CSE-4111: Artificial Intelligence Lab
Assignment Three Report

Showkot Hossain
Roll: SH - 042

February 27, 2019

1 Introduction

In this assignment, I designed an arc consistency problem and also solved the problem using **AC-1, AC-2, AC-3, AC-4** algorithms. Finally I measured their performance and plotted them in xy graph. I used Pyhton as programming language to design and solve the problem. At first I generated the graph for variable number of nodes using numpy (creating a random matrix) and then set the domains (domain size is also variable) and distributed the total 4 constraints randomly among the edges of the graph. I implemented **AC-1, AC-2, AC-3, AC-4** algorithms and used them to check whether the graph satisfies the constraints. Finally I plotted their performance using matplotlib.

2 Tools

Python,numpy,matplotlib,visual studio code as IDE

3 Performance Evaluation

I have measured their performance on different angles.

Time vs Number of Nodes: I gradually increased the number of nodes to 10,20,30,40,60,80,100,120,160, and measured how much time is taken by the four algorithms for different number of nodes.

Time vs Domain Size: I fixed the number of nodes to 120. But this time I gradually increased the size of each domain of the graph and measured the time taken by the algorithms.

Time vs Density: I fixed the number of nodes to 120. This time I gradually increase the density of the graph and measured the runtime of four algorithms.

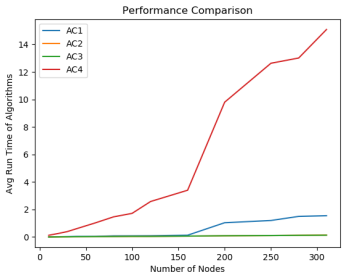

Figure 1: Runtime vs Number of Nodes

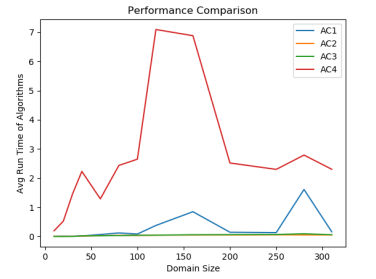

Figure 2: Runtime vs Domain Size

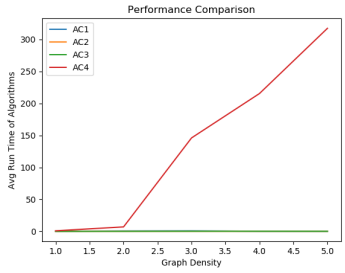

Figure 3: Runtime vs Density

4 Findings

4.1 Problem Solving and Performance

I have found some interesting properties of the algorithms. Here it goes. Firstly, when I increase the number of nodes or the density of the graph, the graph is more likely have no solution. Morever, increasing the number of constraints will also cause no solution. AC4 is always taking significantly more time than any other algorithms whatever the input graph is! Then comes the AC1 as it is a naive approach. AC2 and AC3 are taking almost equal avg time although it depends on the input graph.The type of the constraints is another important factor. At first I used difficult constraints and the graph was hardly having solution. Then I relaxed the constraints to have the solution. When plottig the graph it is difficult to compare AC2 and AC3.

4.2 Statistical Significance Test

I have performed a statistical significance test **Anova Tukey HSD Test** for the different run times of the algorithms. The test shows that run time of AC-4 is significantly high from the other three algorithm's run time since Tukey HSD p-value ¡ 0.01 for AC-4. According to the test AC-1, AC-2, AC-3 are insignificantly different which indicates that their runtimes are close to each other. We can easily verify these information looking at the graph provided.

4.3 Difficulties

It was not that easy to design and solve the problem. At first I used 10 constraints for the graph and my graph was densed because of the random graph generator. As I result my algorithms always failed to find the solution. Then I reduced the constraints to 4 and also made the graph sparse. According to my previously submitted design, my constraints were too difficult to satisfy. So I had to relax the constraints finally. Implementing AC2 and AC4 was difficult because these algorithms were not in book and was hard to find on the web. I worked hard and gave a lot of time to solve the whole problem.