HW8 – CSCE686

Chad Willis

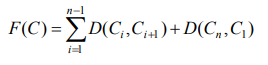
For this homework, I will be using a stochastic search via genetic algorithm for the Traveling Salesman problem. Genetic algorithms are inspired by the process of [natural selection](https://en.wikipedia.org/wiki/Natural_selection) and are used to find solutions to optimize search problems by relying on biologically inspired operators like mutation, crossover, and selection. The “art” of the algorithm comes from the intelligent exploitation of random search provided with historical data to direct the search into the region of better performance in solution space.

a.   
For TSP, we implement a chosen crossover and mutation operator type, for example:  
  
Crossover-  This represents mating between individuals. Two individuals are selected using selection operator and crossover sites are chosen randomly. Then the genes at these crossover sites are exchanged thus creating a completely new individual (offspring).   
-One point - part of the first parent is copied and the rest is taken in the same order as in the second parent  
-Two point - two parts of the first parent are copied and the rest between is taken in the same order as in the second parent  
-None - no crossover, offspring is exact copy of parents

Mutation - The key idea is to insert random genes in offspring to maintain the diversity in population to avoid the premature convergence  
-Normal random - a few cities are chosen and exchanged  
-Random, only improving - a few cities are randomly chosen and exchanged only if they improve solution (increase fitness)  
-Systematic, only improving - cities are systematically chosen and exchanged only if they improve solution (increase fitness)  
-Random improving - the same as "random, only improving", but before this is "normal random" mutation performed  
-Systematic improving - the same as "systematic, only improving", but before this is "normal random" mutation performed  
-None - no mutation

With that, we have a simple GA example for TSP

Travelling Salesman Problem  
Data:n nodes, distance matrix D n×n   
Goal: compute one cycle of minimum total length, visiting each node once.   
  
1. Chromosome and evaluation   
One solution can be coded as a node permutation C.   
Example: C = (5, 1, 4, 6, 3, 2)   
A Fitness Score is given to each individual which **shows the ability of an individual to “compete”.**  
The fitness (solution cost) is:



2. Initial population Pop of nc chromosomes  
Generate nc random permutations, compute their costs.   
  
3. Selection of two parents P1, P2 in Pop  
Binary tournament: randomly draw two chromosomes, keep the best as P1. Repeat to get P2.   
  
4. Simple one-point crossover   
Parent 1: 1 3 9 2|7 4 5 8 6   
Parent 2: 8 3 4 7|6 1 9 5 2   
Child 1: 1 3 9 2|8 4 7 6 5   
Child 2: 8 3 4 7|1 9 2 5 6

5. Simple mutation: swap two nodes  
Child: 1 3 9 2 7 4 1 8 6   
Mutated: 1 7 9 2 3 4 1 8 6  
  
6. Replacement rule  
Child C replaces the worst chromosome in Pop.   
7. Stopping criteria   
- max number of crossovers   
- max number of crossovers without improvement

Pseudocode which requires a local search procedure LS

initialize population Pop  
 improve each s in Pop: s ← LS(s)  
 repeat   
 select two parents P1, P2 from Pop  
 crossover P1 ⊗ P2 → C1, C2   
 for each child C do   
 improve: C ← LS(C)  
 mutate C (small probability)   
 replace one solution B in Pop by C   
 endfor   
until stopping criterion satisfied

Algorithm domain requirements specification form:

Name: stochastic search genetic algorithm  
domains: Ds is a set of satisfizing solutions- a population 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 satisfizing solution

Algorithm domain design specification form:

Name: stochastic search GA (Ds)  
Domains: Di is set of algorithm-internal solutions, Ds is a set of satisfying solutions  
Imports: ADT set, list, real/integer/character  
Initialization of feasible solutions -> Ds; Di empty  
Operations I(x); x in Ds  
O(x,z); x in Ds, z in Ds; condition on z being a satisfying solution

* Next-solution-generator -> x for x in Ds, Ds->Di
  1. Recombination(crossover) x -> y with crossover probability
  2. Mutation x-> y with mutation probability
  3. Feasibility(y) -> boolean (if true union(y,Di)) genotype
* Fitness/objective function mapping f(x) of each x in Di phenotype
* Selection Di -> Ds using f(x) as criteria, x in Di

Axioms: rules that could be used to test algorithm

Algorithm domain function specification form iterative:

Function stochastic-search GA (Ds)  
Initial condition: generate feasible D initial -> Ds, Di empty, pc, pm  
Body  
- While not time/generation termination do ss-ga loop:  
 Next-state solution/ population Ds, D s-> Di; do for each x in Ds, size n

* + - * 1. Crossover(x) = y with pc
        2. Mutation(x) = y with pm
        3. If feasibility(y) then union (y, Di) -> Di

Fitness calculation f(x) for each x in Di  
 Selection (Di) -> Ds based upon f(x), x in Di

-End ss ga while loop  
-Find optimal z in Ds  
END function

b.

Exercise 3.8 method is not as quick as roulette wheel selection. Although, in 3.8 the problem of high fitness controlling the selection which limits the diversity of genetics for the future generation. This problem of random roulette is not dealt with in 3.9, which is an O(n) method for selection for the roulette wheel. This is faster. With 3.9’s implementation, the solutions are put into a list a number of times proportionate to their fitness. So, n random selections from the last need to be made for the population size n.

References

1. <https://www.obitko.com/tutorials/genetic-algorithms/tsp-example.php>
2. <http://www.or.deis.unibo.it/gendreau_seminar/Memetic_VRP_Prins.pdf>
3. CSCE686 – GA- Lecture2020-1 ppt
4. <https://en.wikipedia.org/wiki/Genetic_algorithm>
5. <https://www.geeksforgeeks.org/genetic-algorithms/>
6. *J.E Beasley, P.C Chu,A genetic algorithm for the set covering problem,European Journal of Operational Research,Volume 94, Issue 2,1996,Pages 392-404,ISSN 0377-2217, https://doi.org/10.1016/0377-2217(95)00159-X.*