Emergent Computing for Optimisation Coursework

Introduction

* Decided to modify both the pacing and transition in order to have the most things to vary to improve the score.
* Modified selection, crossover, mutation methods
* Roulette selection, uniform crossover, multipoint crossover, creep mutation, scramble mutation.
* Implemented a hillclimber which can stop worse children from being added to the population.
* Added diversity to the population through the sawtooth diversification method.

Method

* Created solution by creating new methods of mutation, crossover and selection.
* Used base fitness function provided (explain what it does anyway)
* Used varying combinations of my crossover, selection and mutation algorithms as the EA algorithm.
* Hillclimber to only include children better than the previous generation.
* Sawtooth to help diversify the population of each generation.
* Systematic exploration of the EA parameters testing the results with different sized mutation rates, crossover rates and population/number of generations.

Experiments and analysis

* Tested several configurations a number of times to provide accurate results (Table)
* Calculated mean score averages for each configuration. (Graph/table)
* Tested different configurations of EA parameters on one algorithm configuration to see the change in performance changing these values has.
* Comment on how becoming stuck in local optima is very common in several configurations (hillclimber can mitigate this a bit)
* Roulette improved speed of algorithm tremendously but the end score generated was worse than tournament.
* Diversity mechanism vs no diversity mechanism (does it affect results?)
* Compare my score to starting score. Tracktrat
* Tested using two children instead of one but concluded that the results were not affected enough by the improvement to justify its inclusion as it slowed the algorithm a bit.

Conclusions

* Best race time achieved with spec that achieved it.
* Comment on the successful parts of my algorithm (Mutation methods I implemented, hillclimber, crossover methods)
* Comment on unsuccessful parts(roulette not providing a good result in comparison to tournament, initialisation could have been altered to improve starting score (potentially reduce time to find a good score.)
* Comment that implementing new mutation and crossover methods for transition had very little effect when compared to the pacing strategy (pacing had more possible values to experiment whereas transition only had two possible values).

Future work

* Create new initialisation function to provide better starting values.
* Implement a distributed approach e.g. island to create a more diverse population and hopefully improve a better score.
* Test other mutation/selection/crossover methods to see if the results improve.
* Possibly add a hyper heuristic such as tabu search to select between several heuristics