Route Planning For Cyclists

Xiaoyang Guo - 999578514, Chi Zhang - 1001234130, Shixiong Gao - 1007038260
Department of Electrical and Computer Engineering - University of Toronto

Abstract

In this paper, route planning for cyclists using bio-inspired algorithms is explored. The route must traverse a set of locations once and only once and finally return to the starting location, which makes the route planning essentially a travelling salesman problem (TSP). To solve it, a python program utilizing Simulated Annealing (SA), Particle Swarm Optimization (PSO), and Firefly Optimization (FO) is implemented and tested. A comparative study between the three bio-inspired algorithms reveals their difference in qualitative behavior and similarity in performance. In the end, it was shown that the program can generate safe and cycling-suited routes that meet the user's fitness goal, within a reasonable timeframe.

I. Introduction

Cycling as a physical exercise is shown to have many positive effects on both body and mind. Route planning is an important preparation for people who regard cycling as a regular exercise. With route planning, cyclists would be able to ride on safe and flat roads. In addition, they could explore new regions and places by riding on different routes. The objective of the project is to design and implement an algorithm to plan loop routes for a cyclist based on his/ her training goal. By using the algorithm, the user can save efforts in finding routes that are safe to ride on while meeting his/ her training goal. The user can also discover new routes to make the training more enjoyable, since the algorithm is stochastic and might generate a new route at each run.

The project faces two main challenges. First, there are not many bike route planning applications on the market. Benchmarking would be limited to comparing against existing research projects. Second, cyclists are sensitive to many factors of a route, such as safety, environment and elevation, hence the bike route planning problem is not simply searching for the fastest route but needs to make trade-offs among multiple objectives.

The paper first conducts a survey on existing solutions to the similar bicycle routing problems. Next, it describes the mathematical model for the problem. Then, it proposes three bio-inspired algorithms to solve the problem and presents the evaluation results of the algorithms. Finally, it suggests future improvements for the project.

II. Literature Review

A. Practical Multicriteria Urban Bicycle Routing [1]:

Unlike our project which targets bicycle route planning for training, this related research paper mainly tackles transportation based (from point A to point B) bicycle routing problems. Though the goal is quite different, there are a lot we can learn from their research.

The routing algorithm used is the combination of multi-criteria label-setting algorithm and some heuristic speedup techniques. The criteria selection is broken down into 3 parts: time, elevation and comfort. The comfort criteria takes road surface quality, traffic volume, obstacles and need for dismount into consideration.

The highlight for this article lies in the comfort criteria selection and execution. It truly captures what cyclists care about. Which was rarely accomplished in prior researches.

B. Conflation of OpenStreetMap and Mobile Sports Tracking Data for Automatic Bicycle Routing [2]:

This project uses a mobile sport-tracking app to retrieve exercise routes from multiple different users. Then based on the massive historical data, new routes are planned and recommended to users. Although the process of build-in trajectories is different, we could

learn some related pre-processing steps and algorithms from their project.

For pre-processing data, they did not pick up all data instead discarding some points which are close to each other, and they set a range of road's speed in order to filter the ways that are accessible to cyclists, excluding all motorways and footways. Those technologies and ideas would help us to build a more clean graph and improve the performance of search algorithms. Their project used the advanced HMM-based algorithm, which calculates the most probable route using distance and topological data of the network; then they used the weighted networks to find routes with maximum popularity given a start and a goal position by Dijkstra's shortest path algorithm. The criteria selection contains total elevation gain, vegetation, turns, and intersection with traffic.Those would contribute to the value of popularity, which would affect the output of routines.

The highlight part of this article is the use of cost function to produce biased tracking datasets from crowd sources. Therefore, the calculated routines would be based on the history and popularity of other users who have similar preferences and body conditions. Our project would regard this functionality as an optional and advanced version if possible in future development.

C. Multi-Objective Design of Time-Constrained Bike Routes using Bio-inspired Meta-Heuristics [3]:

The paper uses four population-based bio-inspired meta-heuristics and the OpenTripPlanner (OTP) to generate a group of bike routes that best balance the two conflicting objectives: the distance of the route and the safety level of the route, while complying with the time constraints. The paper uses the bio-inspired algorithms to modify a set of factors that are used by the OTP route generation engine to calculate a route, then passes the modified factors to the engine to obtain a route that reflects the factors, and repeats the process until the output routes converge to a Pareto front.

The greatest advantage of the approach is that it does not focus on the computation of the route. Instead, it uses the well-implemented OTP engine to perform the computation and focuses on bio-inspired algorithms to tune the weights of the criteria used by the computation, so that it can obtain a set of non-dominated fittest routes that have different trade-offs.

D. Comparison

Our design differs from the above mentioned projects qualitatively. Most of the existing projects aim to find an optimum route between two points, whereas our design dedicates to the case in which the rider will come back to the origin, which we believe to be a more common exercise scenario.

III. Problem Formulation and Modeling

The problem is modeled as a travelling salesman problem (TSP). A set of points of interest (POIs) are selected around the starting point specified by the user. The goal is to find the fittest route that starts from the starting point, traverses all POIs once and only once, and finally returns to the starting point.

A. Data and Sources

- Map (OpenStreetMap)
- Geocoding (Nominatim)
- Elevation of each OSM node (Google Maps Elevation API)

B. Conditions

The following conditions are specified by the user before the algorithm runs:

- The address of the starting point.
- The number of calories to burn, which can be any value between 200 kcal and 4000 kcal.
- The average speed, which can be any value between 10 mph and 20 mph.
- The weight of the user in lbs.
- The weight of the bike in lbs (default is 30 lbs).

C. Constraints

The route must be subject to the following constraints:

- The route must be a subset of bicycle networks.
- The route must traverse all the POIs once and only once.
- The ending point of the route must be the same as the starting point.

D. Criteria

The following criteria capture the preferences towards a route:

- Calorie: Smaller percentage error between the calorie goal and estimated calories burned by traversing the route is preferred.
- Safety and Convenience:
  - Traffic Light: Less traffic lights are preferred.
  - Road Surface: Paved road surfaces are preferred to unpaved road surfaces.

E. POI Selection

A rectangular search space centering at the starting point is used to select the POIs. The dimensions of the search space are determined by the calorie goal and average speed. Here is the formula used for determining search range (coefficients are tuned experimentally):

$$Range = calorie\_goal \times average\_speed \div (2 \times \pi \times calorie\_burining\_rate) \times coefficient\_a + coefficient\_b$$

$$coefficient\_a = 0.7$$

$$coefficient\_b = 0$$

The search space is then evenly partitioned into 4 sections. Several POIs will be randomly chosen from each section, so that the POIs are distributed around the starting point in a relatively uniform manner.

F. Mathematical Model

- Offline calculations for TSP

Let set $S = \{O, P_1, \ldots, P_N\}$, where O is the starting point and $P_1, \ldots, P_N$ are all the N POIs. For each ordered pair (U, V) where $U \in S, V \in S, U \neq V$, find the route from U to V with minimum λ:

$$\lambda = t_{movement} + t_{delay}$$

λ is the total travel time of a route in seconds. It is the sum of the time of movement $t_{movement}$ and waiting time at traffic lights t_{delay}, which are both in seconds.

$$t_{movement} = t_{movement-paved} + t_{movement-unpaved}$$

Where $t_{movement-paved}$ is the time of movement on paved road segments in seconds, and $t_{movement-unpaved}$ is the time of movement on unpaved road segments in seconds.

$$t_{movement-paved} = l_{paved} \div v \times 3600$$

$$t_{movement-unpaved} = l_{unpaved} \div (0.8 \times v) \times 3600$$

Where l_{paved} is the total length of paved segments of the route and $l_{unpaved}$ is the total length of unpaved segments of the route, which are both in miles. v is the average speed specified by the user in miles per hour. Note that v is reduced by timing a factor of 0.8 on unpaved road segments so that those segments are penalized.

$$t_{delay} = 15 \times n_{light}$$

t_{delay} is used to penalize traffic lights. n_{light} is the number of traffic lights on the route. The average waiting time at a traffic light is assumed to be 15 seconds [4].

- TSP

The TSP's design vector $X = [\, x \,]^T$ consists of a single decision variable x, which is the permutation of all the POIs. Let $x = [N_1, \ldots, N_N]$, where $N_i \in \{P_1, \ldots, P_N\}, i \in \{1, \ldots, N\}$ and $N_i \neq N_j, i \neq j$.

Let the TSP route that traverses the POIs in the order specified by x be $r = [O, N_1, \ldots, N_N, O\,]$.

The goal is to find the value of X that minimizes the objective function Ω:

$$\Omega = |\,(C_r - C_{goal})\,| \div C_{goal}$$

Where C_r is the calculated calories burned after completing r, and C_{goal} is the calorie goal set by the user, which are both in kcal.

$$C_r = c \times t \div 3600$$

Where t is the total time to complete r in seconds, and c is the calorie burning rate in kcal per hour [5].

$$t = \sum_{i=1}^{N+1} \lambda_{i,\, i+1}$$

Where $\lambda_{i,\,i+1},\ i \in \{1,\ \ldots,\ N+1\}$ is the travel time of the sectional route connecting the i^{th} point and $(i+1)^{th}$ point in r. Note that all possible sectional routes and their travel times are found in offline calculations for TSP.

$$c = [v \times w \times (0.0053 + g) + 0.0083 \times v^3] \times 14.4$$

Where v is the average speed in miles per hour, w is the total weight of the user and bike in pounds, and g is the average gradient of r.

$$g = h \div l_{horizontal}$$

Where h is the total elevation of r in miles, and $l_{horizontal}$ is the total horizontal length of r in miles.

IV. Proposed Solution

To solve the TSP described in the mathematical model, three bio-inspired algorithms are used.

A. Simulated Annealing

Simulated Annealing is a metaheuristic approach to find the optimum solution to a function in a large search space. The process works in the same way as annealing in metallurgy, where seemingly bad states are accepted in the early "volatile" stage, but less and less likely to be accepted in the late "stable" stage.

The probability of accepting a worse state is represented by the following formula:

$$p = exp(-\Delta\Omega) / (k \times exp(-lam \times iteration))$$

States with lower cost are always accepted. k and lam are parameters that are tuned experimentally.

B. Particle Swarm Optimization

The particle swarm optimization (PSO) algorithm is a metaheuristic algorithm based on the concept of swarm intelligence capable of solving complex mathematics problems existing in engineering. It is inspired by the behaviour of social organisms in groups, such as bird and fish schooling or ant colonies. This algorithm emulates the interaction between members to share information. In the beginning, a set of random particles is initialized. Each particle represents a solution in the search space, i.e. a permutation of the POIs. In each iteration, the cost of each particle is calculated, and the particles with the lowest costs are selected as the leaders. Each leader leads a swarm that contains a number of particles. The particles in a swarm are mutated by crossovering with the leader, and all the leaders are then mutated by crossovering with each other. Therefore, the particles seek the optimal solution with both their own experience and the experience of the swarm as a whole.

C. Firefly Optimization

The "firefly algorithm" is classified as swarm intelligent, metaheuristic and nature-inspired, and it emulates the characteristic behaviors of fireflies. It adapted the behavior of firefly swarms to develop an algorithm for optimizing functions with multiple optima. In particular it used the concept of how the brightness of individual fireflies drew them together and a randomness factor to encourage exploration of the solution space.

In our algorithm, the shortest path is not the only goal to pursue, and the function of luminosity wouldn't be that the shorter of two paths would be more luminous and attractive. The luminous function would be equal to the cost of the route plus the number of nodes, because we are trying to minimize both of them.

$$luminosity_{\,i} = luminosity_{\,j} \times e^{-\gamma \times distance}$$

This γ is the attractiveness coefficient between the flies which will be assumed constant to simplify the implementation.

V. Performance Evaluation

To evaluate the performance of the algorithms, the following user input is used:

- Calorie goal: 1000 kcal (calorie)
- Average speed: 20 mph
- Weight: 150 lbs
- Bike weight: 30 lbs
- Starting address: "1070 Major Mackenzie Dr, Richmond Hill, ON"

The search space and POIs are shown in Figure 2.

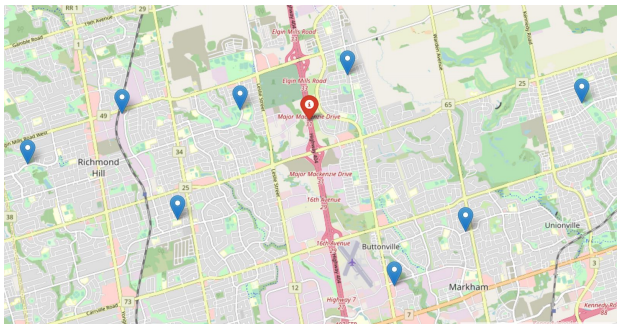

Figure 2. Search space and POI selections

The algorithms are evaluated using two metrics, percentage deviation from calorie goal and time to converge. Since the performance of the algorithms is heavily impacted by the parameters, a wide range of parameter values and combinations are tried. Below is the parameter value combination that yields the lowest percentage deviation from calorie goal for each algorithm:

A. Simulated Annealing

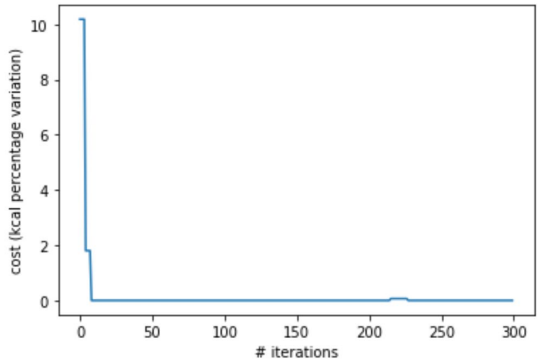

Figure 3. Simulated Annealing Cost vs Number of Iterations

- Best parameters: {k=5, lam=0.0005}
- Percentage deviation from calorie goal: 0.000539%
- Time to converge: 0.0076 sec

B. Particle Swarm Optimization

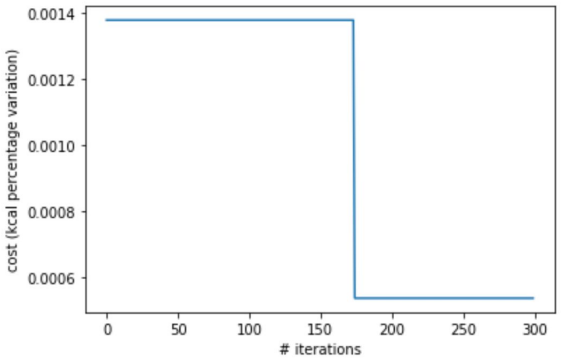

Figure 4. Particle Swarm Optimization Cost vs Number of Iterations

- Best parameters: {particles_swarm=100, num_of_swarms=4}
- Percentage deviation from calorie goal: 0.000539%
- Time to converge: 9.154 sec

C. Firefly Optimization

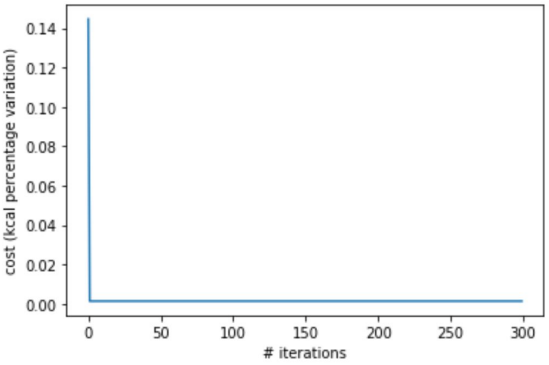

Figure 5. Firefly Optimization Cost vs Number of Iterations

- Best parameters: {gamma=1, flies=25}
- Percentage deviation from calorie goal: 0.01376%
- Time to converge: 0.154 sec

In order to reduce error, three trials were run for each parameter combination, and the best result was used. All three algorithms are able to give qualifying results. The Firefly Optimization and the Particle Swarm Optimization can usually start at a relatively low-cost random state since their spreaded “particles” can cover larger portions of the search space. They can usually converge in a few number of iterations, but sometimes can take around 200 iterations to converge. Simulated Annealing on the other hand relies on a probability formula to climb out of local minima for

diversification. Based on observation, it usually requires around 200 iterations to converge. However, due to the shorter time needed per iteration, Simulated Annealing is the quickest out of the three algorithms we tested. The reason why Simulated Annealing finishes faster per iteration is that it generates a new permutation by simply swapping two random elements in the current permutation. By contrast, the other two algorithms use partially-mapped crossover (PMX) to generate new permutations, which adds a significant overhead to each iteration.

VI. Conclusions and Recommendations

In this project, we focused on loop route planning for cyclists. The following steps are performed by the algorithm:

1. Select POIs around the starting point.
2. Calculate the best bicycle route between each pair of points.
3. Use bio-inspired algorithms to find the optimum order to traverse all the POIs once and only once.

The objective function used in step 3 solely targets minimizing percentage variation in calorie cost compared to the calorie goal. The algorithm is able to produce routes that stick very well to the calorie goal. On the other hand, some disadvantages were observed on the routes produced, including repetition in routes and 180 degree turns, which are not ideal for cycling. We propose the following future improvements:

1. Use A* algorithm instead of Dijkstra algorithm to find the fittest route between each pair of points, since A* algorithm uses a heuristic to find the optimal solution and can improve the runtime.
2. Only select POIs from the OSM nodes that have certain values for the amenity tag (e.g. community centers and parks) so that the selected POIs are more controlled.
3. Penalize turns, especially left turns, so that the generated route is safer.
4. Penalize repeated paths, so that the generated route is more refreshing.

VII. References

[1] *Practical Multicriteria Urban Bicycle Routing - IEEE Journals & Magazine*. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7519077. [Accessed: 27-Oct-2020].

[2] C. Bergman and J. Oksanen, “Conflation of OpenStreetMap and Mobile Sports Tracking Data for Automatic Bicycle Routing,” *Wiley Online Library*, 10-Mar-2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1111/tgis.12192. [Accessed: 27-Oct-2020].

[3] E. Osaba, J. D. Ser, M. N. Bilbao, P. Lopez-Garcia, and A. J. Nebro, “Multi-objective Design of Time-Constrained Bike Routes Using Bio-inspired Meta-heuristics,” *SpringerLink*, 16-May-2018. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-91641-5_17. [Accessed: 27-Oct-2020].

[4] “Traffic Signal Timing Manual,” *Traffic Signal Timing Manual: Chapter 5 - Office of Operations*. [Online]. Available: https://ops.fhwa.dot.gov/publications/fhwahop08024/chapter5.htm. [Accessed: 27-Oct-2020].

[5] “Thread: how do you determine calories burned?,” *Road Bike Cycling Forums RSS*. [Online]. Available: https://forums.roadbikereview.com/racing-training-nutrition-triathlons/how-do-you-determine-calories-burned-11524.html. [Accessed: 27-Oct-2020].