Application of Greedy Algorithm in Solving Service Prioritization

Chengyu Luo[1, a]

[1]School of Control and Computer Engineering, North China Electric Power University, Baoding 071000, China

412524797@qq.com

Keywords: greedy algorithm,response radio,service order,optimal sub-structure,local optimal solution

Abstract: Nowadays,there is an increasingly hot topic among the academia which concerns the exiquisite place that algorthm produce.A superior algorithm can not only soup up one thing, but also maximize the benefits and generate the smallest overhead.The kinds of algorithms abounds,such as dynamic programming method,greedy algorithm and backtracking method.As is shown in this paper,the issue of customer service priorities,to make arrangements for the priority of service , allow all customers to wait for the shortest time.The solutions to this problem overflow,first come first service,the shortest service first,for example.but all these don't very well in meeting the real needs.Using the greedy algorithms ,the response radio as the primary consideration to solve the problem , is a very simple but effective results of the method.

Introduction:

The Meaning of Greedy Algorithm

The greedy algorithm can simply be described in this way:For a sequence number from large to small sort, the largest on the front, that priority, the smallest on the back, that the final consideration. When each round is selected, a number is taken from the top of the sequence, which is the maximum value of the current sequence, that is, the local optimum of the current case, and each step is such that it is desired to find the current optimal solution.This strategy is a very simple method, it can produce the overall optimal solution for many problems, but can not guarantee always effective, because it is not for all problems can be the overall optimal solution, can only say that the solution must be A good approximation of the optimal solution.

Greedy algorithm of the basic ideas

A local solution is used to construct a global solution, that is, a certain initial solution of the problem is approximated to a given target to obtain a better solution as quickly as possible. When a step in an algorithm can no longer continue to move forward, the algorithm stops.The nature of greedy algorithmic thinking is divided, or divided, is the basis of greed.To solve the problem by greedy strategy, we need to solve two problems, one is whether thEach time the formation of local optimal solution, for another way, that is, give each one the best solution.To solve the problem by greedy strategy, we need to solve two problems, one is whether the problem is suitable for solving with greedy strategy, the other is how to choose the greedy standard to get the optimal and optimal solution of the problem.

Greedy algorithm core

The core problem of the greedy algorithm is to choose the measure which can produce the optimal solution. Namely, the specific greedy strategy is a kind of problem-solving method which gets the optimal solution from the initial state of the problem and several greedy choices.In fact, from the greedy strategy can be seen, The greedy strategy always makes the best choice at the moment, that greedy strategy is not taken into account from the whole. The choice it makes is only a local optimal solution in a certain sense.And many of the characteristics of the problem itself determines the use of greedy strategy can be the best solution or better solution.

Greedy choice of nature

The greedy choice property is that the overall optimal solution of the problem can be achieved by a series of local optimal choices.This is the first basic element of the greedy algorithm and is the main difference between the greedy algorithm and the dynamic programming algorithm.In the dynamic programming algorithm, the choice of each step is often dependent on the solution of the relevant subproblem, so only after solving the relevant subproblem can we make a choice.In greedy algorithm, only in the current state to make the best choice, that is, the local optimal choice. And then go to solve this choice generated by the corresponding sub-problems, greedy algorithm is usually top-down approach to iterative way to make successive greedy choices, each greedy choice will be done once the problem is simplified For smaller subproblems.

Properties of Optimal Substructure

When the optimal solution of a problem contains the optimal solution of its subproblem, we call this problem has the optimal substructure property.Using the greedy strategy to obtain the optimal sub-structure property of the optimal solution problem at each transformation is the key feature of this problem which can be solved by greedy algorithm or dynamic programming algorithm.Each operation of the greedy algorithm has a direct effect on the results, while dynamic programming is not. Greedy algorithm for each sub-problem solution to make a choice, can not fall back. Dynamic programming will be based on the previous selection results on the current choice, there are back-off function dynamic planning is mainly used in two-dimensional or three-dimensional problem. Greed is generally a one-dimensional problem.

greedy algorithm to achieve the process

Problem - solving ideas for

1) Establish a mathematical model to describe the problem
2) the problem is divided into several sub-problems
3) Solving for each pair of problems "gets the local optimum of the subproblem"
4) The local optimal solution of the subproblem is combined into a solution of the original solution problem.

The process of implementing the algorithm:

Starting from an initial solution of the problem;
(While)can progress toward a given overall goal
(Do) solves a solution element of feasible solution;
All solution elements are combined into a feasible solution to the problem

Customer Service Priorities

Suppose there are n customers need service, they are not arrived at the same time, they need to set the service time si, has been waiting for the time wi, 1 <= i <= n, how to arrange the average waiting time to the shortest? The average waiting time is the sum of all customer waiting times divided by the number of people n.

This problem is in fact the minimum order of the customer waiting time.

The greedy algorithm can quickly find the optimal solution or approximate optimal solution of the problem. It has to be taken into account that if it is simply for the shortest service time to provide priority services, then it will cause the first to arrive but the service time for a long time the customer is not a service problem, so greedy strategy is as follows: Priority service for customers with the shortest service response time, The response time ti is the customer waiting time divided by the time wi that it has waited, i.e., ti = wi / si, and if the service response time is longer than the same two customers, the customer waiting for a longer time has a higher priority. Because the waiting time is changing, so every time the selection finished, it need to wait for the remaining customers to update and then re-sort, and then select the next minimum response time than the customer. The loop continues until all services are complete. Algorithm is as follows.

```
for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
if(service[j].rnk==-1)
```

```
PQ.push(service[j]);
Snode temp=PQ.top();
service[temp.idx-1].rank=i+1;
PQ.pop();
totaltime+=temp.wtime;
   for(int k=0;k<n;k++)
{
if(service[k].rank==-1)
{
service[k].wtime+=temp.stime;
PQ.pop();
}
}
cout<<endl;
}
```

Table 1 is the customer waiting for the initial situation of service, you can see that each customer waiting time and service requirements of the time is different.

Table 1 service sorting table

Customer i	1	2	3	4	5	6	7
Waiting time wi	4	8	2	1	4	7	6
Service hours si	3	6	7	9	8	10	4
Service response ratio	1．33	1.33	0.29	0.11	0.5	0.70	1.20
Service order	-1	-1	-1	-1	-1	-1	-1

The first choice, according to the customer's response time ratio to sort them, get service priority queue. In the queue to select the largest response time ratio, that is, the customer with the highest priority is served. From the table we can see that customer 7 response time ratio is the maximum, so it was the highest priority, the first being served. After the customer 7 gets the service, the remaining customers are reordered, and their waiting time is added to the service time of customer number 7.

Table 2 service sorting table

Customer i	1	2	3	4	5	6	7
Waiting time wi	8	12	6	5	8	11	6
Service hours si	3	6	7	9	8	10	4
Service response ratio	2.66	2.00	0.85	0.56	1.00	1.10	1.20
Service order	-1	-1	-1	-1	-1	-1	1

Greedy algorithm difficult

How to construct a larger solution with a small solution? Overall, this is related to the problem itself. But most of them are regular, because greedy algorithm has such a nature, it can be faster than other algorithms. To prove the correctness of greedy nature, greedy algorithm is the real challenge, because not every local optimal solution will be linked with the overall optimal solution, the greedy algorithm is often generated by the solution is not the optimal solution. In this way, greedy nature of the proof has become the key to the correct greedy algorithm. For some problems greedy nature may be wrong, even if it is feasible in most of the data, but must take into account all possible special circumstances, and prove that the nature of greed in these special cases is still correct, and So easy to fall into the nature of the greedy nature of the mud can not extricate themselves , because the scope of

greedy algorithm is not large, and some very difficult to prove that if not sure, it is best not to take risks, there are other algorithms should be insurance.

Conclusions

Through the analysis, greedy algorithm can solve the problem with high efficiency and low time complexity, meanwhile,it is consistent with people's conventional way of thinking, being widely used and popularized. Although it can not guarantee that the final solution is necessarily the best, the algorithm can determine a range of possibilities for some problems. The choice of the greedy algorithm relies on the choices made in the past, but never on future ,Which makes the algorithm has a certain speed advantage in the encoding and execution process. The greedy algorithm is a better one to find the optimal solution to the problem when the optimal solution can only be obtained by Exhaustive Attack method.

Acknowledgments

This work was financially supported by xxx fund.

References

[1]Xi Cheng.0-1 knapsack problem based on greedy algorithm
[2]Yulin Huang.A Greedy Algorithm for 0-1 Knapsack Problem
[3]Junjun Dong.Comparison and Analysis of Dynamic Programming and Greedy Algorithm
[4]Heng Xiao.On Greedy Algorithm
[5]Youqu Chang.Research and Research on Greedy Algorithm
[6]Shuying Yang.Greedy Algorithm and Its Case Study