Reid Elkins

Homework 8 - CSCE 686

June 24, 2020

Problem a:

I will be using a simple genetic algorithm to solve the 0-1 knapsack problem which is used in my project. Each chromosome will be represented as a binary list with the length equal to the amount of potential objects to choose from. A ‘1’ in the list will represent that the item is being chosen to be in the knapsack and a ‘0’ will be there otherwise.

The fitness function for the algorithm is the sum of the values that correlate with the chosen items for each potential solution if the sum of the weights for those same correlated items is less than or equal to the max capacity of the knapsack. Otherwise, the fitness value is 0.

The population is a set of binary lists of size n.

Selection, choose the fittest parents based on the fitness function to be used in the crossover function.

Crossover function will be a one-point crossover. This means that a number between 0 and n - 1 will be randomly decided, on the parent lists, the values to the right of that point are swapped between the two parent chromosomes.

The mutation function changes a random gene from 0 to 1 or 1 to 0 with a given probability for all children that are also set to be mutants.

Delay termination will be a pre-determined number of generations that the algorithm will run for.

Algorithm Design for ss-ga:

* Name: stochastic-search genetic algorithm
* Domains: Ds is a set of satisfying solutions; the population size n is the cardinality of Ds
* Operations: I(x); x in Ds; x is a possible solution from population

O(x, z); x in Ds, z in Ds; z is a satisfying solution

* Name: stochastic-search-GA (Ds)
* Domains: Di is set of algorithm-internal solutions, Ds is set of satisfying solutions
* Imports: integer and list ADT
* Initialization of feasible solutions: Ds and Di empty
* Operations I(x); x in Ds

O(x,); x in Ds, z in Ds where

- next-solution-generator: x1 and x2 for xi in Ds, Ds  = Di

x1 and x2 are chromosomes in population with highest fitness where fitness function is

1. crossover function -

xi = xi’ and x2 = x2’ with random probability pc

2. mutation function

Randomly choose one of the newly created children

Change a bit from 0 to 1 or 1 to 0 in the child with random probability pm

Pseudocode:

* Name: stochastic-search-GA (Ds)
* Initial condition: list of values, v, of size n, list of weights, w, of size n, weight capacity, population size s, population h, list of size s of lists of size n that are made of up 0s and 1s which are randomly assigned, max generations g, crossover probability pc, mutation probability pm
* Body
  + while not g, do ss-ga loop:
    - for x in h:
      * calculate fitness of x
    - Selction: x1 is x in h with max fitness, x2 is x in h with second max fitness
    - Crossover: pick random value from 0-1

If value is greater than pc

Pick random int from 0 to n-1, j

Swap bit values from j to n from x1 and x2

* Mutation: pick random value from 0 - 1:

For both of the new children

If value is greater than pm

Pick random int from 0 to n-1, j

If bit in selected child is 1 at index j, change to 0

Else, change to 1

* Population is now the selected parents as well as the mutated children

End While loop

Calculate the fitness for all the chromosomes in the final population

Chromosome with max fitness is final solution

REFERENCES:

<https://www.dataminingapps.com/2017/03/solving-the-knapsack-problem-with-a-simple-genetic-algorithm/>

<https://medium.com/koderunners/genetic-algorithm-part-3-knapsack-problem-b59035ddd1d6>

Problem b :

3. 8:

Both of the tournament selection complexities being multiples of the amount of tournaments makes since as that decision determines how many times the selection is done. The other two selections also make since as the complexity is decided based on whether the lambda solutions are being added to the original mu solutions or if only the lambda solutions will be used.

3.9:

The SUS’s complexity is only O(n), thus quicker than the roulette wheel selection, since it samples all possible solutions by choosing them at evenly spaced intervals rather than the roulette wheel selection which finds the fitness of n individuals but then also sorts them to find the max.

*Reference for both problems:*

[*https://www.researchgate.net/publication/220132816\_Evolution\_strategies\_-\_A\_comprehensive\_introduction*](https://www.researchgate.net/publication/220132816_Evolution_strategies_-_A_comprehensive_introduction)

[*https://en.wikipedia.org/wiki/Stochastic\_universal\_sampling*](https://en.wikipedia.org/wiki/Stochastic_universal_sampling)