Clustering Social Networks Using Genetic Algorithms

Group Blue

Dinesh Acharya, Radu Homorozan, Kiril Kafadarov, Denis Rochau, Annu Thapa, Valentin Vasiliu

Backend: General

- globally controlled by a singleton GlobalBackendController class
- implements SocialClusteringService
- includes multiple fitness crossover and mutation functions
- entirely multithreaded
- multiple changes to the original design (detailed explanations follow)

Backend: Configuration

- implemented custom settings class ClusteringParams
- implemented ConfigurationManager class
  - read settings from a file
  - write settings to a file
  - pass settings directly
- contains all configurable parameters of the algorithm
- can also be set from the GUI

Backend: Graph Readers/Writers

- separate readers for vertex-pair-weight and movie-lens type graphs
- once graphs are parsed, they can be written in a simplified form to a text file
- movie-lens graph reader generates graph and computes edge weights using similarity, then stores them in a graph object

Backend: Movie-Lens Dataset

Reader (Derived)

- Data was read into unordered map of set of pairs
- User Id was seperately stored into sets
- Weight was computed using similarity and stored in Graph

Backend: Movie-lens Dataset

- **Weight was computed based on Similarity Index**

$$Sim_{(A,B)} = x := \frac{1}{|S(A) \cap S(B)|} \sum_{m \in S(A) \cap S(B)} [R(A,m) - R(B,m)]^2$$

$$w = \frac{1}{1 + \sqrt{x}}$$

$S(A)$:= Set of movies rated by node (aka. user) A

$R(A,m)$:= Rating given to movie m by user A

$|S(A) \cap S(B)|$:= Number of movies that are rated by both A and B

$Sim_{(A,B)}$:= Similarity between A and B

w := weight of the edge connecting nodes A and B

Backend: Crossover

- parents for next generation selected using tournament selection

- three types:
  - uniform
  - clusterwise
  - combined

Backend: Mutation

Four Classes:

- MutationEngine

Abstract class that provides interface for mutating the particular configuration with virtual mutate() function

- ExplorationMutation

inherited from MutationEngine; mutate clustering solution by choosing a new clusterId based on the maximum clusterId in the cluster

- ExplorationJoin

inherited from MutationEndine; implements joining mutation

- ExplorationSplit

inherited from MutationEngine;implements spliting mutation.

Backend: Fitness Functions

- three fitness functions:
  - MQAnalyzer
  - PerformanceAnalyzer
  - FitnessAnalyzer (combination of both)

Backend: ClusteringService

- Controls the execution of the algorithm
- Runs everything multithreaded, number of threads dependent on settings
- Algorithm can be stopped and resumed, population may be initialized or loaded
- Feeds results into a concurrent locking queue so that they can be safely fetched from frontend

Algorithm

1. Initializes Population
   Compute Fitness
   Sorts it

2. Check Conditions
   a. Phase Refinement
      i. refine it
      ii. crossover
   b. Phase Exploration
      i. crossover
      ii. mutate

3. Reevaluate Population
   Sort them
   Select them
   Goto to 2.

Changes to the Original Design

1. Removed the struct ConfigurationParams defined in SocialClusteringService.hpp and created its own file; added managing utility for it which is a proxy for the structure; added ability to load a configuration from a file.

2. Changed the folder structure of the project: “include” folder contains all header files, “src” folder contains all source files; the project is being built in the “build” folder (which is not committed).

3. Changed AbstractGraph class to be a non-template class; added separate Vertex class to simplify the implementation of this and other related classes; most provided header stubs were incompatible with the AbstractGraph<V> class and had to be modified, this we opted to modifying the graph class instead.

Changes to the Original Design

4. Renamed “Encoder” classes to “Encoding” classes; makes sense for them to be internally holding a solution and performing utility operations on it instead of just the latter.

5. Removed “compiler_defs.hpp” file as the defines were unnecessary and not needed by most of our classes

6. Simplified the convoluted namespace to a simpler model; we are using the following namespaces:
   The main namespace clusterer (main namespace for the project);
   three sub namespaces and three aliases to clusterer::subnamespace :
   a. common - all modules and classes with common functionality - alias clc
   b. backend - all modules and classes with backend functionlity - alias clb
   c. frontend - all modules and classes with frontend functionality - alias clf

Changes to the Original Design

7. Replaced the simple and incomplete logger by a better one:
   - new logger replaces multiple log function with single veradiac one
   - logging is on a separate thread (useful as logging might be costly)
   - design is policy based (easy to exchange and extend logger)
   - we provide a singleton threadsafe access to our logger class

8. Created a new base class for all analyzers called ClusteringSolutionAnalyzer, as FitnessAnalyzer cannot be the base class:
   - added new parameter to the analyze function as we need to pass the graph
   - changed first parameter from the encoding representation to the class representing the encoding
   - removed the score typedef as the classes are not always reporting a score of a solution, but sometimes just certain measures (i.e. coverage)

Changes to the Original Design

9. Changed returnValue from readFile of GraphReader from int to bool; renamed GraphReader to AbstractGraphReader and implemented a normal GraphReader; added test_files folder to simulate reading the data from files and storing it into a graph

10. Changed the signature of CrossoverEngine::crossover to be using the ClusterEncoding class instead of the underlying data structure ClusterEncoding::Encoding; changed the method to void with a parameter child, as the abstract ClusterEncoding class cannot be instantiated

11. Completely redesigned SocialClusteringService as it was incoherent with the design and our changes; renamed it to ClusteringService; created a singleton class as an access point for client applications around it.

Changes to the Original Design

12. Added new classes that were necessary to implement the algorithm

13. Removed the class FlagParser as it did not make sense in our design

14. Removed the incomplete GeneticStrategy abstract class and used a simple class TwoPhaseStrategy as our genetic strategy

15. Removed the Person related classes as they are useless in our design (the idea of storing data in a database was an extra feature)

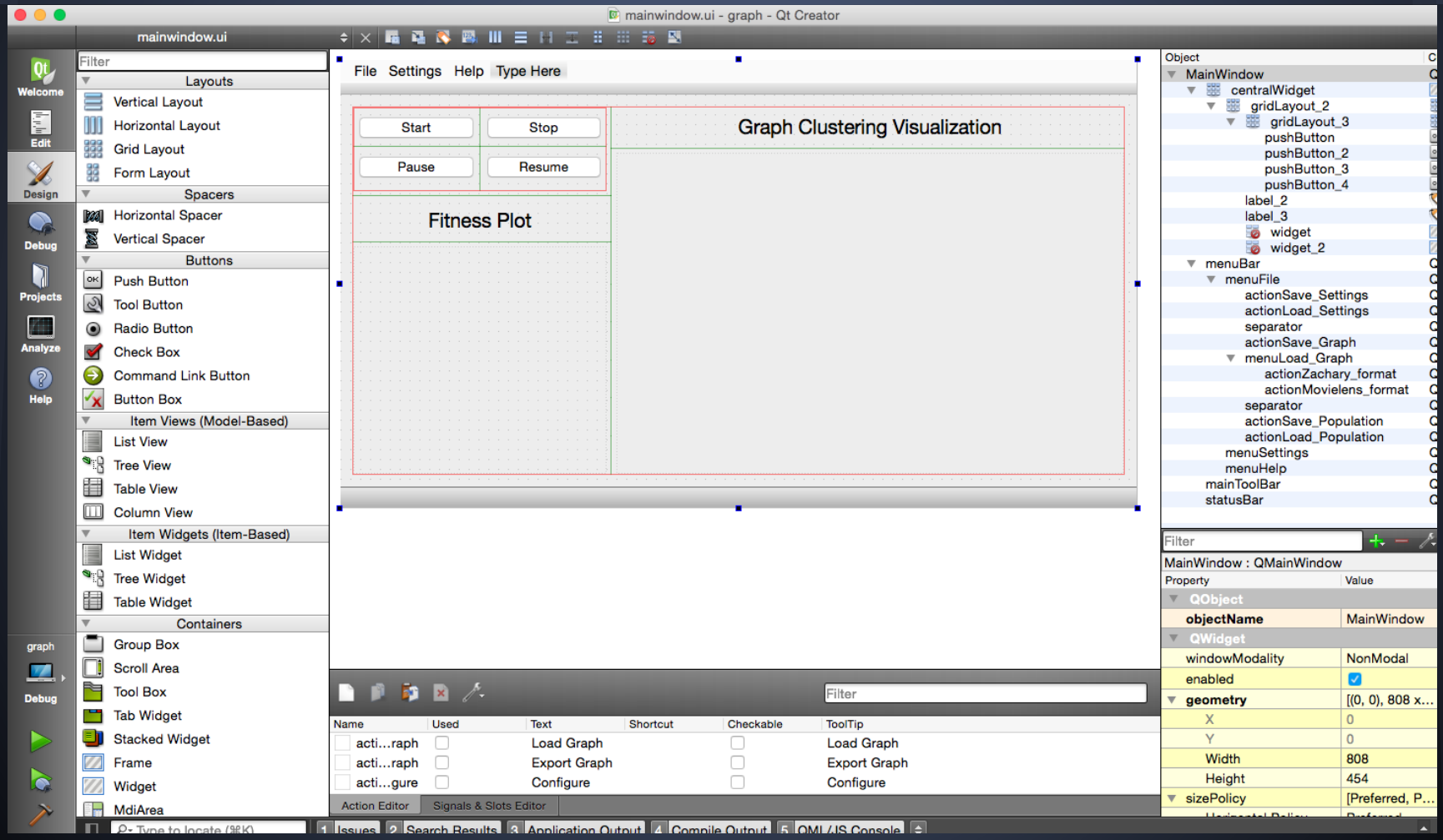
Frontend: General

- implemented using Qt, cross-platform native interface
- generated code with Qt Creator
- incorporated auto-generated code in our project
- manually added custom widgets to plot the data
- controls connected to the backend

Major challenges:

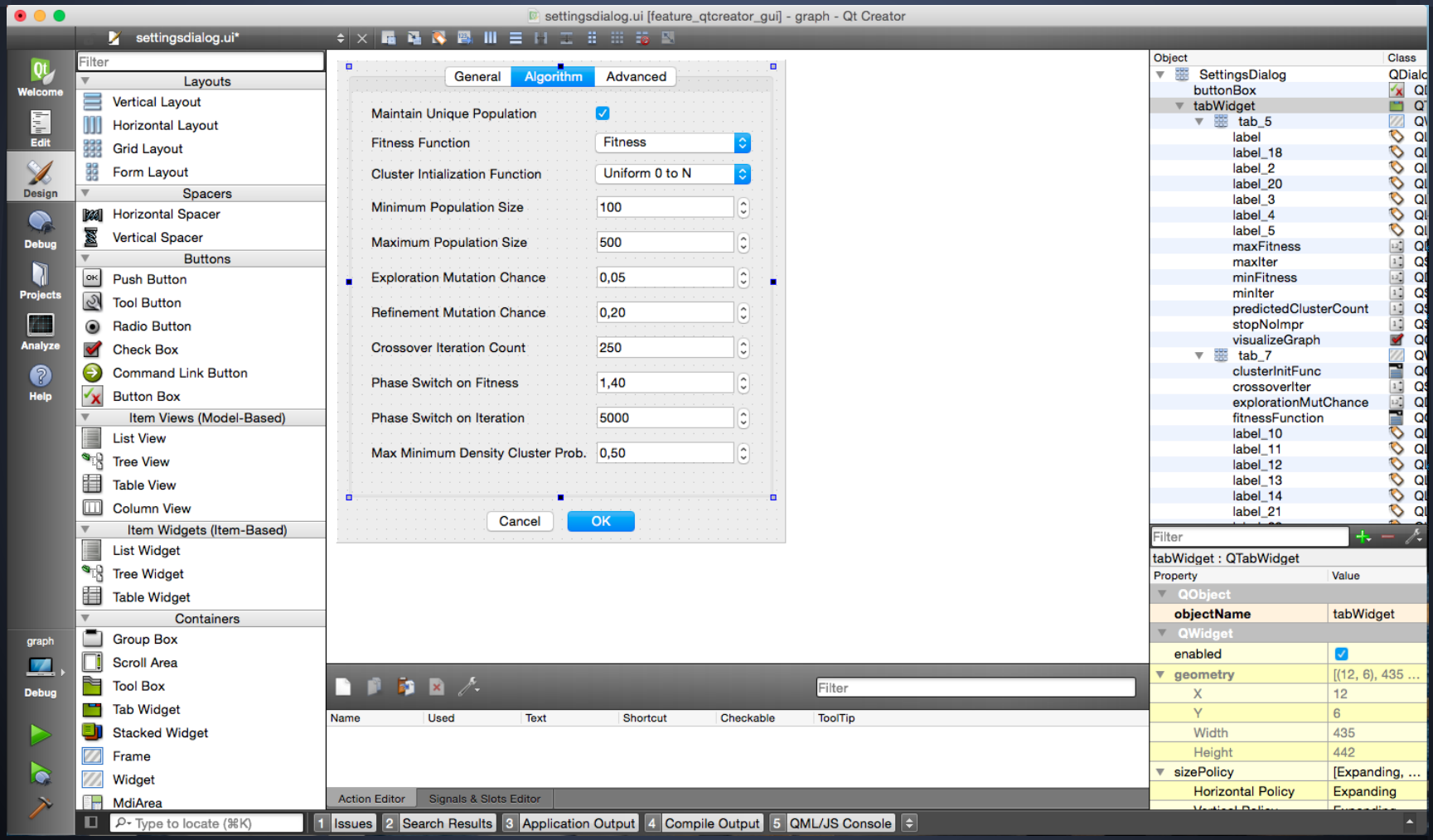
- figuring out the layout
- compiling the auto-generated code with CMake
- fitting qwt custom widgets in the placeholder slots

Frontend: Creating the GUI



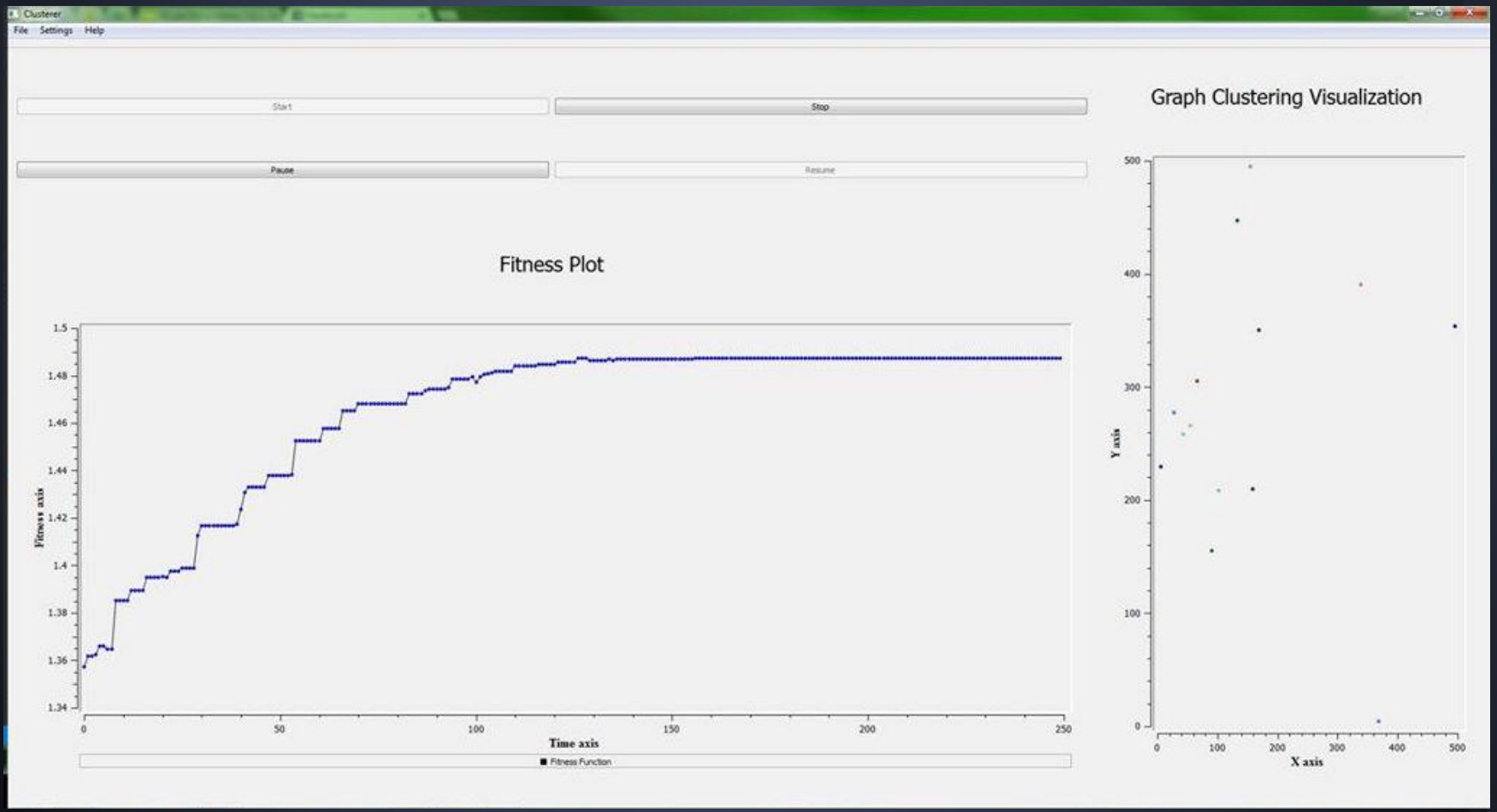
Creating the GUI in Qt Creator

Frontend: Creating the GUI

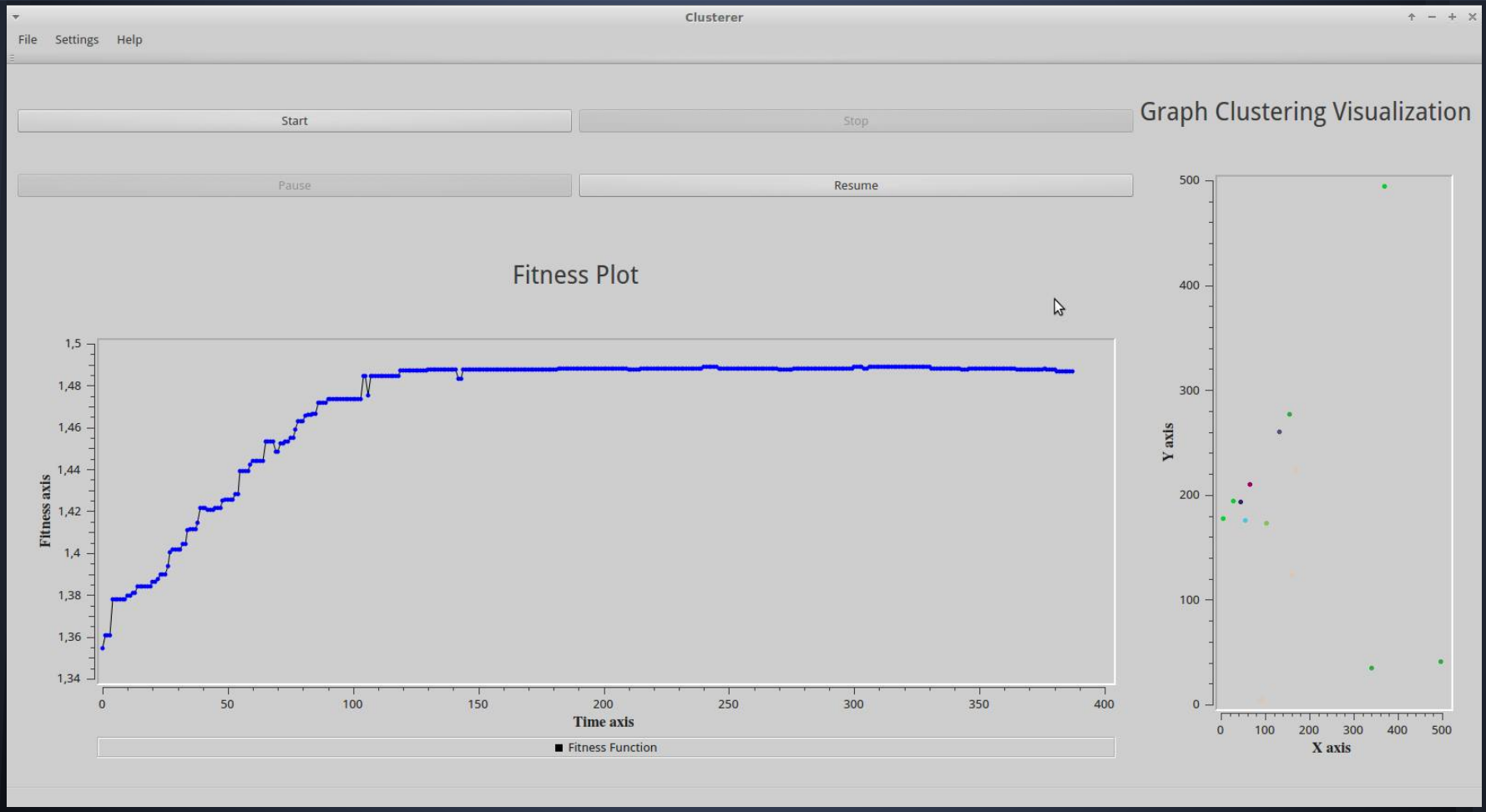


Settings dialog in Qt Creator

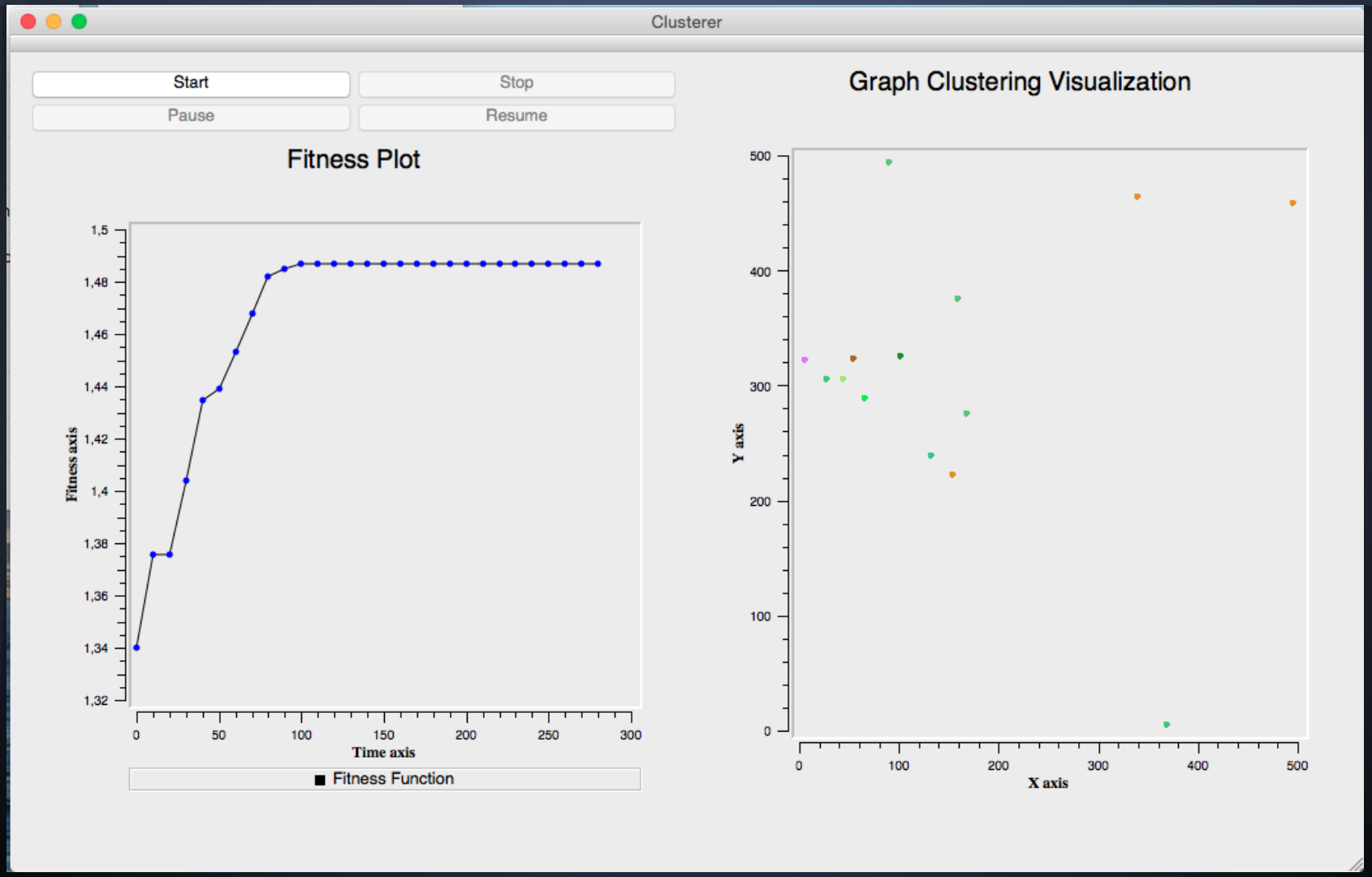
Frontend: UI on Windows



Frontend: UI on Linux



Frontend: UI on Mac OS



Frontend: Visualization

- Fitness Analyzer: the plot updates in real time as the algorithm running in the backend supplies fitness values to it
- Graph Visualization: the graph encoding of the data is processed and transformed via the MDS algorithm into a set of coordinates. Random symbols are set for each cluster at initialization and updated in real time as new solutions are provided by the running algorithm

Compilation and portability

- Compiled using cmake
- Fully portable, working on Ubuntu, Mac OS and Windows
- The project depends on Boost, Qt, QWT, and Armadillo (or Eigen on Windows)

Performance Tuning

- Cluster Refinement
- Maintaining unique population
- Initializing with a suggested number of clusters
- Swapping between different fitness functions
- Running on a variable number of threads

Movie-lens Dataset: Results

- Visualization not feasible (graph is too big)
- Algorithm was left to run for 24 hours, with one iteration taking about 10 minutes
- The produced results were stored in a log file

07/05/2015 01:18:18 : INFO : [ALG] Iteration: 0 Current maximal fitness: 0.309152
07/05/2015 03:24:23 : INFO : [ALG] Iteration: 10 Current maximal fitness: 0.340581
07/05/2015 07:36:26 : INFO : [ALG] Iteration: 30 Current maximal fitness: 0.375143
07/05/2015 10:45:23 : INFO : [ALG] Iteration: 45 Current maximal fitness: 0.390364
07/05/2015 13:54:06 : INFO : [ALG] Iteration: 60 Current maximal fitness: 0.398732
07/05/2015 15:59:44 : INFO : [ALG] Iteration: 70 Current maximal fitness: 0.405011
07/05/2015 18:05:45 : INFO : [ALG] Iteration: 80 Current maximal fitness: 0.407096
07/05/2015 22:04:52 : INFO : [ALG] Iteration: 99 Current maximal fitness: 0.412487

Movielens: Results

07/05/2015 22:04:52 : INFO : [ALG] Iteration: 99 Current maximal fitness: 0.412487

07/05/2015 22:04:55 : INFO : Current population:

0 1 2 3 4 5 6 5 6 7 7 8 8 8 9 9 8 10 11 11 11 11 12 12 12 9 12 13 14 13 13 9 15 14 15 12 15 16 16 9 17 17 15 16 18 15 19 19 19 19 18 19 14 17 19 20 20 21 22 20 18 16 23 17 18 24 22 25 22 24 25 25 23 26 27 26 27 28 27 20 24 21 29 27 24 30 29 29 31 26 26 30 32 31 29 30 29 32 33 34 35 19 30 23 0 29 30 34 36 37 34 32 36 34 32 38 37 38 35 38 12 39 39 35 32 33 40 41 34 42 43 33 31 40 44 40 41 45 45 29 41 37 45 44 44 46 45 47 47 15 30 47 46 46 48 48 49 47 45 45 40 49 40 50 50 50 50 51 50 49 49 48 39 44 10 35 52 53 51 53 22 52 46 47 54 53 55 6 56 42 55 18 52 50 54 1 57 45 53 52 58 8 55 57 58 56 50 59 60 55 61 61 62 54 60 35 31 61 32 61 63 59 59 48 62 64 63 65 62 66 54 67 29 63 64 62 35 59 68 58 67 69 70 64 67 63 70 63 70 71 67 72 64 67 67 73 72 65 70 72 74 71 66 53 75 76 74 73 74 63 73 74 67 75 75 76 1 68 46 56 66 77 78 75 79 63 27 68 75 74 80 78 76 61 65 80 62 75 81 71 82 80 56 81 73 27 68 83 84 78 31 76 38 83 85 82 84 86 83 87 81 73 84 75 85 88 82 80 85 89 80 90 47 91 81 92 54 90 88 6 85 93 60 81 78 89 91 94 88 94 81 83 75 86 90 95 57 91 79 96 88 97 84 98 87 96 96 35 68 94 92 97 99 100 100 99 55 101 95 98 102 59 95 103 103 104 102 90 76 103 74 97 16 100 105 106 95 106 103 107 106 100 105 34 108 107 98 106 109 102 100 110 111 110 103 103 78 17 112 113 112 103 106 71 105 83 106 112 111 109 114 112 111 105 98 106 115 16 107 116 111 117 102 112 118 119 119 88 116 107 120 98 121 118 117 112 78 116 89 109 119 109 122 109 119 123 121 115 118 109 120 107 122 124 125 126 127 119 125 72 54 121 116 113 125 124 104 20 124 128 122 129 116 100 121 124 101 21 124 125 130 131 132 116 133 121 131 131 112 134 21 135 130 133 130 42 136 16 129 122 137 135 1 102 121 125 138 16 126 107 121 139 136 140 135 135 133 22 124 26 134 141 67 139 136 134 139 142 143 137 129 144 145 125 142 129 146 143 147 135 148 149 143 150 150 151 142 146 134 147 152 147 143 149 150 104 146 40 139 141 110 122 153 68 150 136 154 149 73 131 101 122 155 142 117 148 156 153 148 148 152 73 137 133 154 136 127 144 143 112 83 155 154 152 149 112 157 158 96 136 159 157 160 152 145 159 144 161 158 155 135 157 162 163 136 164 160 60 165 166 155 65 158 162 129 36 167 72 158 162 145 136 152 163 164 145 165 168 148 166 106 166 90 169 157 144 169 148 156 163 35 123 170 171 172 173 153 172 123 171 173 170 174 168 157 7 150 146 175 157 46 156 172 148 77 170 160 163 48 164 176 171 113 43 175 177 165 166 140 177 94 164 178 171 14 177 168 145 179 166 180 176 128 175 180 43 179 167 173 3 170 178 181 179 59 182 179 183 184 185 186 154 138 187 100 137 168 188 170 183 46 182 188 5 189 183 190 181 179 155 170 156 87 188 191 142 185 97 185 192 189 193 189 190 141 191 191 184 184 4 192 183 172 194 191 195 196 197 190 189 193 189 10 174 192 170 191 198 198 94 129 190 199 200 24 171 195 197 182 161 192 184 201 196 200 114 185 186 153 193 202 184 196 203 190 189 0 194 204 198 190 128 195 205 119 153 197 115 198 197 206 177 141 68 181 122 69 198 202 4 205 62 141 166 186 207 205 14 90 196 189 208 38 207 207 207 209 210 63 211 212 210 213 214 198 206 73 154 204 83 215 204 55 216 197 154 217 126 206 209 94 218 206 214 17 166 209 206 209 205 219 220 214 209 81 216 143 180 180 218 209 221 212 222 59 198 223 215 220 207 187 218 142 138 128 96 216 185 222 219 212 217 211 208 219 96 192 213 206 171 218 196 220

MovieLens: Results Optimized

Running at the moment in the background. If we look at the current autosaved population we see that the right settings lead to good results fast.