

CS 202 Iditarod - Kaltag

Bryan Beus

April 22, 2020

Source Code Link: <https://github.com/siddhartha-crypto/cs202/tree/master/iditarod/kaltag>

1 Design

1.1 SVG Output - Part I

I intend to make the SVG a separate class.

In looking at the two challenges, I believe that it's best to design the class so that it can easily display a work-in-progress output. Therefore, I will attempt to do both this assignment and the next assignment simultaneously.

2 Post Mortem

2.1 TSPLIB

I struggled a little bit with getting the binary to render properly. In the end, I resulted to trial and error with the size for which I would add in the data at the end of the svg file.

3 Commit History

3.1 (First Half – Both Projects Simultaneous)

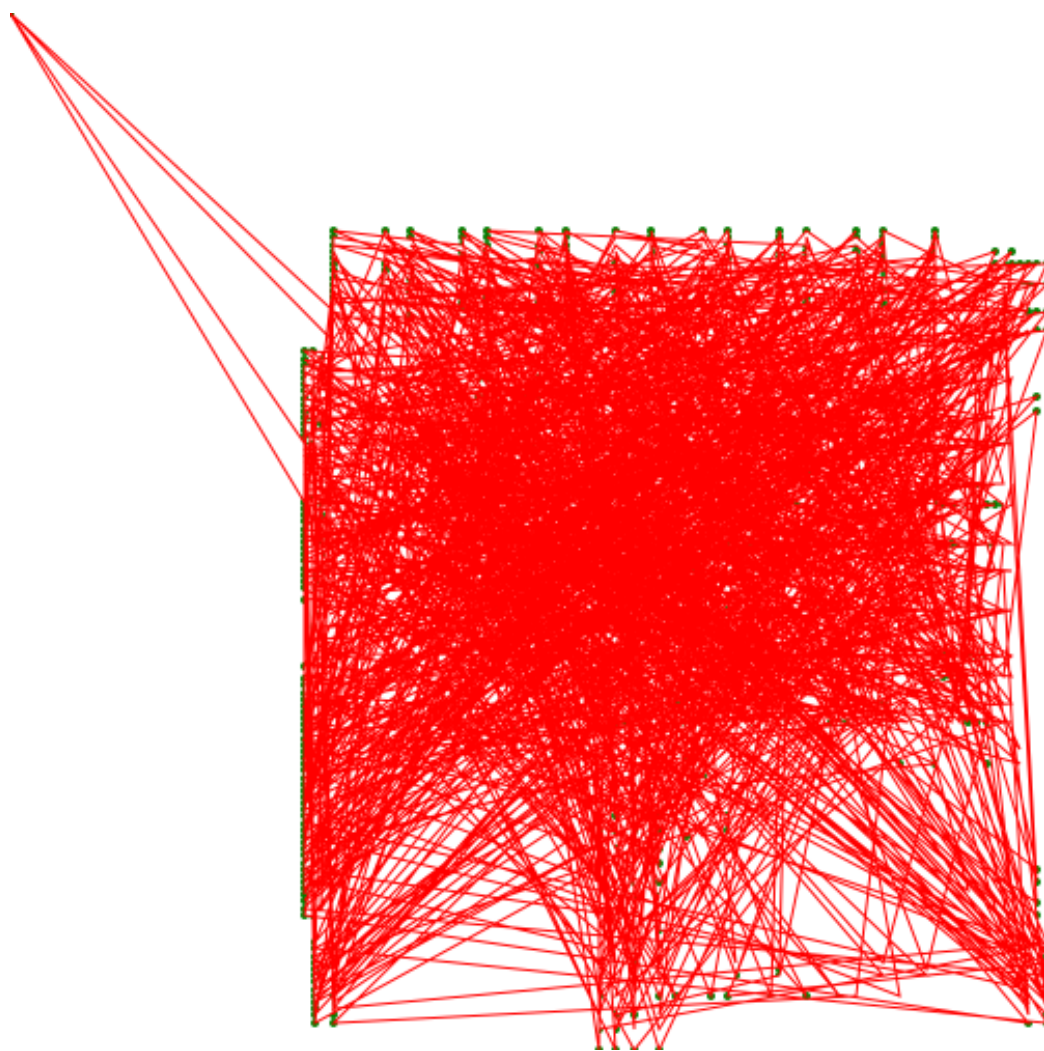
2020-04-21 Iditarod: Initiate Galena
2020-04-21 Initiate Kaltag
2020-04-21 Initialize SVG file
2020-04-21 Add a circle to an existing SVG using binary output
2020-04-21 Print a test circle to an SVG
2020-04-21 Print a test circle to a SVG file
2020-04-21 Set max lat and lons
2020-04-21 Draw rudimentary path for file
2020-04-22 Successfully complete path circuit through at least one file

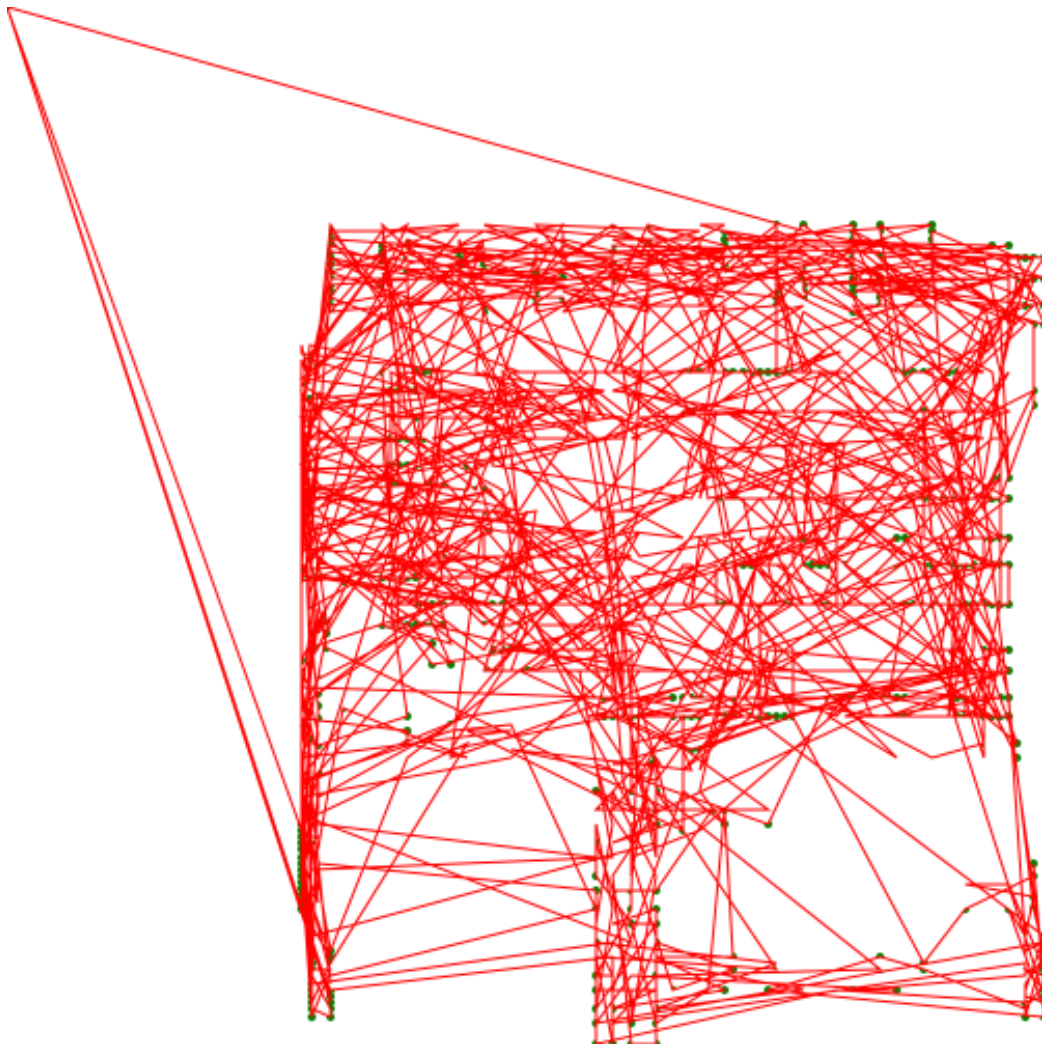
4 Sample Output

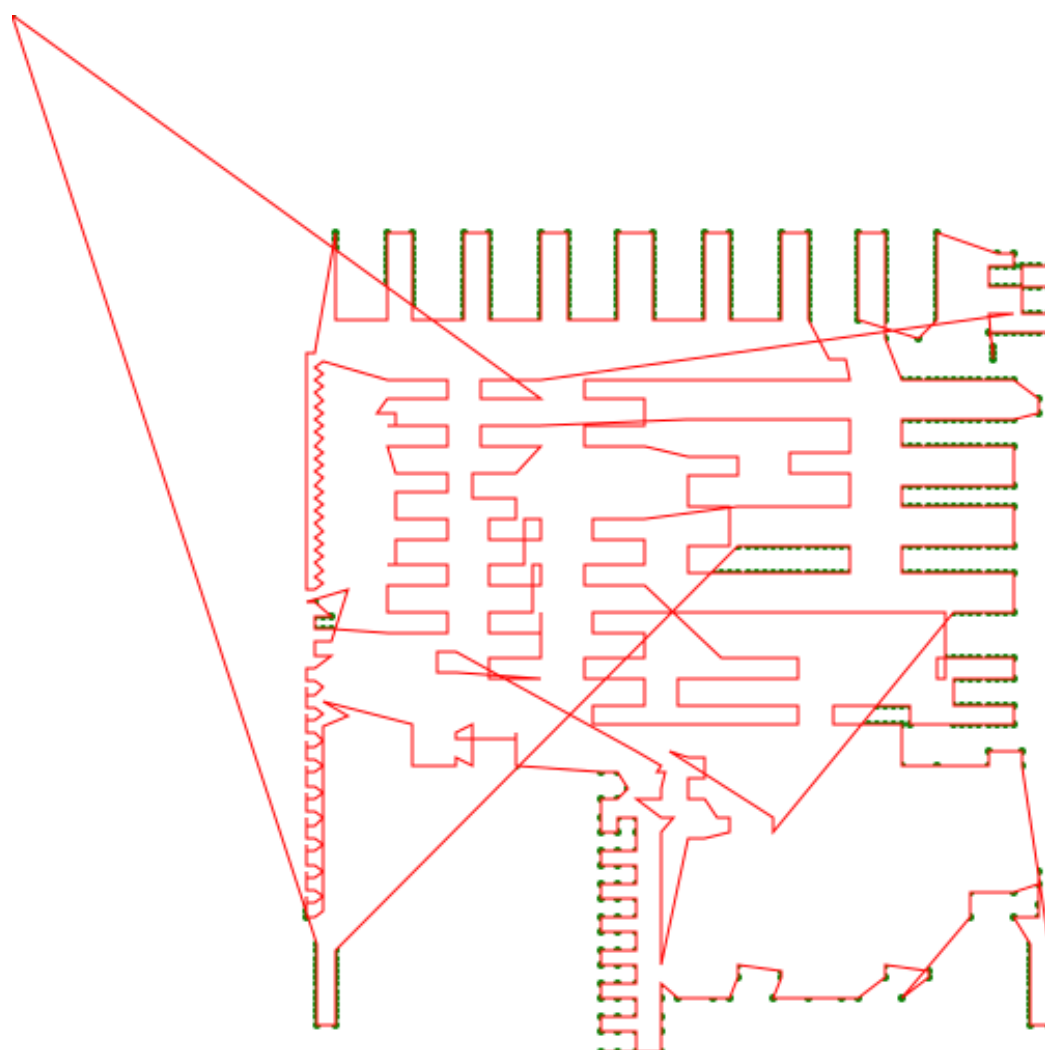
4.1 Kaltag

```
1 Printing SVG for: 567 and 536
2 Printing SVG for: 536 and 535
3 Printing SVG for: 535 and 508
4 Printing SVG for: 508 and 507
5 Printing SVG for: 507 and 506
6 Printing SVG for: 506 and 509
7 Printing SVG for: 509 and 534
8 Printing SVG for: 534 and 537
9 Printing SVG for: 537 and 566
10 Printing SVG for: 566 and 569
11 Printing SVG for: 569 and 590
12 Printing SVG for: 590 and 593
13 Printing SVG for: 593 and 1
14 Printing SVG for: 1 and 2
15 Printing SVG for: 2 and 1267
16 Printing SVG for: 1267 and 1268
17 Printing SVG for: 1268 and 1269
18 Printing SVG for: 1269 and 1270
19 Printing SVG for: 1270 and 1271
20 Printing SVG for: 1271 and 1272
21 Printing SVG for: 1272 and 1273
22 Printing SVG for: 1273 and 1274
23 Printing SVG for: 1274 and 1275
24 Printing SVG for: 1275 and 1276
25 Printing SVG for: 1276 and 1277
26 Printing SVG for: 1277 and 1278
27 Printing SVG for: 1278 and 1279
28 Printing SVG for: 1279 and 1280
29 Printing SVG for: 1280 and 1281
30 Printing SVG for: 1281 and 1282
31 Printing SVG for: 1282 and 1283
32 Printing SVG for: 1283 and 1284
33 Printing SVG for: 1284 and 1285
```

34 Printing SVG for: 1285 and 1286
35 Printing SVG for: 1286 and 1287
36 Printing SVG for: 1287 and 1288
37 Printing SVG for: 1288 and 1289
38 Printing SVG for: 1289 and 1290
39 Printing SVG for: 1290 and 1291
40 Printing SVG for: 1291 and 872
41 Best Distance for SolveGreedy: 62838.5







5 My Programs

5.1 SVG main.cpp

```
1  /*
2   * main.cpp
3   * CS202
4   * April 22, 2020
5   * Bryan Beus
6   * Kaltag station for Iditarod Challenge
7   */
8
9  #include <iomanip>
10 #include <vector>
11 #include <string>
12 #include <iostream>
13 #include <fstream>
14 #include <filesystem>
15 #include <stdlib.h>
16 #include <memory>
17 // #define BOOST_NO_CXX11_SCOPED_ENUMS
18 // #include <boost/thread.hpp>
19 // #undef BOOST_NO_CXX11_SCOPED_ENUMS
20
21 #include "Takotna.hpp"
22 #include "CityNode.hpp"
23 #include "CityPath.hpp"
24 #include "CityList.hpp"
25 #include "Miscellaneous.hpp"
26 #include "TspSolver.hpp"
27
28 using std::cin;
29 using std::cout;
30 using std::to_string;
31 using std::endl;
32 using std::vector;
33 using std::string;
34 using std::ofstream;
35 using std::ifstream;
36 using std::stringstream;
37 using std::pair;
38 using std::make_pair;
39 using std::setw;
40 using std::right;
41 using std::left;
42
43 namespace fs = std::filesystem;
44 // using namespace boost;
45 // using namespace boost::this_thread;
46
```

```

47 int main() {
48     clearConsole();
49
50     vector<string> fileNames;
51     callFileNames(fileNames);
52     vector<CityList> citylists;
53
54     // Parse File
55     // for (size_t i = 0; i < fileNames.size(); i++) {
56     for (size_t i = 0; i < 1; i++) {
57         cout << "Parsing file: " << fileNames.at(i) << endl;
58         string file = "../big/" + fileNames.at(i);
59         ifstream fin(file);
60         if (!fin) {
61             cout << "Error loading file: " << file << endl;
62             exit(0);
63         }
64
65         CityList newList;
66         newList.parseFile(fin);
67         citylists.push_back(newList);
68     }
69
70     // SolveRandomly()
71     double bestDistanceRandom = 1000000000000;
72     string filepath = "../output_images/randomly";
73     fs::create_directory(filepath);
74     for (size_t i = 0; i < citylists.size(); i++) {
75         // for (size_t i = 0; i < 1; i++) {
76         CityPath citypath;
77         TspSolver tsp;
78         double randomDistance = tsp.SolveRandomly(citylists.at(i),
79             ↪ citypath, to_string(i));
80         if (bestDistanceRandom > randomDistance)
81             ↪ bestDistanceRandom = randomDistance;
82     }
83     cout << "Best Distance for SolveRandomly: " <<
84         ↪ bestDistanceRandom << endl;
85
86     // SolveMyWay()
87     double bestDistanceMyWay = 1000000000000;
88     filepath = "../output_images/my_way";
89     fs::create_directory(filepath);
90     for (size_t i = 0; i < citylists.size(); i++) {
91         // for (size_t i = 0; i < 1; i++) {
92         CityPath citypath;
93         TspSolver tsp;
94         double MyWayDistance = tsp.SolveMyWay(citylists.at(i),
95             ↪ citypath, to_string(i));
96         if (bestDistanceMyWay > MyWayDistance) bestDistanceMyWay
97             ↪ = MyWayDistance;

```

```

94     }
95     cout << "Best Distance for SolveMyWay: " << bestDistanceMyWay
96         << endl;
97     // SolveGreedy()
98     double bestDistanceGreedy = 1000000000000;
99     filepath = "./output_images/greedy";
100    fs::create_directory(filepath);
101    for (size_t i = 0; i < citylists.size(); i++) {
102        // for (size_t i = 0; i < 1; i++) {
103            CityPath citypath;
104            TspSolver tsp;
105            double greedyDistance = tsp.SolveGreedy(citylists.at(i),
106                << citypath, to_string(i));
107            if (bestDistanceGreedy > greedyDistance)
108                << bestDistanceGreedy = greedyDistance;
109        }
110    cout << "Best Distance for SolveGreedy: " <<
111        << bestDistanceGreedy << endl;
112    return 0;
113 }

```
