Offline One-dimensional Bin Packing using a Genetic Algorithm

Quinton Weenink
dept. Computer Science
University of Pretoria
Pretoria, South Africa
u13176545@tuks.co.za

I. Introduction

This report aims to investigate solving the offline one-dimensional bin packing problems [1] using a *genetic algorithm* (GA). This involves choosing a fitness function, correct selection method and genetic operators which suite the problem.

II. Background

Bin packing is a problem which involves placing items with weights w into the least amount of bins. Each bin can only store up to its bin capacity c. This problem can therefor be seen as a minimisation problem, rearranging the items to reduce the number of bins required.

A. Chromosome

The problem map below maps a chromosone to the weights of the associated item:

map	0	1	2	3	4	5
weights	3	2	5	1	9	7

A chromosone represents a specific collection of weights:

chromosome	4	2	1	5	0	3
corresponding weights	9	5	2	7	3	1

The above chromosome results in the bellow representation its fitness:

b	w_f		space
1	9		1
2	5	2	3
3	7		3
4	3	1	6

B. Fitness

As proposed in [2] just measuring bins required is not a good enough fitness evaluation function when provided with a more difficult problem resulting in a rough fitness landscape. This prevents the best solutions from being found.

Below is a fitness function suggested in [2] which helps prevent this.

$$FF = \sum_{i=1}^{nb} \left(\frac{F_i}{c} \right)^k \tag{1}$$

Where:

- FF is the fitness function
- nd is the number of bins
- F_i is the sum of weights for bin i
- c is the bin capacity
- k is a heuristic exponential factor set to 4 as suggested in [2]

This fitness function coverts the problem to a maximisation problem requiring the arrangement of the items in order to maximise FF.

C. Selection method

Tournament selection was used to select individuals. Reverse tournament selection was used to determine which individuals will be replaced. A tournament size of 3 was use throught this report in order to have a low selective pressure.

D. Crossover

Parents are selected using tournament selection and a random point of the chromosome is copied over to the children. This is repeated randomly in range [1, 3].

parent 1:	4	2	1	5	0	3
parent 2:	5	0	4	2	3	1

child 1:	4	0	1	5	2	3
child 2:	5	2	4	0	3	1

For this report a crossover rate of 100% was used. If two parents that have the same chromosome were selected then mutation is applied to generate a child that is different. This was done in order to prevent the population from containing only one individual.

E. Mutation

An individual was selected and weights at random indexes in the chromosome are swapped. This again is repeated randomly for each mutation in the range [1, 3].

parent 1:	4	2	1	5	0	3

child 1:	0	2	1	5	4	3

A mutation rate of 70% was used for all problems in this report.

F. Fitness landscape ruggedness measure

Below is a group of measurements that can be used to determine to some extent the ruggedness of the fitness landscape.

TABLE I
RESULT OF FF OVER 1000 SAMPLES. MEANS ARE REPORTED WITH STANDARD DEVIATIONS IN PARENTHESIS

problem	$mean$	std	max	min
N1C1W1_K	13.501622	(0.8957952)	16.853831	11.148543
N2C3W2_I	22.173831	(1.4157928)	26.403985	18.178036
N1W4B2R2	4.1555599	(0.1865943)	4.7666051	3.7694667
N2W1B1R4	21.273435	(1.1363297)	24.460151	17.692581
HARD5	35.279532	(1.0859528)	38.725135	32.064869
HARD6	36.023681	(1.0188546)	39.364814	32.837201

Problems with higher standard deviation could be said to have a more rugged landscape than those with lower standard deviation. It is however worth noting that this is not a good ruggedness measure as a slanted landscape could result in similar results as that of one with many peaks and valleys.

III. RESULTS

These results listed in table II are reported over a population of 100 individuals for 2 problems in each of the easy, medium and hard data sets in [1].

TABLE II
RESULTS AFTER 5000 GENERATIONS. MEANS ARE REPORTED OVER 10 SAMPLES WITH STANDARD DEVIATIONS IN PARENTHESIS

problem	m*	FF	nb	$best$
N1C1W1_K	26	20.894682478 (0.228596568)	26 (0.0)	26 10
N2C3W2_I	44	38.303106795 (0.524228967)	46.3 (0.640312423)	45 1
N1W4B2R2	6	4.9956391124 (0.009447657)	6.0 (0.0)	6 10
N2W1B1R4	34	29.786214322 (0.632602532)	35.1 (0.538516480)	34 1
HARD5	56	44.400740235 (0.711231525)	60.3 (0.458257569)	60 7
HARD6	57	29.786214322 (0.716193231)	61.3 (0.458257569)	61 7

Figures 1, 2, 3, 4, 5 and 6 represent the best solution fitness over time for the problem. All results were with 5000 generations sampled over 10 samples. The known best is plotted on the graph as an indication as to how close the algorithm is to the best possible solution. While only the fitness function is used the algorithm the bins required is plotted to reference the known minimal bins.

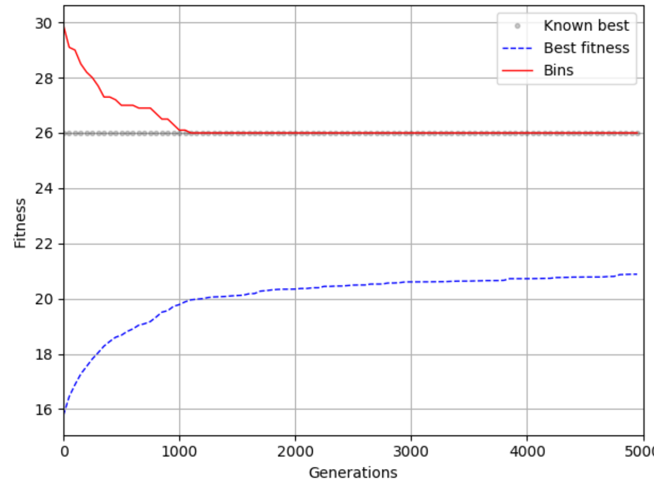

Fig. 1. problem N1C1W1_K fitness objectives over time

Figure (1) indicated that a minimal solution to the problem is found within 1200 generations. The GA came to a solution quickly.

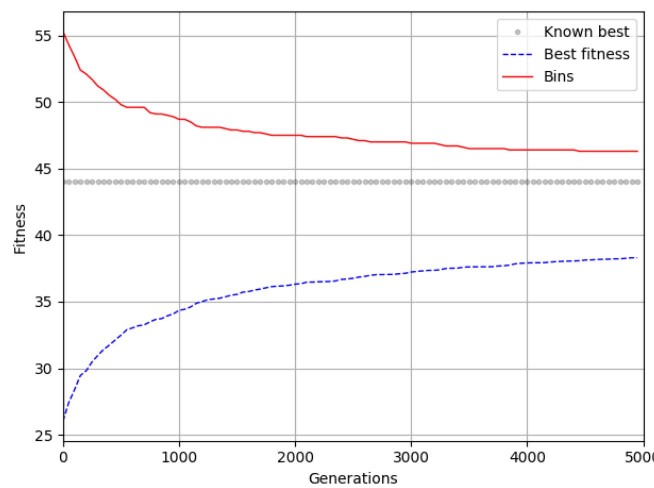

Fig. 2. problem N2C3W2_I fitness objectives over time

Even though the the problem in figure (2) is from the easy data set in [1] the algorithm did not come to a minimal solution as seen in table II. This indicated that some improvements need to be made to the GA.

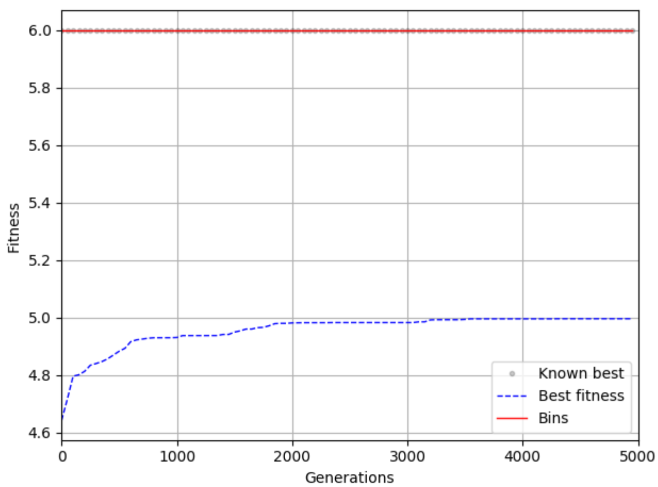

Fig. 3. problem N1W4B2R2 fitness objectives over time

The solution for figure (3) is immediately found meaning that there is most likely a large solution space.

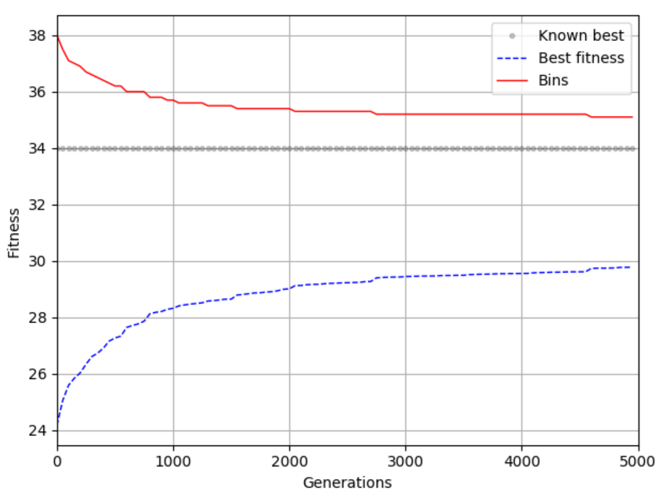

Fig. 4. problem N2W1B1R4 fitness objectives over time

The GA almost reach the minimal number of bins but only found a best solution once over the 10 samples as can be observed in figure (4)

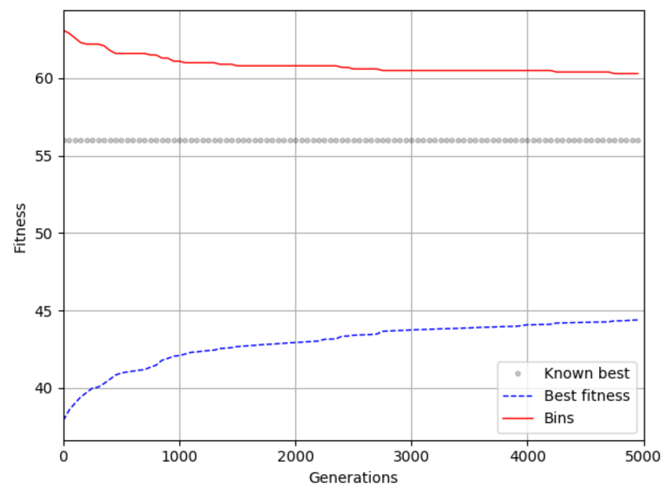

Fig. 5. problem HARD5 fitness objectives over time

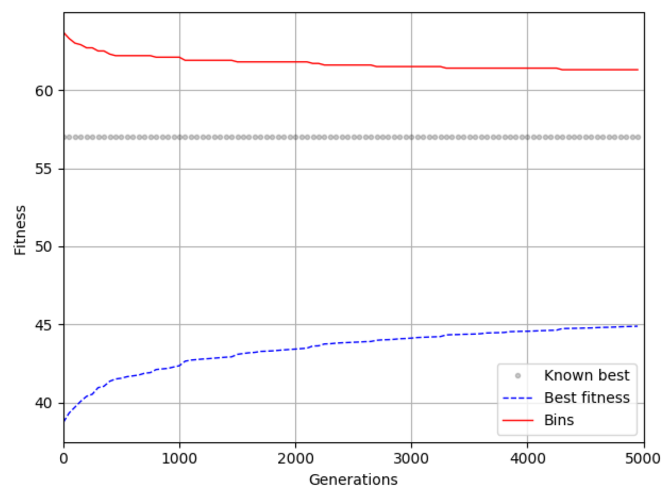

Fig. 6. problem HARD56 fitness objectives over time

While the algorithm did improve the fitness of the solution over time as can be seen in figure (5) and (6) in did not come close to the minimal number of bins for the hard problems.

IV. Conclusion

In conclusion the genetic algorithm struggled to solve the more difficult problems. Crossover could be improved as to more reliably keep the good parts of parents. The swap method which was implemented in this report could be switched out for a push method which could preserve the good parts of the chromosome while crossing over that which could be good of the other parent. With more study of the fitness landscape better crossover, mutation and selection methods could be chosen to increase exploration.

REFERENCES

[1] R. K. Armin Scholl, "one-dimensional bin packing problem datasets," 2013. [Online]. Available: https://www2.wiwi.uni-jena.de/Entscheidung/binpp/index.htm

[2] E. Falkenauer, "A new representation and operators for genetic algorithms applied to grouping problems," *Evolutionary Computation*, vol. 2, no. 2, pp. 123–144, 1994. [Online]. Available: https://doi.org/10.1162/evco.1994.2.2.123