Jacob Hopkins

10/6/2020

Project 3 Progress Report

* **BIN-PACKING:** Given a set of *B* bins, each with capacity *b*, and a set of *n* packages, each with positive size *si* in [1,*b*/2], assign packages to the minimum number of bins without allowing the total size of the packages in any bin to exceed b.

**GA Design Choices**

* Chromosome Representation
  + Unlike traditional k array genetic algorithms bin packing problems tend to be hybrid grouping genetic algorithms.

This means a two-dimensional array:

Each list in the array represents each bin.

If we had per say 4 empty bins we would have:

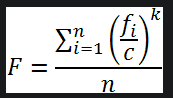
{ [], [], [], [] }

Item count is user inputted, but suppose fitting 10 items into bins would be such:  
{ [1,2,3], [4,5,6], [7,8,9], [10] }

Where a parallel array contains the corresponding sizes generated randomly by 1 to b/2

Capacity per bin, user inputted: 50

Weights {10,5,6,8,2,16,8,19,12,25}

* Fitness Evaluation
  + 
  + F = fitness
  + N = bin count
  + Fi = fill of the i-th bin
  + C = capacity of the bin
  + K = number > 1. 2 for default and higher is more even bin fill
  + Maximise this value
* Selection
  + Pass the top 10% of population in fitness and crossover the rest into existance
* Crossover Operation
  + Select two points in the chromosome and swap those sections. Find points that are duplicates and remove them from all bins they are in and reassign them to bins using a first fit or something similar heuristic. If you cannot fit it then create a new bin
* Mutation Operation
  + Remove a bin random. Reassign them using the first fit heuristic similar to above
  + https://www.codeproject.com/Articles/633133/ga-bin-packing