GEE9018 (5094) F'17: Parallel Programming
Programming Assignment 01

[Problem] 2-opt local search algorithm with parallelization on TSP

In optimization, 2-opt is a simple local search algorithm first proposed by Croes in 1958 for solving the traveling salesman problem (TSP). In Figure 1, given a route (…, A, E, D, C, B, F, …) in advance, the 2-opt strategy is to choose two cities (points) randomly and evaluate the improvement of the total distance after swapping. For example, city B and city E are chosen in Figure 2. The total distance is reduced for this new route (…, A, B, C, D, E, F, …), which becomes the next target in the 2-opt iteration. On the contrary, if there is no improvement after swapping two cities, this new route will be discarded.

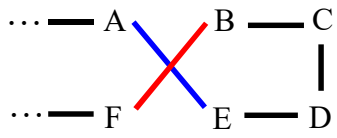

Figure 1

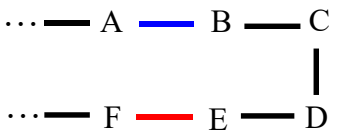

Figure 2

The psuedo code is shown as follows:

```
2optSwap(route, i, k) {
    1. take route[0] to route[i-1] and add them in order to new_route
    2. take route[i] to route[k] and add them in reverse order to new_route
    3. take route[k+1] to end and add them in order to new_route
    return new_route;
}
```

For the example shown in Figure 1 and Figure 2:

```
example route: A => E => D => C => B => F => A
example i = 1, example k = 4
new_route:
    1. (A)
    2. A => ( B => C => D => E )
    3. A => B => C => D => E => ( F => A )
```

Input city coordinate (.dis files) : The input file contains a set of points (cities) with corresponding index, x and y axis. The first line is the number of cites, followed by each cites' index, x axis, and y axis. Each number in this file is positive integers.

```
53                  // number of cities
1  42  30           // index, x axis, y axis
2  52  13
3  49  48
…
…
…
50  49  6
51  44  37
52  43  29
53  3   8
```

Input route (.in files): This input route file contains the default route which is the target to be optimized.

```
1876.53          // Total distance
1              // City index
32
13
51
36
9
…
…
8
29
34
10
52
```

Output optimized route (.out files): The output optimized route is the final trip which your program returns. Make sure that the output file will be found in time!

```
773.56      // Total distance
1           // City index
36
10
32
52
51
9
…
…
29
13
34
8
```

Execute: The name of your program must be "ans_2opt" and be able to parse the input and output file name.

```
> ./ans_2opt   <input_city_coor>   <input_route>   <output>
> ./ans_2opt   test0.dis   test0.in   test0.out       // An example of case "test0"
```

Verify: We will provide an executable code "verify_2opt" for verification.

```
> ./verify_2opt   <input_city_coor>   <your_output>
> ./verify_2opt   test0.dis   test0.out

This is an evaluator.
                    -- Written by TA
================================
Pass!
```

Limitation:

1. Please implement your program in **C** with Pthreads.
2. Please report the best answer of your program in **10 minutes**.

Notice:

1. Please upload your code to E3 with the filename "ans_2opt.c".
2. Please upload your report with the filename "studentID_hw1_report.pdf" and follow the given report template.

Score:

1. Correctness and Reduce at least 50% distances (30%)
2. Distance comparisons with those from other classmates (60%)
3. Quality of written report (10%)

Reference:

2-opt Algorithm *https://en.wikipedia.org/wiki/2-opt*