

An Evolutionary Approach to Timetabling

Dieter Brehm & Elias Gabriel

2 May 2020

An Evolutionary Approach to Timetabling

by Elias Gabriel and Dieter Brehm

Background

Evolutionary algorithms are different from raw implementations of solutions, and are a subset of optimization problems that derive strategies from real world speciation and evolution. As opposed to building globally-optimal strategies or combining locally-optimal ones, evolutionary algorithms adjust and adapt their current solutions over many generations to best fit some pre-defined desired goal. The process is intended to follow evolutionary theory, where a species adapts its gene pool over time to best match environmental conditions and maximize likelihood of survival.

Genetic algorithms phrase solution-finding in terms of those natural adaptive strategies, like gene mutation and natural selection. The solutions, which are all progressive generations on some defined common ancestor, are ranked based on their position in some “fitness landscape.” In the nature, the fitness function is most often the likelihood to survive and procreate. In the case of homebrew algorithms that fitness could be anything, like “how far a robot moves before falling.”

Problem Space

We chose to study the details and implementation surrounding home scheduling

We implemented a simple genetic homework scheduling system which assists students in optimizing when to study. It can be difficult to balance NINJA hours, class due dates, and lengthy assignments, but using a heuristic-based schedule in the context of time, we will try and give suggestions for how to improve the studying experience. Parameters to our weighting system could include due dates for specific homework, time entry from the users, a user’s current grade on canvas, when class is happening, ninja-hours optimization, and desired sleeping hours.

We see parallels to the knapsack and other pathfinding problems, and also in concepts used in CPU scheduling.

References

An overview of the A* algorithm: <https://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html>. This will help build our foundational knowledge on heuristic-based path finding.

A heuristic-based approach to optimal path finding: <https://theory.stanford.edu/~amitp/GameProgramming/Heuristics.html>. We need to know about heuristics in-depth in order to figure out how we may be able to adapt it to our own algorithm.

A comprehensive overview of heuristic functions: <https://medium.com/@rinu.gour123/heuristic-search-in-artificial-intelligence-python-3087ecfece4d>. Algorithms like A* do not guarantee that they'll find the globally best solution, but do guarantee that they'll find the best solution for the given heuristic functions. Therefore, resources like these will be critical in learning more about how to select better and more optimal functions.