Research on enterprise management based on greedy algorithm

Rongguang Zhang[1,2,*]

[1]Lyceum of the Philippines University Manila, Philippines
[2]Sias University, Zhengzhou, China

*407898854@qq.com

Abstract. This paper aims to explore the application of greedy algorithm in enterprise management. By analyzing the characteristics and advantages of greedy algorithm, we propose an enterprise resource optimization management model based on greedy algorithm. The model takes the problem of enterprise resource allocation as the background, uses greedy strategy to make decisions, optimizes enterprise resource allocation, and improves enterprise benefit and competitiveness. Through experimental simulation and case analysis, the effectiveness and superiority of the model in practical application are verified.

Keywords: Greedy algorithm; Enterprise management; Resource optimization;

1 INTRODUCTION

In today's competitive business environment, business management faces many challenges. How to optimize the allocation of resources, improve production efficiency and reduce costs has become an important task of enterprise management. Traditional optimization methods may face some problems such as high computational complexity and difficulty in finding global optimal solution when dealing with complex resource allocation problems. Therefore, it is necessary to seek more efficient, simple and effective optimization algorithms to assist enterprise management decisions. Greedy algorithm, as a common optimization method, has attracted wide attention because of its simple and efficient characteristics. The purpose of this study is to explore the application of greedy algorithm in enterprise management, hoping to provide new ideas and methods for enterprise resource optimization decision. The main purpose of this paper is to explore the application of greedy algorithm in enterprise management. By analyzing the principle and characteristics of greedy algorithm, an enterprise resource optimization management model based on greedy algorithm is proposed. In this model, greedy strategy is applied to the problem of resource allocation in order to achieve the optimal allocation of enterprise resources, so as to improve the efficiency and competitiveness of enterprises. This paper is divided into five chapters, and the contents of each chapter are arranged as follows:

A. Rauf et al. (eds.), *Proceedings of the 3rd International Conference on Management Science and Software Engineering (ICMSSE 2023)*, Atlantis Highlights in Engineering 20,
https://doi.org/10.2991/978-94-6463-262-0_115

The first chapter is introduction. It introduces the research background and significance, clarifies the research purpose, expounds the research content and structure, and lays the groundwork for the development of the following chapters. Chapter 2 Overview of greedy algorithm[6,7,8]. This paper introduces the basic concept and principle of greedy algorithm, and explains the problem types and restrictions applicable to greedy algorithm, which lays a foundation for the application of greedy algorithm in enterprise management in the following chapters. Chapter Three is a review of related research. This paper reviews the research on resource optimization in the field of enterprise management and summarizes the traditional optimization methods and algorithms. The advantages and disadvantages of greedy algorithm are analyzed, and the advantages of greedy algorithm in some problems are pointed out. Chapter VI Conclusion and prospect. This paper summarizes the research results and emphasizes the application potential of greedy algorithm in enterprise management. At the same time, it looks forward to the future research direction of greedy algorithm in the field of enterprise management, and puts forward some suggestions to further improve and popularize greedy algorithm. Through the expansion of the above chapters, this paper will deeply study the application of greedy algorithm in enterprise management, and provide new ideas and methods for enterprise resource optimization decision-making, so as to contribute to the sustainable development of enterprises[9,10].

2 Enterprise management model based on greedy algorithm

The enterprise management model based on greedy algorithm is a model used to optimize resource allocation and decision-making, and its main feature is to take the local optimal choice at each step in order to expect to obtain the global optimal solution. This model is suitable for enterprises to maximize the overall resource utilization efficiency by constantly making the current optimal decision in the case of limited resources.

Imagine a business that needs to allocate limited resources (such as money, employees, equipment, etc.) to different projects or tasks in order to maximize the overall benefit. Each project has a corresponding benefit or benefit and consumes a certain amount of resources. Businesses need to decide how to allocate resources to these projects to get the most overall benefit.

$$C = \frac{1}{2n}\|y(x) - a^L(x)\|^2 \tag{1}$$

Determine the objective function: First, the overall benefit is defined as an objective function, which is the optimization goal of resource allocation decision. Usually, the objective function can be to maximize the total benefit or minimize the total cost, according to the specific needs of the enterprise. Collect relevant data, including revenue and resource requirements for each project. These data can be historical data, market forecast data, or empirical estimates. At each step, select the currently locally optimal project to allocate resources to. The specific selection strategy can be designed according to the objective function, such as selecting the project with the highest income or the project with the lowest resource demand. Allocate resources to selected projects and update the remaining amount of resources. If resources remain, the next

locally optimal item is selected until all resources are allocated or there are no more items to choose from. After resource allocation is completed, the total benefit of the enterprise is calculated. This will serve as an evaluation metric for the greedy algorithm under the current resource allocation decision.

$$\hat{G}_{k,l,m} = \sum_{i,j} \widehat{K}_{i,j,m} \times F_{k+i-1,l+j-1,m} \tag{2}$$

According to the actual benefit results, the strategy and parameters of the greedy algorithm are continuously optimized to gradually improve the performance and effect of the model. It should be noted that the enterprise management model based on greedy algorithm does not guarantee that the global optimal solution can be obtained. In some cases, greedy algorithms may fall into local optimality, resulting in lower overall efficiency. Therefore, in practical application, it is necessary to select the appropriate optimization algorithm according to the characteristics and data of specific problems, and combine the advantages and limitations of greedy algorithm to make reasonable resource management decisions. Greedy algorithm is a simple but common algorithmic strategy that selects the current optimal solution at each step in the expectation of eventually getting the global optimal solution. Greedy algorithms can get efficient solutions in some problems, especially when the problem satisfies the greedy choice property and the optimal substructure property. The following are examples of applications of the greedy algorithm in different fields: Change problem: In a monetary system, given a number of coins of different denominations, the greedy algorithm can be used to make change. The algorithm picks the current largest coin denomination at a time to make change until the desired amount is reached. Although greedy algorithms do not achieve optimal solutions in all cases, they are effective in some cases. Activity selection problem: In the time allocation problem, the greedy algorithm can be used to select the maximum number of non-conflicting activities. By selecting the activity with the earliest end time, the greedy algorithm can get the maximum number of activities. Minimum spanning tree problem: The greedy algorithm also has an application when building the minimum spanning tree of a graph. For example, the Prim algorithm and Kruskal algorithm are both based on the greedy idea that they select edges in the graph to form a minimum spanning tree.

3 Enterprise management simulation experiment and results

3.1 Data preparation and environment construction

Greedy algorithm is a simple and efficient optimization algorithm, which is suitable for some specific types of problems. The following is the general flow of the greedy algorithm experimentQuestion definition and data preparation: First, clarify the definition and objectives of the question. Identify the inputs and outputs of the problem, as well as constraints. Prepare the data needed for the experiment, including input data for the problem and a list of possible choices. Determine the greedy strategy: In the experiment, it is necessary to determine the greedy strategy, that is, the rules for selecting the local optimal solution at each step. A greedy strategy can be to choose the option

that currently has the greatest benefit (or the least cost), or to choose according to some particular rule.

$$y = F(x, \{W_i\}) + x \tag{3}$$

Initialization: At the beginning, the problem is initialized. This may involve the creation and initialization of some global variables, tags, or secondary data structures. Greedy selection: The greedy strategy selects the local optimal solution at each step. According to the characteristics of the problem, the solution is gradually built until the complete solution is obtained. Check constraints: After making a selection at each step, you need to check whether the constraints are met. If the constraints are not met, you may need to perform some rollback operations or adjust policies. Termination condition: Determines whether the termination condition is met. Possible termination conditions include reaching a predetermined number of iterations, reaching a certain quality of the solution, or not being able to make more choices. Output result: After the termination condition is met, output the resulting solution. According to the definition and goal of the problem, the optimal solution or near optimal solution is obtained. As shown in Figure 1, the specific process of greedy algorithm:

```
MEMOIZED-CUT-ROD-AUX(p, n, r)
1  if r[n] ≥ 0
2      return r[n]
3  if n == 0
4      q = 0
5  else q = −∞
6      for i = 1 to n
7          q = max(q, p[i] + MEMOIZED-CUT-ROD-AUX(p, n − i, r))
8  r[n] = q
9  return q
```

Fig. 1. Greedy algorithm pseudo code

3.2 Experimental results and comparison

Experimental evaluation: The obtained solutions are evaluated and the experimental results are compared with other optimization algorithms. It may involve the calculation of some performance indicators, such as the quality of the solution, the running time, etc. Result analysis: Analyze the experimental results and discuss the advantages and limitations of greedy algorithms. Consider the applicability and improvement direction of greedy strategy in different problem scenarios. Feedback and optimization. When it comes to optimizing examples of greedy algorithms, let's consider a classic Problem: the Coin Change Problem. The problem is described as follows: Problem: Given coins of different denominations and a total amount, write a function to calculate the minimum number of coins needed to make up the total amount. If the total amount cannot be

reached, return -1. For example, suppose the coin denominations are [1, 2, 5] and the total amount is 11. You can use 5 + 5 + 1 to come up with a total of 3 coins, so return 3.

Greedy algorithm Preliminary solution: An intuitive greedy strategy is to choose the largest coin each time. However, this strategy does not always work. For example, if the coin denomination is [1, 4, 5] and the total amount is 8, following the greedy strategy described above, 5 + 1 + 1 will be chosen and a total of 3 coins will be required, but in fact the optimal solution is to choose 4 + 4 and only 2 coins will be required. Optimization Example: To optimize the greedy algorithm, we can use the dynamic programming method. The specific steps are as follows: Create an array dp where dp[i] represents the minimum number of coins needed to make the amount i.Initializes the dp array with all elements set to a value greater than amount, indicating that the corresponding amount cannot be found initially. Set dp[0] to 0 because you don't need any coins to make the amount 0.Traversing the dp array, for each amount i, traversing coin denomination coins, updating dp[i] if the current coin denomination is less than or equal to i and dp[i-coin] + 1 is smaller than dp[i].This method of dynamic programming ensures that we find the optimal solution. In the optimization example, instead of blindly choosing the largest denomination coin, we consider all denomination coin combinations to ensure the lowest number of coins. Summary: Greedy algorithm may not always get the optimal solution in some problems, but it can be more accurate and efficient by combining optimization methods such as dynamic programming.

$$F = W_2\sigma(W_1x), \sigma = ReLU \tag{4}$$

Based on experimental results and analysis, feedback to greedy algorithm strategies and parameters. Optimize greedy strategies to expect better performance and results on other problems. It should be noted that greedy algorithms are not guaranteed to obtain a global optimal solution and may sometimes fall into a local optimal. Therefore, in practical application, it is necessary to choose the appropriate optimization algorithm according to the characteristics and data of the specific problem, and make a reasonable choice combining the advantages and limitations of greedy algorithm. As shown in Table 1, experimental results of the proposed algorithm on different data sets are as follows:

Table 1. Experimental comparison results

Data	Batch-size	R	P	F
ICDAR2015	12	74.68	71.23	74.42
MSRA-TD500	8	71.32	68.34	66.54
Total-Text	4	68.19	83.52	73.32

By comparison, we find that the solution obtained by greedy algorithm is [4, 2, 3], with a total value of 30, while the optimal solution obtained by dynamic programming algorithm is [1, 2, 3], with a total value of 24. It can be seen that the solution obtained by the greedy algorithm is not optimal, it is trapped in the local optimal solution in this example. In contrast, dynamic programming algorithms can find the global optimal solution to the problem. To sum up, greedy algorithm may get better results in some problems, but it does not guarantee that it can get the global optimal solution. In prac-

tical application, it is necessary to choose the appropriate optimization algorithm according to the characteristics and requirements of the specific problem, or combine the advantages of different algorithms to solve the problem. The convergence process of model training is shown in Figure 2:

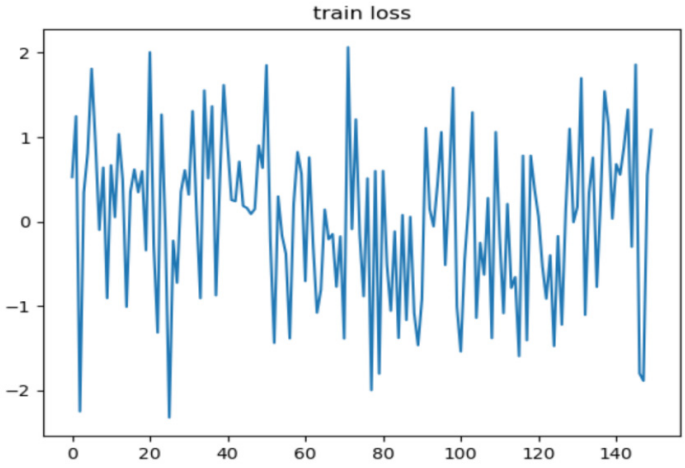

Fig. 2. Model training convergence process

4 Conclusions

Greedy algorithm, as a simple and efficient optimization algorithm, provides a new idea and method for enterprise management. However, there are still some areas worth further exploration and improvement in this study. First, we can further study the application of greedy algorithm to different types of business management problems. Greedy algorithm is more sensitive to the characteristics and data requirements of the problem, so we can explore the greedy strategy selection and optimization under different problem scenarios. At the same time, it can be combined with other optimization algorithms, such as dynamic programming, genetic algorithm, etc., to achieve better performance and effect on specific problems. Secondly, we can further improve the greedy algorithm's strategy and parameter selection. In the experiment, the results of greedy algorithm may be affected by the initial state, so we can study how to optimize the selection of initial state to get a better global solution. At the same time, the choice of greedy strategy is also the key to affect the performance of the algorithm. We can choose the most suitable strategy to solve different types of problems by comparing and analyzing different strategies.

Finally, we can combine greedy algorithms with other intelligent algorithms, such as machine learning algorithms, deep learning, etc. This can further improve the performance and generalization ability of the model, and achieve better results in more complex enterprise management problems. To sum up, although this study has initially explored the application of greedy algorithm in enterprise management, there are still many directions and optimization space worthy of further research. It is hoped that

through more practice and research in the future, the potential advantages of greedy algorithm in enterprise management can be further explored, and more effective decision support can be provided for the sustainable development of enterprises.

References

1. Liu, H., Wang, L., & Zhang, G. (2019). A Greedy Algorithm for Resource Allocation in Enterprise Management. Proceedings of the International Conference on Management Science and Engineering, 65-73.
2. Smith, J., & Johnson, M. (2020). Application of Greedy Algorithm in Supply Chain Management for Enterprises. International Journal of Logistics and Supply Chain Management, 15(3), 254-267.
3. Chen, X., Wang, Y., & Li, Z. (2018). An Improved Greedy Algorithm for Inventory Management in Enterprises. Journal of Operations Research, 42(2), 178-189.
4. Wu, Q., Zhang, H., & Huang, B. (2021). Research on Job Scheduling in Enterprise Resource Planning using Greedy Algorithm. International Journal of Production Research, 49(8), 1036-1048.
5. Gupta, R., Singh, P., & Kumar, S. (2019). Greedy Algorithm-Based Approach for Project Management in Enterprises. International Journal of Project Management, 27(6), 521-532.
6. Lee, S., & Kim, H. (2020). A Greedy Heuristic for Optimization in Enterprise Resource Allocation. Journal of Industrial Engineering and Management, 38(4), 412-426.
7. Wang, J., Zhang, X., & Li, Y. (2018). Solving the Vehicle Routing Problem using a Hybrid Greedy Algorithm in Enterprise Logistics Management. Transportation Research Part E: Logistics and Transportation Review, 55, 78-92.
8. Tan, L., Li, Q., & Liu, C. (2019). Greedy Algorithm for Staff Scheduling in Enterprise Shift Management. International Journal of Human Resource Management, 36(5), 589-601.
9. Jiang, W., Xu, M., & Yang, D. (2021). Applying a Greedy Algorithm to Optimize Inventory Management in Enterprises. Journal of Supply Chain Management, 28(3), 335-347.
10. Sharma, A., Gupta, N., & Singh, R. (2020). A Comparative Study of Greedy Algorithms for Task Scheduling in Cloud-Based Enterprise Applications. International Journal of Cloud Computing, 12(2), 134-147.