MIE-PAA
Paul Rondeau
rondepau
December 9, 2014

Homework 4

In this assignment, we were tasked to use an advanced iterative method to solve the 0/1 knapsack problem. The advanced iterative method used was a Simulated Annealing algorithm.

This algorithm is a heuristic designed to be an accurate and efficient method to finding solving many problems. There are several parameters that can be modified to change the accuracy and efficiency of the algorithm. The algorithm takes two common algorithms and combines them in a way that yields great results. The random walk algorithm and the first improvement algorithm. The first is where a random outcome is chosen each step. The first improvement algorithm looks at all possible neighbours and takes the best outcome.

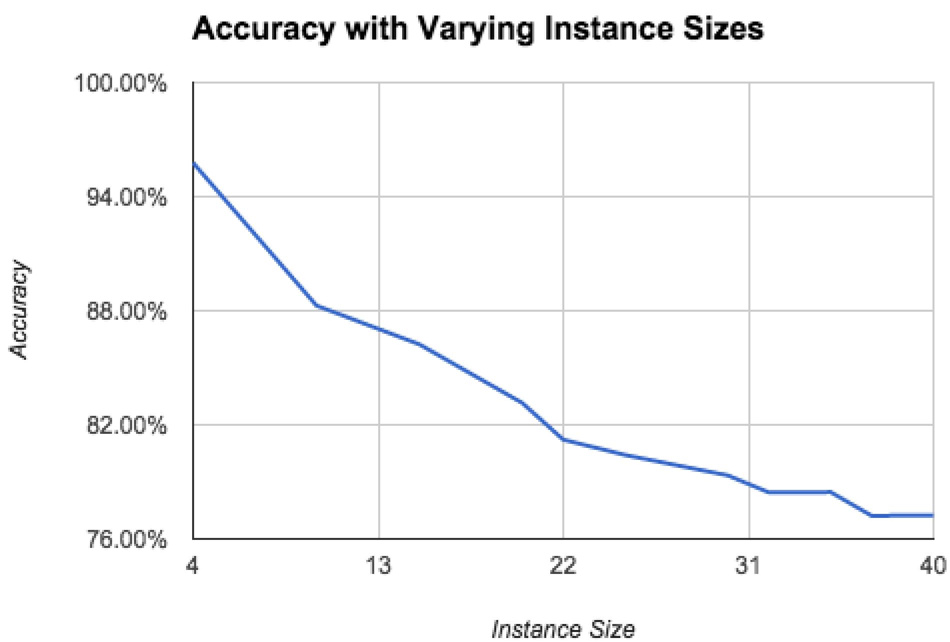
The specific algorithm is outline below:

```
#Set initial params
INITIAL_TEMP = 500
FINAL_TEMP = 100
ALPHA = 0.9
INNER_LOOP_STEPS = 1000

#used to keep track of best state found
highestCost = 0
t = INITIAL_TEMP
        state = INITIAL_STATE
        while ( t > FINAL_TEMP ) {
                for (i = 0; i < INNER_LOOP_STEPS; i++) {
                        # create a new state that is a neighbour to the current state
                        new = random_neighbour(state)
                        # repair the state if it isn't valid anymore
                        new = repair(new)

                        #find the new cost
                        curCost = cost(new)
                        if newCost > highestCost: highestCost = newCost
                        delta_C = newCost - cost(state)
```

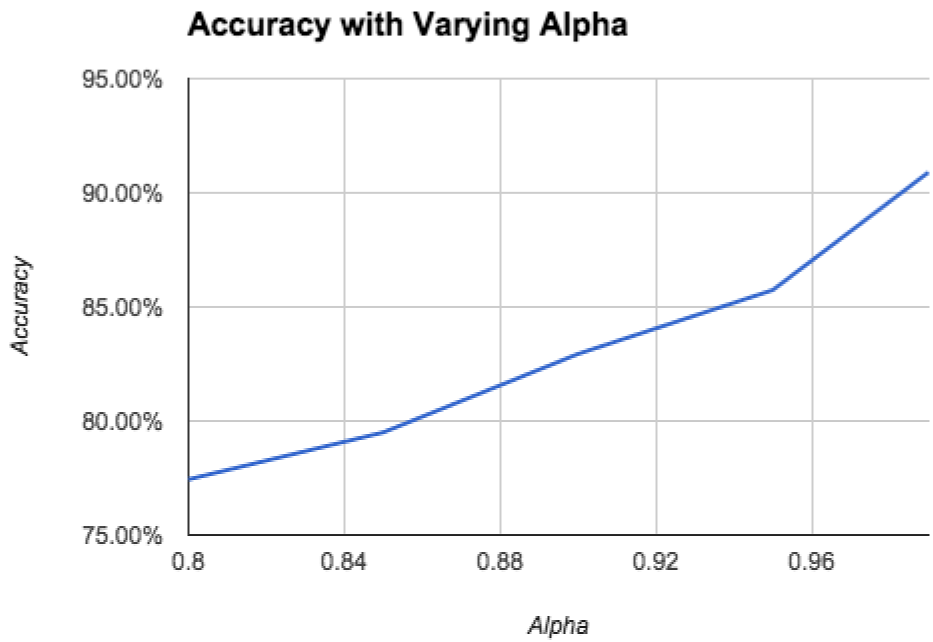
```
                    #check if the new state is better than the old one
                    if ( delta_C > 0 ):
                              state = new
                    #if not, check if it's within threshold
                    elif (accept(delta_C, t)):
                              state = new
          }
          #adjust the temperature with a geometric algorithm
          t = cool(t, ALPHA)
}
```

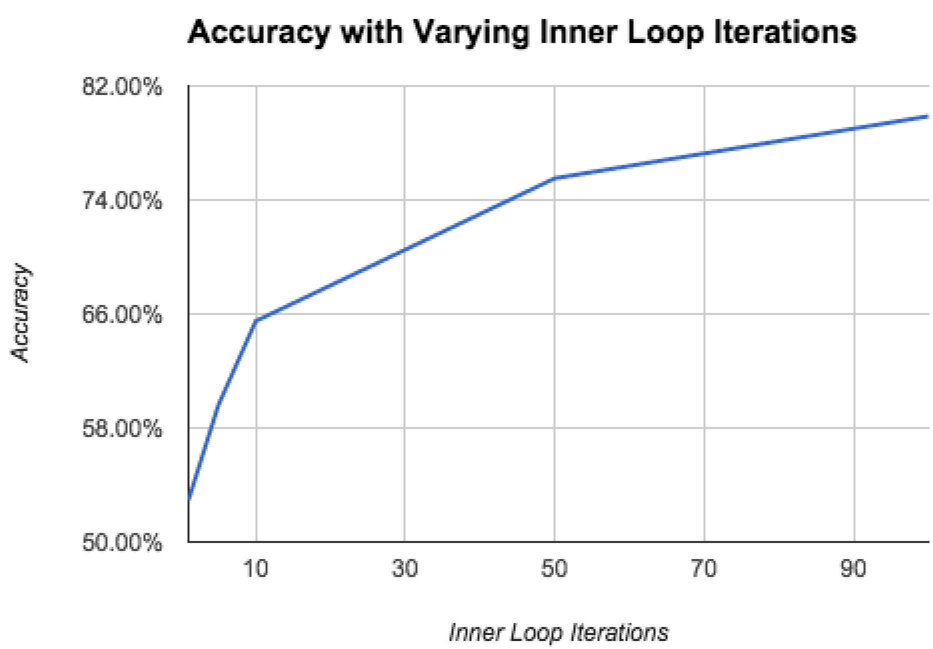
The parameters varied were instance size, final to initial temperature ratio, alpha, and inner loop iterations. The results found are outlined in the following graphs and tables. The default values used were:

defaultSize=40
defaultTempRatio=5
defaultAlpha=9
defaultInnerLoop=10

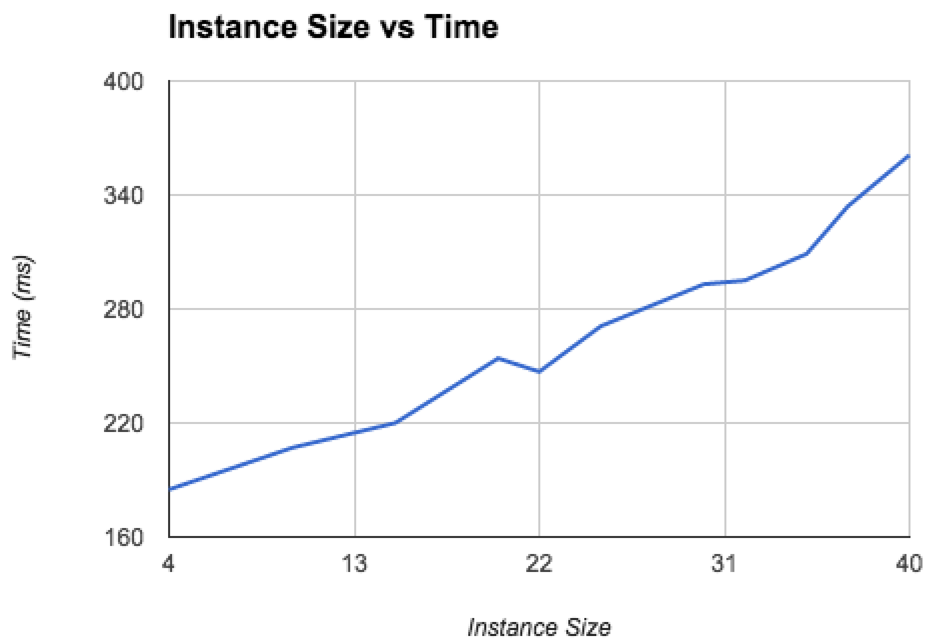


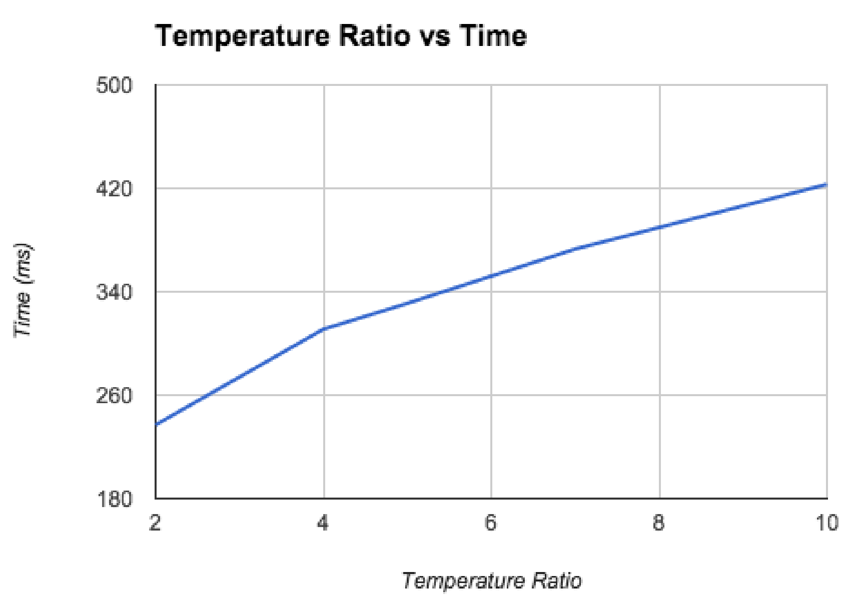
	Temperature Ratio				
	2	**4**	**5**	**7**	**10**
Average Accuracy	64.44%	65.46%	63.34%	61.42%	64.32%

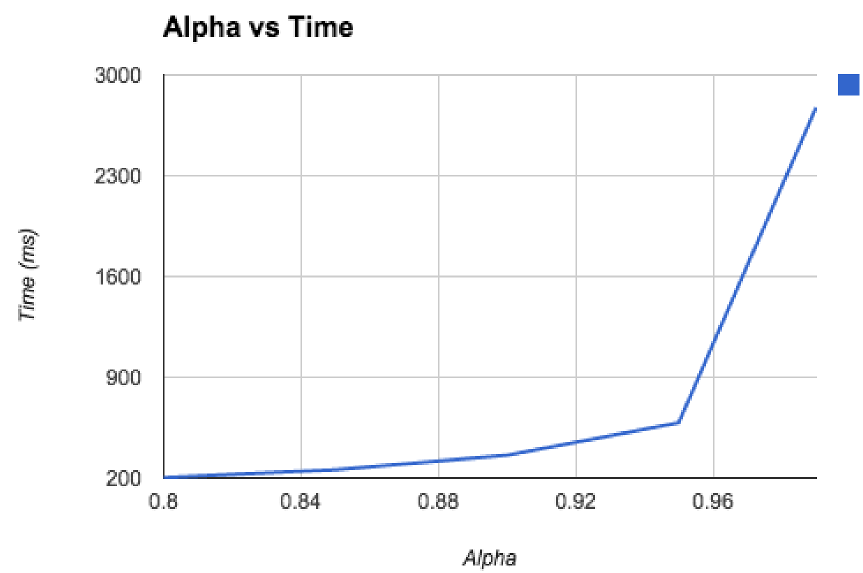


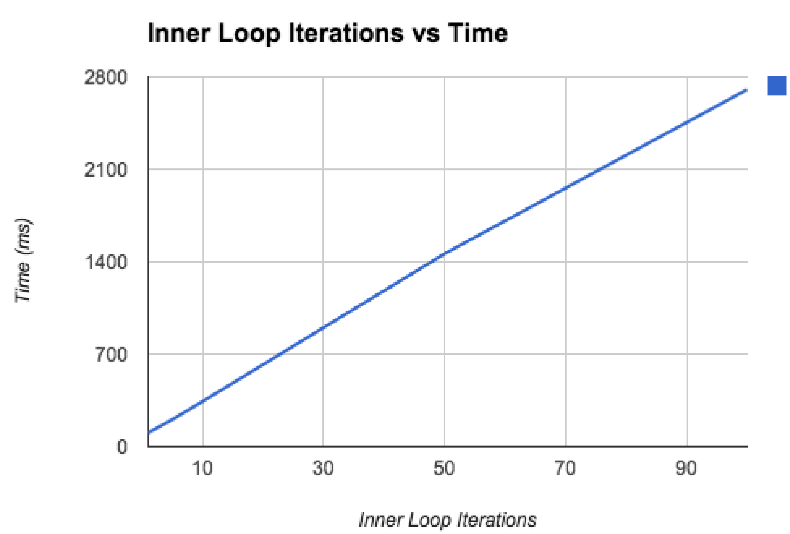


As we can see from these graphs and tables, the accuracy decreases as the knapsack size increases, this is due to the increased number of choices for the random neighbour function to choose. The accuracy increases exponentially as alpha increases and logarithmically as the inner loop iterations increases. The ratio between the final and initial temperature seemed to not affect the accuracy at all.
Next we'll look at the computation time as each parameter is varied.









The power of the heuristic was tested while turning the parameters to:
Instance Size: 40
Temp Ratio: 100
Alpha: 0.99
Inner Loop Ratio: 1000
The time for completion was **594.988 seconds** yielding 98.67% accuracy.

Conclusions:
This is a very powerful heuristic that can efficiently compute this problem. The parameters give a lot of freedom for the users, to gather the results as they desire. The alpha was one of the biggest contributors to success. As we can see from the heuristic with the parameters turned up extremely high, the accuracy approaches 100%.

SOURCE CODE:
https://edux.fit.cvut.cz/courses/MI-PAA/_media/en/student/rondepau/assignment4.zip