been programmed in C, it's a header file named CNF.h in the file attached.

3>Task 3: Experiment 2

1. PL-resolution and WalkSAT: These have been programmed in C and the source code for the same are under the names PL-Resolution.c and walk_SAT.c is present in the attached file.

4>Task 4: The graph of PL-Resolution vs WalkSat is shown below

The list PL shows the percentage of satisfiable sentences among 50 sentences for each of the probability enemy relationship. The domain ranges from 0.02 to 0.2 probability.

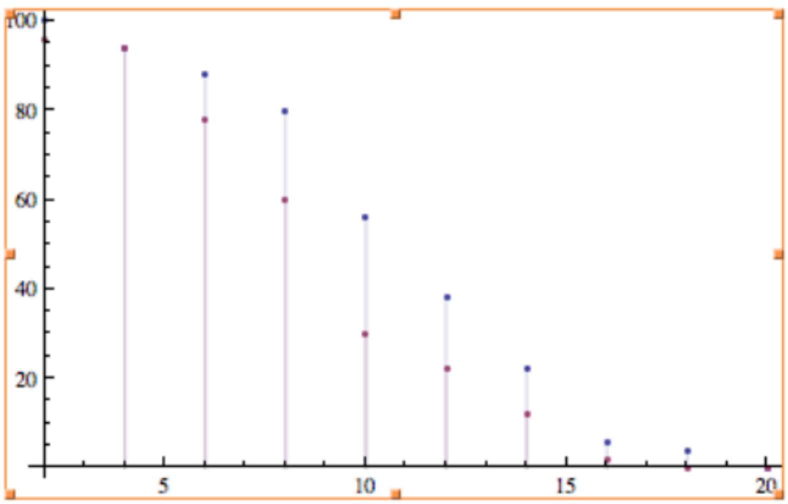
The list WS shows the percentage of satisfiable sentences obtained through the WalkSAT algorithm, among 50 sentences for each of the probability enemy relationship. The domain ranges from 0.02 to 0.2 probabilities.

The dots in blue represent PL –resolution and the dots in red represent WalkSAT.

```
In[48]:= pl = {100, 94, 88, 80, 56, 38, 22, 6, 4, 0}
Out[48]= {100, 94, 88, 80, 56, 38, 22, 6, 4, 0}

In[51]:= ws = {96, 94, 78, 60, 30, 22, 12, 2, 0, 0}
Out[51]= {96, 94, 78, 60, 30, 22, 12, 2, 0, 0}

In[52]:= DiscretePlot[{pl[[i/2]], ws[[i/2]]}, {i, 2, 20, 2}]
```



It can be clearly seen from the graph that these are not equal. The reason is that a PL-resolution though time consuming, by the end of its run, is clearly able to point out to the satisfiable from the non-satisfiable. This is not the case with walksat, it depends on the max_number of flips allowed, and if the sentence is not decidable within the maximum number of flips then the algorithm ends without making a decision.

5>Task5:
Increasing the maximum number of flips with the enemy probability remaining the same and the probability of friends increasing at an interval of 2% from 2 to 20% we get the below graph.
Where green points to 1000 flips,
Pink points to 500 flips and blue points to the 100 flips.

From the graph if we neglect aberrations due to randomness, it can be seen that the more the number of flips the greater the probability of the sentence decided as satisfiable or not. And in this case it is seen that we have greater number of satisfiable sentences.

Wolfram Mathematica | FOR STUDENTS Demonstrations | Ma

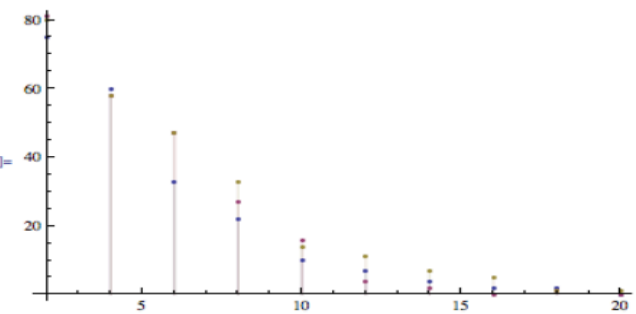
```
In[56]:= ws100 = {75, 60, 33, 22, 10, 7, 4, 2, 2, 0}
Out[56]= {75, 60, 33, 22, 10, 7, 4, 2, 2, 0}

In[57]:= ws500 = {81, 58, 47, 27, 16, 4, 2, 0, 1, 0}
Out[57]= {81, 58, 47, 27, 16, 4, 2, 0, 1, 0}

In[62]:= ws1000 = {80, 58, 47, 33, 14, 11, 7, 5, 1, 1}
Out[62]= {80, 58, 47, 33, 14, 11, 7, 5, 1, 1}

In[63]:= DiscretePlot[{ws100[[i/2]], ws500[[i/2]], ws1000[[i/2]]}, {i, 2, 20, 2}]
```

Out[63]=



1. As <f> increases the number of satisfiable sentences Decreases, this is because of the constraint F puts on the sentence, it expects if 2 people are friends they need to sit in the same table, these constraints would tend to contradict the previously satisfiable sentences, which turned out to be true without the above constraint.

2. Yes, I do have confidence on those sentences that have been displayed as satisfiable. But, sentences that have ***not*** been marked as satisfiable could turn out to be satisfiable at greater value of max_flips as it is evidently seen from the graph.

6>Task6:

A. For 16ppl and 2 tables with <f> 0.02 and <e> 0.02
   average clause /symbol = 1.246875.
   average iterations = 16.049999.

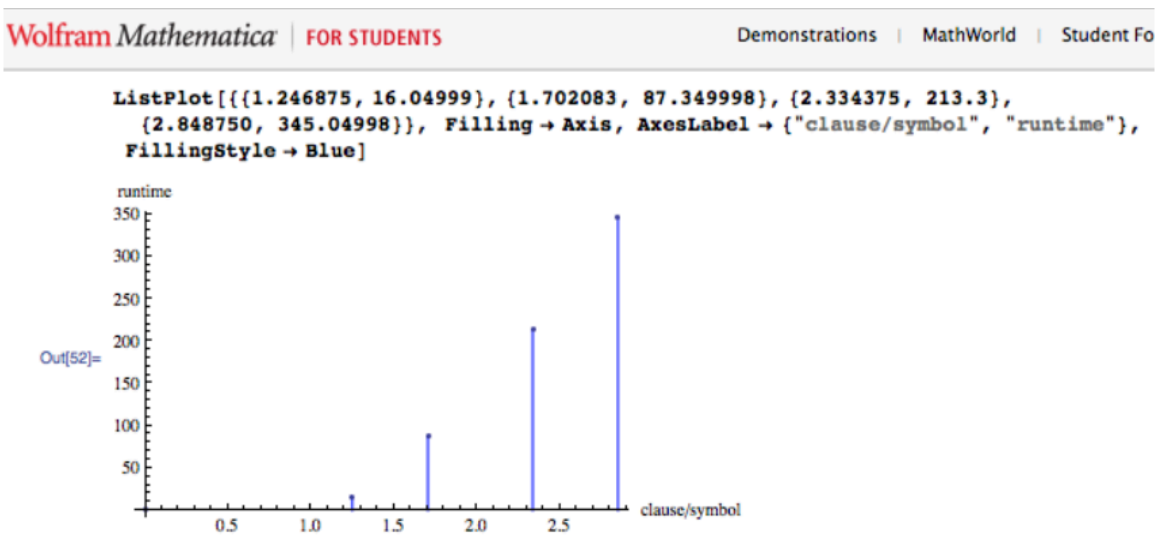
B. For 24ppl and 3 tables with <f> 0.02 and <e> 0.02
   average clause /symbol = 1.702083.
   average iterations = 87.349998.

C. For 32 ppl and 4 tables with <f> 0.02 and <e> 0.02
average clause /symbol = 2.334375.
average iterations = 213.300003.

D. For 40ppl and 5 tables with <f> 0.02 and <e> 0.02
average clause /symbol = 2.848750.
average iterations = 345.049988.

E. For 48ppl and 6 tables with <f> 0.02 and <e> 0.02
average clause /symbol = 3.451042.
average iterations = 549.200012.

Below is the graph for run time verses clause/symbol ratio for each of the above <M,N> for 20 satisfiable sentences of each.



1. The Clause to symbol ratio of the satisfied clauses do match with the previously derived equations. This could be tested through the CNF generator and matched. And this would verify the same.

2.The graph plotted is similar and hence consistent with the one seen on the AIMA, it is consistent with the runtime being around the same as shown in the graph in the AIMA for similar clause/symbol ratio.