Lab 8 report

Fernando De Santiago

University of Texas at El Paso

Introduction

For this lab I had 2 requirements to meet which was create a randomization and backtracking algorithm. The first with randomization I had to check if trig identities were equal to one another in all combinations possible and for the backtracking I had to check a set could be cut into two subsets and equal the same value and if they did to display the two sets.

# Randomization

For this method I edited a model that was given and changed the range to -π to π. I also changed the importing of math into a different library called mpmath in order to use secant. Then I put the trig identities into a list of strings. I then created a method called discover and it contained 2 for loops and 1 if in order to create the method. The for loop started at the beginning and the second for loop was to get each other trig identity starting at the first one in the first for loop. I then used an if to check if the trig identities were equal and if they were then it would print them out.

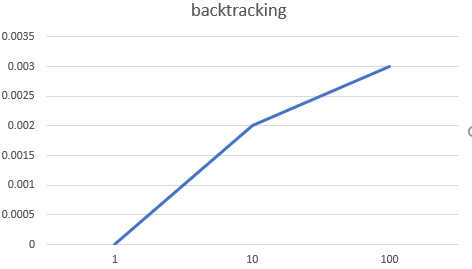
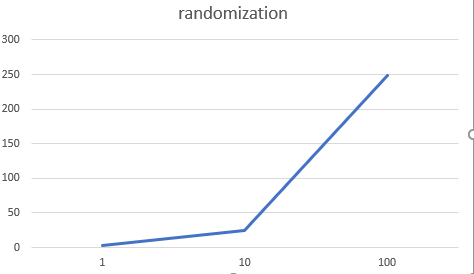
**Backtracking**

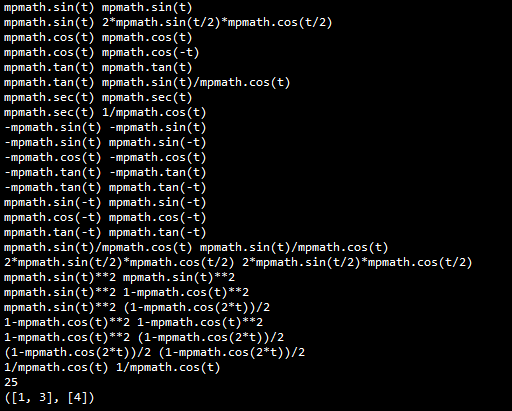
For this method I used one of the methods given to us in a model and left it alone. I then created a second method called partition that would take in two sets. This method would first check if the sum of the set was divisible by two with no remained and if it wasn’t that meant that no partition existed, else if would use the method given to us to get the first set. It would then check if the length of the set that was given was equal to zero and if it was then it would say no partition existed, else it would go into a for loop that would compare the values and if they equaled then if would pop those values from the set passed to it and would return the two sets.

### Conclusion

I learned in this lab how to use randomization in order to solve a problem and how it worked. I also learned how to use backtracking in order to check if sets can be cut into subsets and see if they can equal each other and if unionized equal to the original set.

**Runtimes**





**Appendix**

**# -\*- coding: utf-8 -**

**#CS2302**

**#Fernando De Santiago**

**#LAB8**

**#Olac Fuentes, Anindita Nath and Maliheh Zargaran**

**#last edited 5/9/19 22:01:00 PM**

**#Section M/W 10:30-11:50**

**#purpose of lab is to get a hand of randomized algorithms and backtracking.**

**"""**

**Created on Thu May 9 09:44:26 2019**

**@author: Fernando**

**"""**

**import random**

**import numpy as np**

**import mpmath**

**def equal(f1, f2,tries=1000,tolerance=0.0001):**

**for i in range(tries):**

**t = random.uniform(-mpmath.pi,mpmath.pi)**

**y1 = eval(f1)**

**y2 = eval(f2)**

**if np.abs(y1-y2)>tolerance:**

**return False**

**return True**

**def subsetsum(S,last,goal):**

**if goal ==0:**

**return True, []**

**if goal<0 or last<0:**

**return False, []**

**res, subset = subsetsum(S,last-1,goal-S[last]) # Take S[last]**

**if res:**

**subset.append(S[last])**

**return True, subset**

**else:**

**return subsetsum(S,last-1,goal) # Don't take S[last]**

**def partition(S,S2):**

**if sum(S)%2!=0:#if summation of sum is odd then return error message**

**return "No partition exists"**

**else:**

**res,s,=subsetsum(S,len(S)-1,sum(S)//2)#gets one set**

**if len(s)==0:#checks if length of set is 0 if it is return error message**

**return "No partition exists"**

**for i in s:#checks every number in set s**

**counter=0#used to get position**

**for j in S:#checks all numbers in big set S**

**if i == j:#if i and j are the same then pop it out of big S**

**S.pop(counter)**

**counter+=1**

**return s,S**

**def discover(S):**

**count=0**

**for i in range(len(S)):#goes through all the strings**

**for j in range(i,len(S)):#goes through all the string starting at i**

**if(equal(S[i],S[j])):#uses equal method to check if similar**

**print(S[i],S[j])**

**count+=1**

**print(count)**

**S=['mpmath.sin(t)','mpmath.cos(t)','mpmath.tan(t)','mpmath.sec(t)','-mpmath.sin(t)','-mpmath.cos(t)','-mpmath.tan(t)',**

**'mpmath.sin(-t)','mpmath.cos(-t)','mpmath.tan(-t)','mpmath.sin(t)/mpmath.cos(t)','2\*mpmath.sin(t/2)\*mpmath.cos(t/2)',**

**'mpmath.sin(t)\*\*2','1-mpmath.cos(t)\*\*2','(1-mpmath.cos(2\*t))/2','1/mpmath.cos(t)']**

**#**

**Set2=[1,4,3]**

**discover(S)**

**print(partition(Set2,Set2))**

**“I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class.” Fernando De Santiago**