Bin-Packing with an Evolutionary Algorithm

ABSTRACT

Bin Packing Problem (BPP) is a Combinatorial Optimization problem, which is used to find the optimal weight from a finite set of objects. This report reviews a general idea of BPP and the Evolutionary Algorithm (EA) and the various operators which are used to solve the BPP.

1 INTRODUCTION

The Bin Packing Problem (BPP) is a well-known NP-hard combinatorial optimization problem. In the BPP, the aim is to combine items into a predefined finite number of bins so as to minimize the difference of weights in each bin. I have used the Evolutionary algorithm with various components like varying population size, binary tournament as the parent selection technique and several variation operators like the recombination and mutation operators and mainly used the replacement of weakest solution as the survival selection mechanism. The BPP in the real world might be used to make efficient use of time and/or space. A bin for example, need not necessarily be a container, but could represent a space in time or a surface area. There are multiple variants of the EA, the generalized method of all the approaches can be defined by the below pseudo code

```
Begin
        1.Initialize the population with n no.of randomly generated candidate solutions
        2.Evaluate the fitness of each candidate
        DO
                1. Select parents (e.g. Binary Tournament method)
                2. Recombine parents (e.g. Crossover) to produce offspring
                3. Mutate the resulting Offsprings
                4. Evaluate the fitness of each offspring
                5. Replace individuals for next generation
                6. If termination condition is fulfilled exit else repeat steps 1 to 5
End
```

.2 EXPERIMENTAL AND COMPUTATIONAL DETAILS

2.1 Representation of Individuals

Two specific Bin Packing Problems have been addressed:

1. **BPP1**: In this method the number of items=500 and the number of bins (b) =10. The weight of an item **i =i/2**, where i starts at 1 and finishes at 500
2. **BPP2**: In this method the number of items=500 and the number of bins (b) =100. The weight of an item **i =*i*^2/2**, where i starts at 1 and finishes at 500

I have used the k-ary representation where a candidate solution (chromosome) is represented as a string of 500 numbers where each number ranges from 1 to b, The ith character in the string indicates which bin the weight is in.

2.2 Evaluation Function (Fitness)

For this problem the Evaluation or the fitness function is the difference of weight between the heaviest and lightest bins. The goal is to minimize the difference to optimize the weights in each bin.

2.3 Population

I have experimented with 10 and 100 randomly generated solutions as initial population. The fitness of each solution is computed as per the above Evaluation Function.

2.4 Parent Selection mechanism

Binary tournament method (with replacement) has been used, where two solutions from the population is chosen randomly and the fittest is chosen as the parent

2.5 Variation Operators

Crossover (Recombination): A random single crossover point(locus) is selected and the gene values between the parents are swapped only for the genes after the crossover point to create two off springs

Mutation: Multi-Gene Mutation operator denoted by Mk is used, where a gene(locus) is selected randomly and changed to a random new value between 1 to b, this is repeated k times. In my experiments I have used M1 and M5 operators.

2.6 Survivor Selector Mechanism (Replacement)

If the fitness of the children is better than the worst solutions in the population, we will replace them with the newly generated children.

2.7 Termination Condition

The algorithm runs once and stops when 10,000 fitness evaluations are reached, we call this a single trial. Each experiment underwent five trials

3 RESULTS AND DISCUSSION

All trials are seeded with random seeds (trail1: seed=10, trial2: seed=20, trial3: seed=30, trial4: seed=40, trial5: seed=50)

3.1 Five trials of the EA with crossover and operator M1 and population size 10

The below table contains the fitness achieved at the end of each trial for BPP1 & BPP2.

Trials	BPP1	BPP2
Trial1	16.5	79502
Trial2	12	68650
Trial3	18	66607
Trial4	5.5	73360
Trial5	7.5	71890.5

Table 1

The mean fitness for BPP1 is 11.9 and for BPP2 is 72001.9

3.2 Five trials of the EA with crossover and operator M1 and population size 100

The below table records the fitness after each trial for BBP1 & BPP2

Trials	BPP1	BPP2
Trial1	4	125104
Trial2	9	128379
Trial3	22.5	118185
Trial4	14.5	119052
Trial5	20.5	109680.5

Table 2

The mean fitness for BPP1 is 14.1 and for BPP2 is 120,080.1

3.3 Five trials of the EA with crossover and operator M5 and population size 10

The below table records the fitness after each trial for BBP1 & BPP2

Trials	BPP1	BPP2
Trial1	74	101888
Trial2	81	98811.5
Trial3	70.5	109981
Trial4	61	117394
Trial5	69.5	103919.5

Table 3

The mean fitness for BPP1 is 71.2 and for BPP2 is106,398.8

3.4 Five trials of the EA with crossover and operator M5 and population size 100

The below table records the fitness after each trial for BBP1 & BPP2

Trials	BPP1	BPP2
Trial1	110	172527
Trial2	83.5	140228
Trial3	114	157317
Trial4	135.5	152448
Trial5	64.5	146791

Table 4

The mean fitness for BPP1 is 101.5 and for BPP2 is 153,862.2

3.5 Five trials of the EA with operator M5 and population size 10

The below table records the fitness after each trial for BBP1 & BPP2

Trials	BPP1	BPP2
Trial1	76.5	93510.5
Trial2	86.5	105790
Trial3	49.5	102368
Trial4	68.5	101215.5
Trial5	78.5	92141

Table 5

The mean fitness for BPP1 is 71.9 and for BPP2 is 99005

3.6 Five trials of the EA with crossover and population size 10

The below table records the fitness after each trial for BBP1 & BPP2

Trials	BPP1	BPP2
Trial1	1009	480340.5
Trial2	1817.5	528959.5
Trial3	2047	440714.5
Trial4	2178.5	483291
Trial5	1023.5	470804

Table 6

The mean fitness for BPP1 is 1615.1 and for BPP2 is 480,821.9

4 ANALYSIS

The most fundamental process in Darwinian evolution is a population's exploration of an adaptive landscape by mutation and selection. As a population scales higher peaks in a landscape, it's mean fitness increase. Many factors influence this process. Among them is the structure of the landscape itself, the presence and incidence of recombination, the rate of mutations and population size. From Table1 & Table2 both for BPP1 and BPP2 the final fitness is better with population size 10 than population size 100. Figure 1 shows the fitness graph for both population sizes, since the aim here is to minimize the weights difference in the bins, i.e., the lower the fitness value the better the solution. Clearly the fitness of population size 10 is better. An adaptive evolution may be more rapid in large populations, larger populations produce more mutant individuals per generation, which helps explore more genotypes and find optimal genotypes faster than smaller populations. But for smaller population size the evolution may be more rapid and give a quicker convergence. In our case **population size 10 gave us better results** by converging rapidly to give an optimal solution.

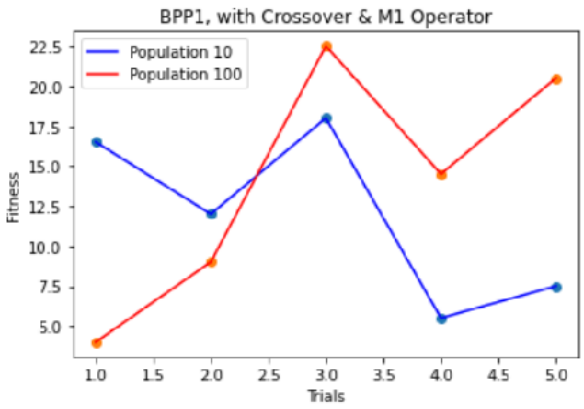

Figure 1

The recombination or crossover operator is inspired in the way in which the genetic code of one individual is inherited to its descendants in nature. By recombination the EA is more likely to create a better solution by making the population more diverse and thus immune to be trapped in a local optimum. Thus, **removal of crossover creates worse solutions** as seen in the below comparison in Figure 2.

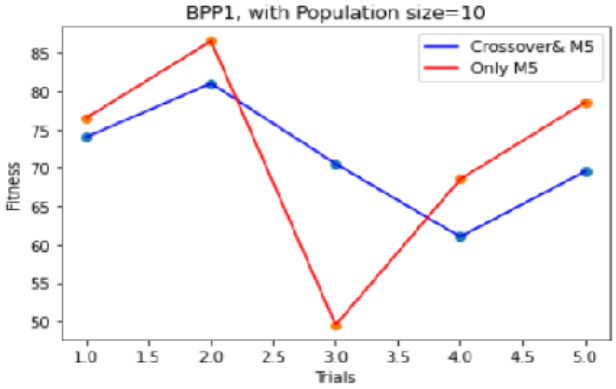

Figure 2

Mutation operators are used to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing, or even stopping convergence to global optimum. From tables 1 and 6 we can see clearly removing the mutation operator has a detrimental effect on the fitness. Thus, **removing the mutation operator again creates worse solutions** as the solutions tend to get stuck in a valley.

The influence of mutations and the distribution of their fitness can be categorized as deleterious, neutral, or beneficial. **Increasing the mutations creates many deleterious solutions** thus dampening the fitness of the population as shown in the below figure.

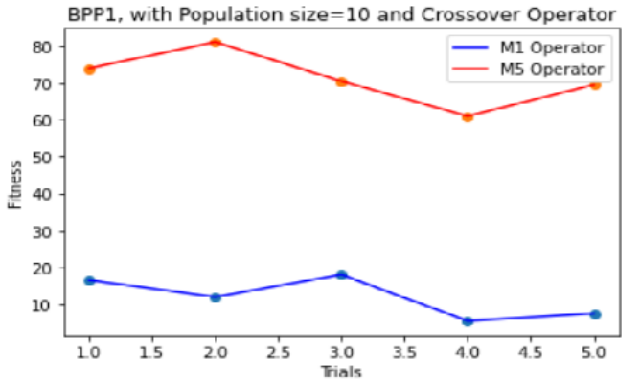

Figure 3

The parameters of EA depend on the specific problem, in general the best way to identify the best combinations of the different parameters would be to do a sensitivity analysis by carrying out multiple trials of the algorithm with different combinations and population size and compare the outcomes. From comparison of the results produced by the five trials of each combination, for our problem the **best results** were provided by using a **population** of **size 10** and using both **single point crossover and M1** mutation operator.

A small population size ensures rapid convergence to an optimal solution, the crossover and mutation increases the exploration of the solution space thus avoids solutions getting stuck in a local optimum. Although using multiple mutations can result in ineffective or harmful mutations which affect the overall fitness of the solution negatively.

4 FURTHER EXPERIMENTATIONS

4.1 Choosing the Fittest as parents

Always choosing the best quality solutions as parents does not produce best quality children. The whole search algorithm could become too greedy and get stuck in a local optimum. This is clearly visible in the below figure that compares the fitness of this approach with binary tournament method.

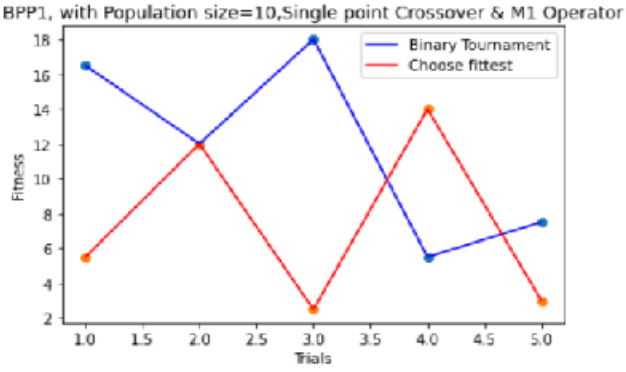

Figure4

4.2 Using Uniform recombination

Using uniform recombination produces better results as it diversifies the search space, explores the topology, and helps reaching better fitness. As we can see from the below figure comparing the fitness of uniform recombination method and single point crossover, 80% of the times the uniform crossover produces better solution.

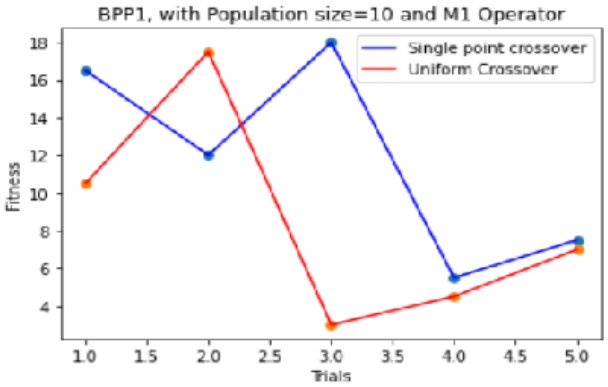

Figure5

4.3 Replacer-First weaker

This approach rapidly removes the lesser fit functions thus converging rapidly to the global optimum. As we can see from the below figure 60% of the times replacer-first weaker works better than replacing the weakest in the solution.

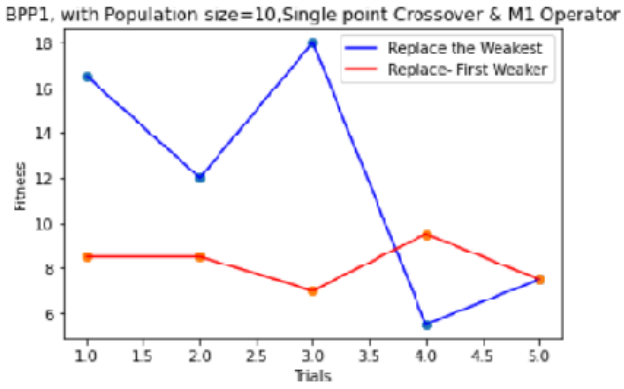

Figure6

5 ALTERNATIVE ALGORITHMS TO SOLVE BPP

I propose an Ant Colony Optimization (ACO) approach to the BPP. ACO is a meta-heuristic for combinatorial optimization and other problems. It uses a colony of ants which stochastically build new solutions using a combination of heuristic information and an artificial pheromone trail. This pheromone trail is reinforced according to the quality of the solutions built by the ants. Each ant's solution is improved by moving some of the items around, and the improved solutions are used to update the pheromone trail. The reason for proposing this approach is the knowledge that ACO and local search can work as a complementary partnership. ACO performs a rather coarse-grained search, providing good starting points for local search to refine the results.

5.1 Define the Pheromone Trail

The quality of ACO application greatly depends on the definition of the pheromone trail. For this problem we essentially want to split the items into groups, hence we can define the pheromone trail between 2 objects i and j, τ(i,j) as the favorability of having the items in the same group.

5.2 Define the Heuristic

The choice of a good heuristic is a very important feature of an ACO implementation, the heuristic is used in combination with the pheromone trail to build good solutions. For BPP we can use a random initial filling of one item in each of the bins then a best fit first i.e., the next item placed in the lightest bin. The heuristic function ή(i)=w(i)/ Σw(i).

5.3 Building a solution

The pheromone trail and heuristic function defined above will be used by ants to build solutions. Every ant starts with the randomly placing one item in each bin then choose the best (lightest) bin for the next items. The probability that the ant k will choose bin b for item j in the partial solution is given by the below equation:

$$p_k(s,b,j) = \begin{cases} \frac{[\tau_b(j)].[\eta(j)]^\beta}{\sum_{g \in J_k(s,b)} [\tau_b(g)].[\eta(g)]^\beta} & \text{if } j \in J_k(s,b) \\ 0 & \text{otherwise} \end{cases}$$

The equation Jk(s,b) is the set of items that qualify for inclusion in the current bin.

5.4 Updating Pheromone Trail

Updating the pheromone trail is given by the below equation:

$$\tau(i,j) = \rho.\tau(i,j) + m.f(s^{best})$$

Using only the best ant for updating makes the search much more aggressive. In combinations which often occur in good solutions will get a lot of reinforcement. This problem can be optimized using a construction graph. Results using ACO are often competitive with Evolutionary Algorithms

6 CONCLUSION

This report has presented the solution of Bin Packing Problem using Evolutionary Algorithm with multiple parameters like population size of 10 and 100, different selection methods, different recombination and mutation methods and several replacement methods. It also discusses the results from five trials for all the approaches and proposes the best method to solve the BPP efficiently.