Lab1.md 2024-10-04

Metaheuristics Lab1

Ricardo Rosa

A62461

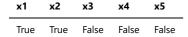
Problem 4:

Run your algorithm on the hoos.cnf instance, and list all solutions that satisfy the problem instance.

The hoos.cnf problem instance consists of:

 $\mathsf{F} = (\neg \mathsf{x} 1 \, \lor \, \mathsf{x} 2) \, \land \, (\neg \mathsf{x} 2 \, \lor \, \mathsf{x} 1) \, \land \, (\neg \mathsf{x} 1 \, \lor \, \neg \mathsf{x} 2 \, \lor \, \neg \mathsf{x} 3) \, \land \, (\mathsf{x} 1 \, \lor \, \mathsf{x} 2) \, \land \, (\neg \mathsf{x} 4 \, \lor \, \mathsf{x} 3) \, \land \, (\neg \mathsf{x} 5 \, \lor \, \mathsf{x} 3)$

And running the instance through the algorithm gives only one possible solution:



Problem 5:

Run your algorithm on the uf20-01.cnf instance. How many solutions satisfy the problem instance?

The uf20-01.cnf instance consists of an instance with 20 variables and 91 clauses

After running the instance through the algorithm, it calculated 8 solutions that satisfy the problem, and these were:

| x1 | x2 | х3 | х4 | х5 | х6 | х7 | х8 | х9 | x10 | х11х | 12 | x13 | x14 | x15 | x16 | х17 | x18 | x19 | x20 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|------|------|-------|------|-------|-------|------|
| False | True | True | True | False | False | False | True | True | True | True | False | False | True | True | False | True | True | True | True |
| True | False | False | False | False | True | False | False | False | False | False | False | True | True | True | False | True | False | False | True |
| True | False | False | False | False | True | False | False | True | False | False | False | False | True | True | False | True | False | False | True |
| True | False | False | False | False | True | False | False | True | False | False | False | True | True | True | False | True | False | False | True |
| True | False | False | True | False | False | False | False | False | True | False | False | True | True | True | False | True | False | False | True |
| True | False | False | True | False | False | False | True | False | True | False | False | True | True | True | False | True | False | False | True |
| True | False | False | True | False | True | False | False | False | False | False | False | True | True | True | False | True | False | False | True |
| True | False | False | True | False | True | False | False | False | True | False | False | True | True | True | False | True | False | False | True |

Problem 6:

It is impractical to do the same for a larger instance, say one contained in uf100-430. How could you estimate the time needed by your computer program to complete the task on such an instance? Justify your answer.

With a larger instance such as uf100-430, which has 100 variables and 430 clauses, it would be inpractical, as the time complexity for this problem would be:

\$\$ O(2^n) \$\$

Where n represents the number of variables

For this problem, to estimate the runtime of the program, we're going to assume *T* as the total number of operations divided by the computer's speed in operations per second.

Assuming the computer can make one billion operations per second (which would correspond to 1GHz), then the total of operations would be:

 $T = {430 * 2^{100} \setminus 0^9} = 5.46 * 10^{23}$ seconds

This corresponds to roughly \$1.73 * 10^{16}\$ years, or 1.73 quadrillion years.

So as we can observe, brute forcing our way to the possible solutions for this problem is not a suitable option for larger datasets, as the time it takes to run the program is exponential to the number of variables.