ChatGPT

TSP with Penalties Solver — README

Overview

This project is a C implementation for solving the Traveling Salesman Problem (TSP) with Penalties. The code uses:

- Morton order (space-filling curve) for a quick initial tour
- Variants of the 2-opt heuristic for tour improvement
- Pruning logic to potentially skip cities if the penalty is less than the added travel distance

IMPORTANT: The solution quality and runtime depend *heavily* on key parameters. You MUST experiment to find the best values for your data. See the "Parameter Quick Reference" below.

Usage

Build:

```
gcc -O2 -o tsp_solver tsp_solver.c -lm
```

Run:

```
./tsp_solver <inputfile> [--maxCities N]
```

- `<inputfile>`: Your city/penalty input file
- `--maxCities N` (optional): Limit the number of cities loaded

Output: Results are printed to the console and also written to `output.txt`.

Input Format

- First line: Penalty value (integer)
- Each subsequent line: `<id> <x> <y>` for each city

Example:

```
100
1 12 35
```

```
2 54 11
3 76 78
...
```

Parameter Quick Reference & Optimization Guide

2-opt Methods

- **Full 2-opt:** Used for small inputs (`n <= 5000`). Tries every possible swap for max improvement, but is slow on big instances.
- **Local 2-opt:** Used for medium inputs (`n <= 20000`). Limits swaps to a `window` around each city (default in code: `window=500`).
- **Random-region 2-opt:** For large inputs, runs local 2-opt in `K` random segments of size `window` (defaults: `window=1000`, `K=3`).

Pruning

- Number of pruning passes: Default is `3`. You can change this for more/less aggressive city skipping. More pruning = faster tour but may skip too many cities.

How To Get Good Results?

You MUST experiment to tune these parameters for your own data!

- Try different `window` values for local/random 2-opt. (e.g., 200, 500, 1000, etc.)
- Try more/less pruning steps, especially if your penalty value is high or low.
- Try more random 2-opt regions for large inputs (`K=5` or more) if you have time.

There is no single "best" setting! What works for one dataset may not work for another.

Tips

- Monitor runtime! Full 2-opt is slow for >5000 cities. Local/random 2-opt is much faster for large datasets.
- The code prints debug output, including timing and tour lengths after each step.
- You can adjust time limits in the code for local 2-opt if needed.

License