Mikayla Timm

10/25/2015

COP4534

TSP Functional Decomposition

|  |  |  |
| --- | --- | --- |
| File | Functions | Purpose |
| BruteForce.h | None | Function declarations for BruteForce.c |
| BruteForce.c | 1. void PermuteIter(int);  2. void perm1(int, long long unsigned int);  3. void swap(int, int);  4. void printS(int);  5. double calculateCost(int); | 1. Main function called for performing the BF approach.  2. Finds every permutation and saves optimal path and cost.  3. Swaps 2 items in an array  4. prints current path  5. calculates the cost of a path |
| Genetic.h | None | Function declarations for Genetic.c |
| Genetic.c | 1. void GA(int, int, double, double, int);  2. void shuffle(int \*, int);  3. double calcCostGA(int, int \*);  4. void mutate2(int, int \*);  5. void newGen(int); | 1. take in presets and perform GA. Find minimal cost.  2. Shuffles the citiies in the array  3. calculates cost of tour  4. mutate 2 indices, swap one on left half with one on right half  5. Create new generation of tours using mutations, elites, and shuffling of remaining tours. Sorts the costs. |
| MergeSort.h – adapted from DSA1 | None | Function declarations for MergeSort.c |
| MergeSort.c – adapted from DSA1 | 1. void mergeSort(double \*, int \*, int);  2. void merge(double \*, double \*, double \*, int \*, int \*, int \*, int, int, int); | 1. Recursive, create 2 subarrays as it goes down, call merge when subarray size=1.  2. Compares the numbers in the arrays (costs) and places them in sorted order. Also sorts the corresponding indices to save the locations of the best costs. |
| TSPTest.c | 1. void getWeights(FILE \*citiesfile);  2. void printBest(int numCities);  3. int main(); | 1. Reads in path weights, fills graph.  2. Prints best path/lowest cost for brute force.  3. runs all the other functions. |