Ryland Atkins
10/7/2016
Dr. Allen
DIY SAT Solvers

Algorithms

WalkSat

WalkSat uses the idea of hill climbing in combination with a random relocation to avoid getting stuck on local maxima. To implement it I used an array to separately hold the current model and clauses. The model is initialized to a random assignment of variables and the clauses are read from a given file.The pseudo code is as follows:

```
WalkSat(model, clauses, p, maxFlips)
    For i to maxFlips
        If model satisfies clauses
            Return model
        Clause = random clause from clauses
        With probability p
            Flip random value
        Else
            Flip symbol that maximizes satisfied clauses
```

GeneticSat

GeneticSat took a different route. I initialized several instances of the model and set of clauses. First, I checked if any model satisfied its set of clauses. If not, then for each model I calculated a *fitness*, then placed all instances in a priority queue based on that fitness. Then pairwise, starting with the two most-fit instances, I crossbred them and mutated them based on a probability p. The pseudo code is as follows:

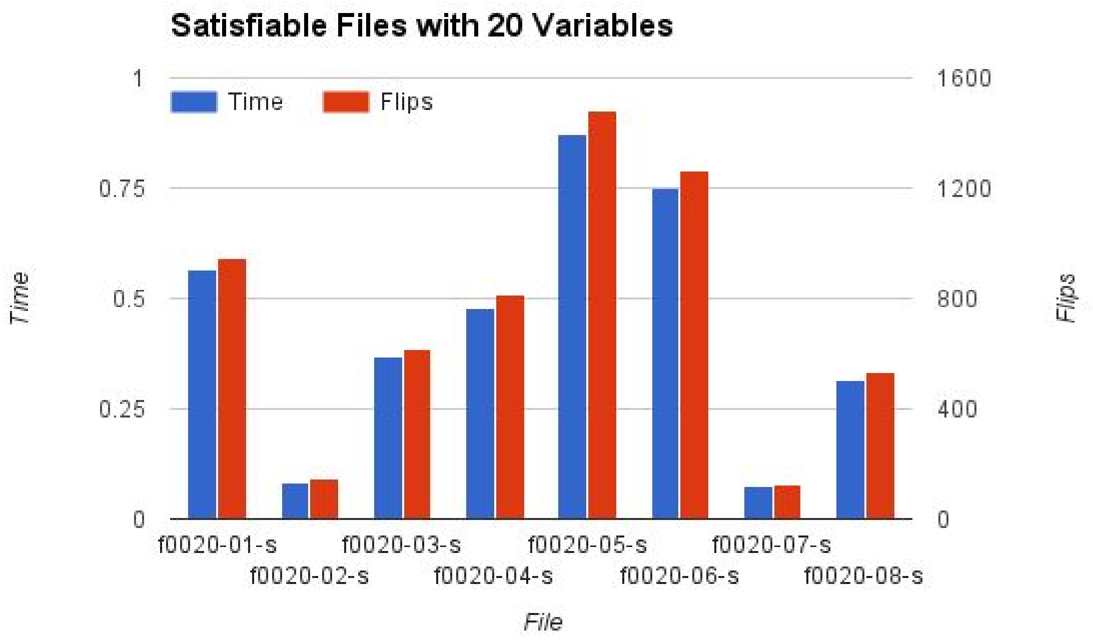
```
GeneticSat(model, clauses, p, maxFlips)
    Initialize population
    For i to maxFlips
        Que each member based on fitness
        For each pair from best to worst
            Cut a selection and cross stitch
        Check for viability of member models
            Return model
        Mutate
    Return Failure
```

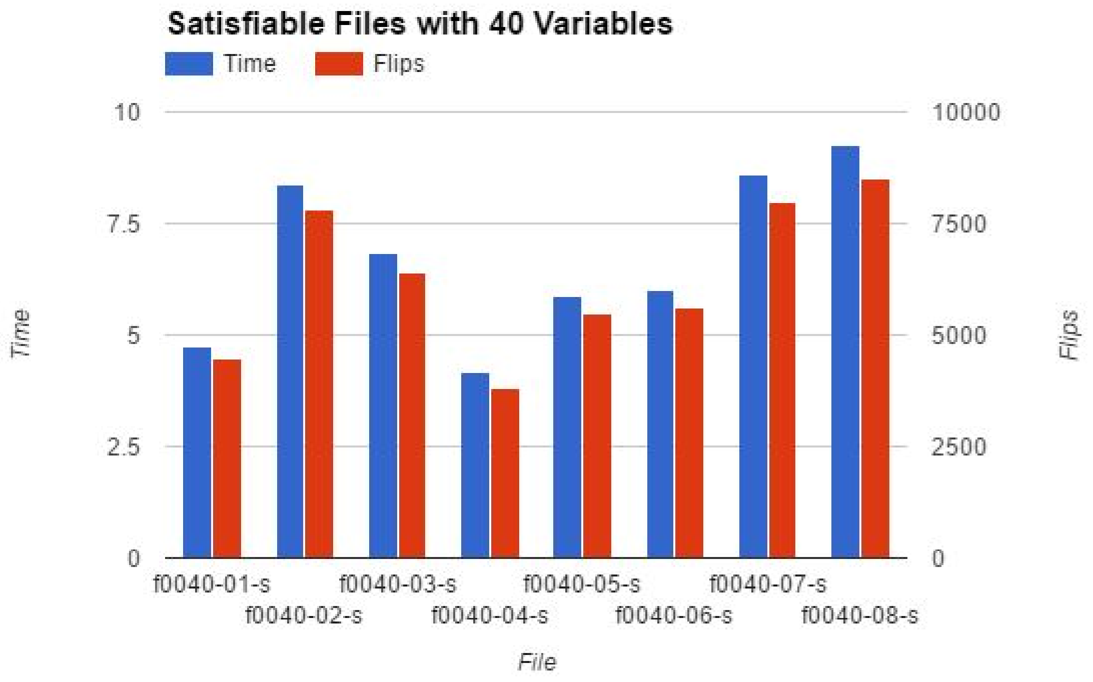
Data

For both WalkSat and GeneticSat, since they both use randomness to determine their starting model, I ran each file ten times and averaged the time and number of flips used.

WalkSat

For WalkSat I set the maximum number of flips to 10,000. This gave the program enough time to find correct values when testing with 40 variables, but kept the amount of time to determine that a problem is unsatisfiable relatively low.

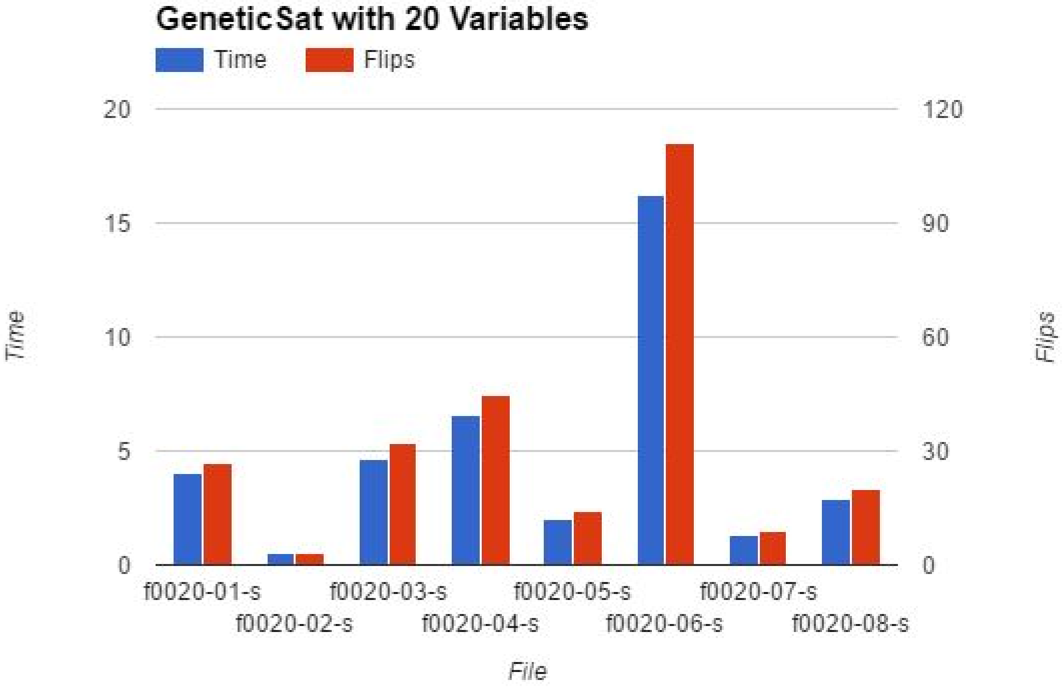

Satisfiable Files with 20 Variables


Satisfiable Files with 40 Variables

Unsatisfiable		
# of Variables	Time	Flips
20	5.869610494	10000
40	10.81551023	10000

GeneticSat

For GeneticSat I used a maximum of 120 flips with 100 models. Such a low number of flips is possible because of the high number of models. By using more random models, the probability of generating a correct model at random increases. Below is a graph illustrating the average run-time and number of flips for the 20 variable files. After excessive amounts of run-time (>2 hours), GeneticSat was never able to find a solution to the 40 variable problems. After checking the 20 variable solutions, I know that it generates correct solutions, but I believe the time in this algorithm is bounded by a quickly growing function (n^2, 2^n, $n!$).



Unsatisfiable	
Time	Flips
17.20818722	120

Conclusions

WalkSat is certainly superior to GeneticSat for large numbers of variables, however GeneticSat has potential when the number of variables stays low. Given the small amount of flips used in GeneticSat as compared to WalkSat, it can be theorized that with the right optimization of models and probability, GeneticSat could be faster than WalkSat.