Exercise 2

Smallest step size

Different from the TTP problem in exercise 1, the items in this exercise is dynamic. If it is available, it can be picked, otherwise, cannot be picked. Every 50 generations, the statues of items will change. At beginning we run those two algorithms in given two object function on the dynamic problem. As expected, the results are much worse than the results in exercise 1, since the problem is more complex. Compared with algorithm 2, algorithm 1 always has a better performance. After analysis, we found that the main difference of these two algorithm is the mutation operator (the flip approach can be seen as mutation operator). Algorithm 1 every generation flips 1 bit to produce a new individual while Algorithm 2 flips all bits with a probability 1/n (n is the length of individual). The minimum change is 1 bit. And the maximum is n bits. In engineering model, the flip bit here is changing step size. In theory, the smaller the step size is, the better result it will get. However, the lower computing speed it will also have. To confirm this suppose, we tried flip 2 bits, 5 bits and 10 bits during the mutation process. Also a dynamic probability of flipping based on algorithm 2 is also tested. As expected, almost all the test results shows that flip 1 bit is a better chose to solve this problem. This is because that only flipping 1 bit every time has **n** kinds of different results, which means it has more chances to obtain the optical value, though it also takes more time. However except last 2 instances with huge number of items, it is easy to obtain the peak for other instances in a short time. Therefore, flipping 1 bit is chosen during mutation process.

After that, another disadvantage of the chosen algorithm is that it does not do well in those instances with hug items in current stage, like the instances fnl4461* and pla33810*. It seems that it does not have ability to come up with a good result in a same generations. To improve it, we begin to use some simple method at first, since complexity of algorithm is another part during the designing of algorithm. Current algorithm every time through mutation operator create one child, and compare new child with its single parent. If the child has better performance it will be accepted. Therefore its evolutionary ability is limited. Then we change it to create 5 children every time and chose the best child. This has improved its evolutionary ability a lot.

Create more children

Through it, it also can avoid getting the local optimal solution to some extent. As we know, current algorithm is based on the Hill Climbing Algorithm which is a greedy algorithm in essence. This algorithm always choose the best solution in near space of solution, until cannot get another better solution. The main problem of this algorithm is that it is easier to get a local optimal solution rather than a global optimal solution. As shown in Fig.1, if chose one new package plan as child, from A to B, and then C, get the best solution which is a local optimal solution. After that it cannot obtain a better solution no matter which direction it chooses while only moving a small step. However, it is clearly that it is not the best solution F.

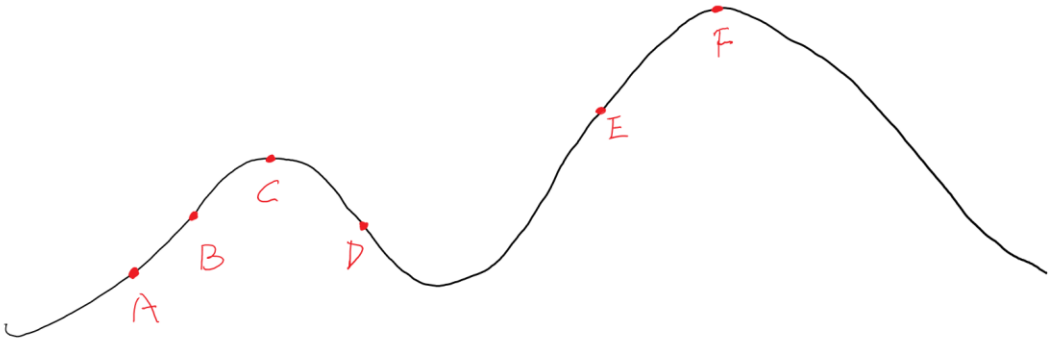

Fig.1 Hill Climbing Algorithm

Avoid Optimal Solution

However using our algorithm can avoid to some extent. During mutation process, we create more than 1 children, which means we have more chance to skip the local optimal solution C to E. Then we will have a chance to obtain the global optimal solution F.

Simulated Annealing Algorithm

Also at the same time, we have tried another solution called Simulated Annealing Algorithm. Different from Hill Climbing Algorithm always choosing the best solution in current stage, Simulated Annealing Algorithm has a probability to choose a lightly worse solution except for the best solution, therefore it has ability to skip current local optimal solution, and then obtain the global optimal solution. This algorithm can be described as follow:

If J(Y(i+1)) >= J(Y(I)), accept this move, which means move to better solution.

If J(Y(i+1)) >= J(Y(I)), has a probability to accept this move, which means have probability to move to a worse solution. And the probability reduces with the increasing of time. The probability is P(△E) = exp (△E/(KT)) which comes from simulated annealing. That's why this algorithm is called Simulated Annealing Algorithm. T is the temperature. △E is the difference of energy and △E < 0. K is a constant. P(△E) is the probability of reducing temperature. The higher the temperature is, the greater the probability is. Therefore, the process of this algorithm is that before near the global optimal solution (Highest peak F in Fig.1), it is more like randomly move to avoid local optimal value. But when it is near the global optimal solution, it will begin to move slightly to approach the global optical solution.

In theory Simulated Annealing Algorithm should have ability to avoid local optimal solution, however in this problem it does less well than improved Hill Climbing Algorithm. So we gave up it.

Results

Following fig shows the average solution quality for each number of generations and each algorithms. Algorithm1 and 2 are original algorithms and Algorithm3 is the new algorithm. From follow figures, it is easy to see that this new algorithm have a better performance than other two algorithms.

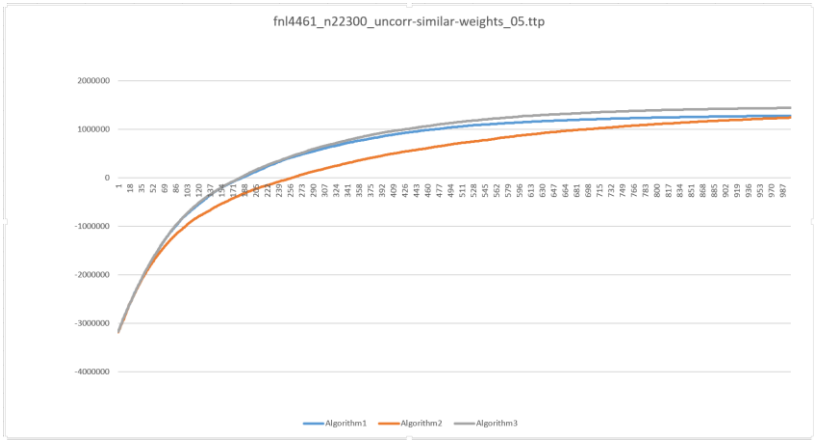

Fig.1

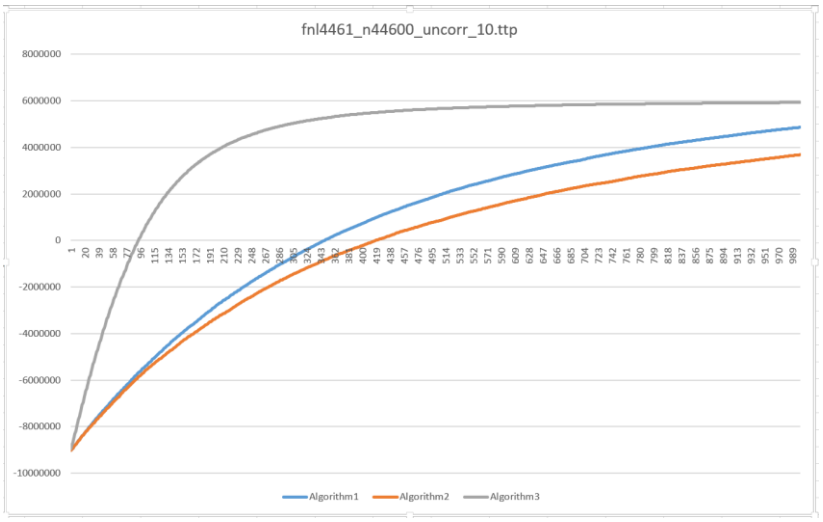

Fig.2

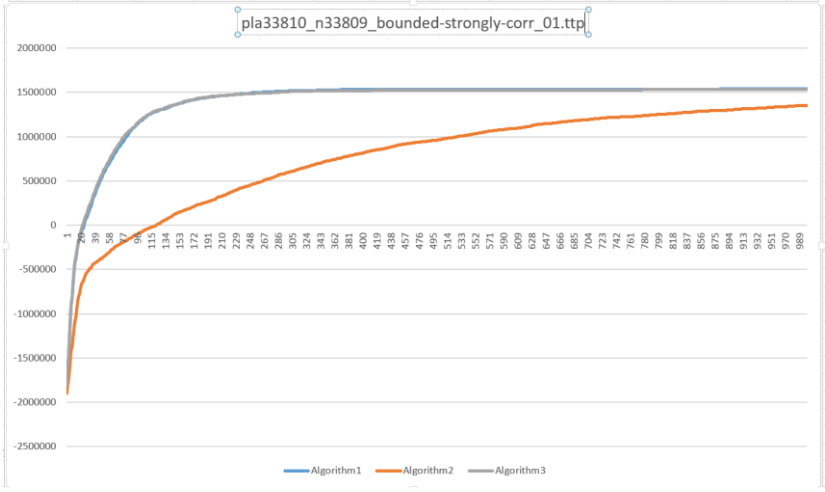

Fig.3

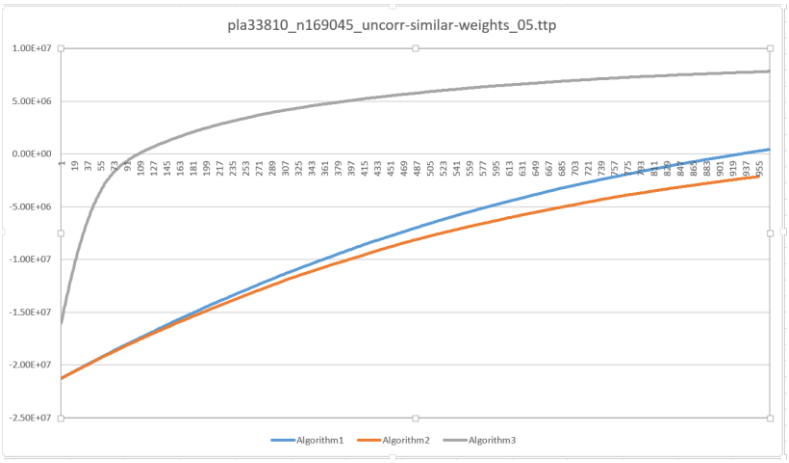
pla33810_n169045_uncorr-similar-weights_05.ttp
1.00E+07
5.00E+06
0.00E+00
-5.00E+06
-1.00E+07
-1.50E+07
-2.00E+07
-2.50E+07
Algorithm1
Algorithm2
Algorithm3

Fig.4