John Beighle

CSCE686 Homework 8, Spring'20

HW8: **(due 6/24)** "Please indicate appropriate references"

a) (30pts) Use a Genetic Algorithm for your project solution using one of the NPC problem objectives (**or both?)***. Define representation and don't forget PD/AD feasibility functions. Explicity indicate your GA algorithm pseudo code (Talbi form and search elements). Discuss the "art" of GA parameter/function selection for your PD/AD. Reference at least one appropriate paper providing insight to your PD and this GA AD. This also could be a GA variation as is employed in your project!

Background: In the interest of thesis related application [See Project+ Documentation] for using a simple algorithm, a hillclimber, to create a daily flight plan for a reconnaissance satellite, wherein the satellite has to change orbits and altitudes as many times as possible during a 24 hour period to perform the mission of taking photographs at specific positions through-out the day and, in addition, moving frequently to lower the risks of being approached and disabled by an enemy satellite, the investigation into multi-objective optimization for the Travelling Salesman Problem (TSP) is considered.

The question of import in the final class project is **to compare a Multi-point, multi-objective optimization using a Genetic Algorithm hill-climber against a deterministic, multi-objective, Single-point hill-climber, both solving the TSP**. *Contrasting these different methods will illustrate two different approaches to multi-objectivization, associated issues, and possible ways in which multi-objectivization can be approached.* In this homework assignment however, the Genetic Algorithm is described without the Single-point hill-climber. [See Project+ Documentation for the comparison contrast]. **The big idea is to explore the concept of a TSP hill-climbing solution that maintains the simplicity and performance of hill-climbing but circumvents the problems of local optimums.**

To multi-objectivize the TSP problem, sub-problems need be identified and solved. Normally, dependencies exist between cities of the problem which make it challenging to decompose. Using the concept of divide and conquer conveys a simple idea for decomposition, although not ideal, simply divide the problem into multiple sub-tours where each of these are to be minimized. This can be done several ways, and some are better than others depending on the data; however, applying a general method, sub-tours are broken into two-objective formulations to be minimized and defined by two cities. For input, set $C = \{c_1, c_2, \ldots, c_N\}$ of cities and for each pair $\{c_i, c_j\}$ of distinct cities there is a distance $d(c_i, c_j)$. Our goal is to find an ordering π of the cities that minimizes the quantity. This becomes:

John Beighle

$$\begin{aligned}\text{“minimize” } \mathbf{f}(\pi, a, b) &= (f_1(\pi, a, b), f_2(\pi, a, b))\\ \text{where } f_1(\pi, a, b) &= \textstyle\sum_{i=\pi^{-1}(a)}^{\pi^{-1}(b)-1} d(c_{\pi(i)}, c_{\pi(i+1)})\\ \text{and } f_2(\pi, a, b) &= \textstyle\sum_{i=\pi^{-1}(b)}^{N-1} d(c_{\pi(i)}, c_{\pi(i+1)}) +\\ &\quad \textstyle\sum_{i=1}^{\pi^{-1}(a)-1} d(c_{\pi(i)}, c_{\pi(i+1)}) + d(c_{\pi(N)}, c_{\pi(1)})\end{aligned}$$

Genetic Algorithms basic definitions:

- **Gene:** a city (represented by (x, y) coordinates)
- **Individual (or chromosome):** a single route between 2 cities
- **Population:** a collection of possible routes (individuals)
- **Parents:** two routes combined to create a new route
- **Mating pool:** collection of parents used to create the next population (next generation of routes)
- **Fitness:** tells us how good each route is
- **Mutation:** introduces variation into the population by randomly swapping two cities in a route
- **Elitism:** a way to carry the best individuals into the next generation

The basic steps of the Genetic Algorithm:

1. Create the population
2. Determine fitness
3. Select the mating pool
4. Breed
5. Mutate
6. Repeat

Candidate Set: The dataset for a candidate solution used by the proposed algorithm will be EUC-50 dataset from TSPLIB which is a dataset wherein cities are listed as numbers 1…50 each with a simple set of coordinates that designate the city’s location within the plane. The distance between two cities is the Euclidian distance.

Selection Function (Representation): The initialization Init_pop(P) begins by selecting two different cities at random and reverses the order of all cities in between and including them. This string is an N-gene permutation of each city’s corresponding number.

Feasibility Function: Tells us how good each route is. A Pareto optimal set, $X^* \subseteq X$, can be used to find a set of solutions with the property:

$$\forall x^* \in X^* \cdot \nexists\, x \in X \cdot \succ x^*$$

Where $x \succ x^* \Leftrightarrow ((\forall i \in 1..K \cdot (f_i(x) \geq f_i(x^*)) \wedge (\exists i \in 1..K \cdot f_i(x) > f_i(x^*)))$. Where $x \succ x^*$ reads: x dominates x*, and solutions in the Pareto optimal set are also admissible. For two solutions x and x’, $x \sim x'$ if and only if $\exists i \in 1..K \cdot f_i(x) > f_i(x') \wedge \exists j \in 1..K \cdot j \neq i \cdot f_j(x') > f_j(x)$. This pair is said to be incomparable and with respect to each other are nondominated. Since pairs like these are no worse than any other, if the selection operator allows, a hillclimber is free to move to it. **This is the key of the Pareto hill-climber reducing the problem of local optima.**

Objective Function: Replace parents with children as children yield better results.

Solution Function (Termination): Terminate by constant *num_evals* evaluations. Num_evals = 500,000 times.

Algorithm 1 Mutation only, Multi-point Hill-Climber, Genetic Algorithm Pseudocode:

```
Input:     p ← 0
           Init_pop(P) //Candidate solutions population initialized randomly
Repeat:    $x_1$ ← Rand_mem(P), $x_2$ ← Rand_mem (P) // Selection: 2 random parents
           x'1 ← Mutate(x 1), $x'_2$ ← Mutate($x_2$) // 2 parents into 2 children randomly
           if (H($x_1$, x'1) + H($x_2$, $x'_2$) > H($x_1$, $x'_2$) + H($x_2$, $x'_1$))
              Swap($x'_1$, $x'_2$) // Feasibility: compare new hamming distances and swap
           if (f($x'_1$) > f($x_1$))
              P ← P U $x'_1$ \ $x_1$ //Objective: replace parent w child in P if it is "non-worse"
           if (f($x'_2$) > f(x2 ))
              P ← P U $x'_2$\ $x_2$ //Objective: replace parent w child in P if it is "non-worse"
Until:     Stopping criteria satisfied //500000 iterations
Output:    return Best(P) //Solution
```

b) (5pts) Checkout Talbi Exercises 3.8 and 3.9 and understand and discuss very briefly the answers given in the text.

Perhaps the key take away from these two problems, since the answers are given and especially because 3.9 highlights the high performance of SUS is that the efficiency of the selection mechanisms in evolutionary algorithms is one of the most crucial concerns as G. Dossi expresses in this quote:

“Interactions, coordination and selection While (imperfect) adaptation and persistent discovery generate variety, collective interactions within and outside markets operate, first, as mechanisms of information exchange and coordination, and second, as selection mechanisms, generating differential growth (and possibly also survival) of different entities that are the ‘carriers’ of diverse technologies, routines, strategies, etc. Indeed, crucial issues here regard (i) the coordinating power of whatever “invisible (or visible) hand” of decentralized interactions; (ii) the drivers, powers and **efficiency of selection mechanisms** and (iii) the interactions between the foregoing two processes.”[2]

John Beighle

References:

[1] Reducing Local Optima in Single-Objective Problems by Multi-objectivization. Joshua D. Knowles, Richard A. Watson, and David W. Corne
[2] Economic Coordination and Dynamics: Some Elements of an Alternative “Evolutionary” Paradigm by GIOVANNI DOSI