Map Generation and Traversal with Dynamic Point Costs and Realistic Conditions

Computer Systems Tech Lab 2006-2007
by Olex Ponomarenko

Abstract

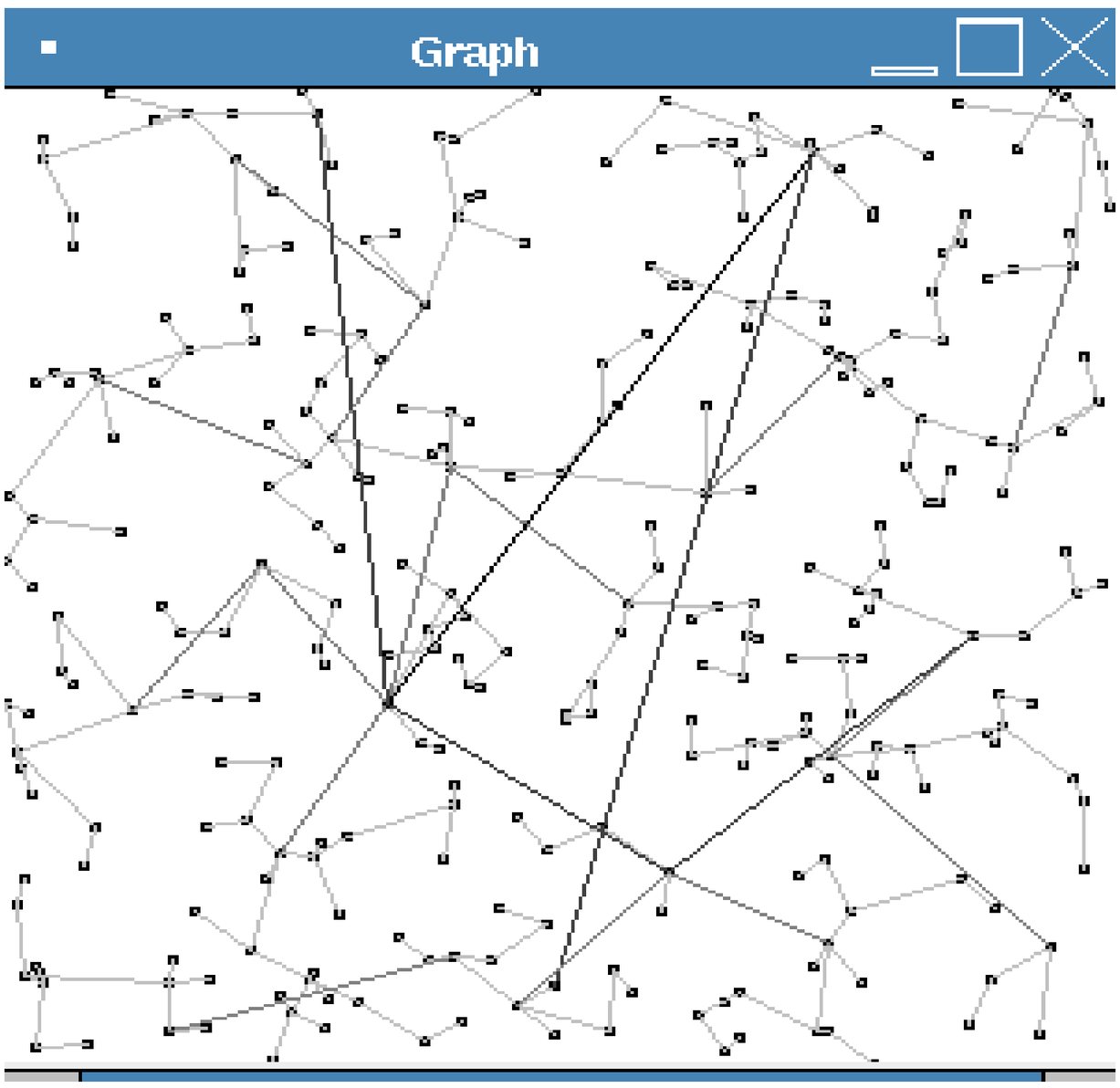
An abstract representation of a map was used to incorporate realistic conditions not currently in place in commercial path finding programs. As the project is focused on the back-end of random graph generation and searching, it can be useful for other projects as an educational tool about efficiency and memory usage, as well as more sophisticated front-end systems.

The project incorporates more realistic aspects of traffic movement such as traffic light and stop sign delays in order to provide realistic delay calculation when traversing a map. The map is generated using a sophisticated random graph generator, which was also developed as part of this project.

The entire project is also designed to be easily testable, flexible, and scalable. All of the parts of the project, from the shell to the final heuristic will be designed and created with a large-scale problem in mind.

Figure

The figure below shows a simple graph of 300x300 pixels with 280 locations. The road size scales from light gray to black (light gray being residential roads, black being the equivalent of an interstate highway) For testing purposes, I worked with larger, 800x800 pixel maps for additional data.



Background

Navigating a map has been a basic Artificial Intelligence problem for a number of years. Sites such as Google Maps and MapQuest are common examples of map traversal. These programs rarely consider traffic factors other than speed limits and come up with paths with a high number of turns that would in reality be slower than a less complex path, simply due to time spent on traffic lights and/or stop signs.

Usually such commercial programs simply use speed limits multiplied by the distance of the road to approximate travel time in their heuristics. On smaller roads, this may vary from the actual travel time on those roads. The main cause of this is the failure to consider traffic lights and stop signs (not to mention traffic congestion). With added factors such as traffic lights and stop signs, the program avoids such potential flaws of Google Maps and similar programs – complicated series of smaller roads leading to the destination.

Methods

The Python language is used for both map generation, map traversal, and cost calculations. Java is used to display the graph, in a similar manner to the figure on the right.

The generator is designed so that all locations connect to all other locations in some way. Road connections are realistic in the sense that small driveways do not have an exit onto a highway passing above. Connections between small roads are stop signs, and scale up to traffic lights for average roads and then exits for highways. Advantages and disadvantages based on direction are also in place; for instance, one-way stop signs and traffic lights favored to a larger road.

Results

The more complex model with point costs constructed in the project provided a faster path only in cases where a lot of turns were required to get to the destination, and the difference in travel time between a conventional search algorithm and one with point costs was on average 5.6 percent.

Since, the more complex method takes up approximately three times the space and performs $2\log_2(n)$ times slower than a conventional search algorithm, this approach is impractical if processing costs are considered.