Design and Implementation of Course Arrangement Model Based on Reforming Deep Reinforcement Learning

Zhang Lei¹, Fangqin Xu²*

¹Admissions Office, Shanghai Jian Qiao University, Shanghai, 201306, China
²Information and technology College, Shanghai Jian Qiao University, Shanghai, 201306, China

*Corresponding author’s e-mail: xufqin@gench.edu.cn

Abstract: Aiming at the problem of curriculum arrangement, this article abstracts it into two subjects and multiple attributes, and successfully abstracts the problem into a curriculum arrangement model design problem. This paper analyzes the characteristics of the curriculum arrangement problem and combines the characteristics of the deep reinforcement learning algorithm to propose a reformed deep reinforcement learning model. At the same time, the state representation method and action selection method of each stage are designed. Different from general deep reinforcement learning, this article no longer uses action set as the behavior process, and selects the course by optimizing the weight to achieve the purpose of course arrangement. Finally, this paper carried out a comparative experiment to verify the algorithm, which proved that the deep reinforcement learning algorithm based on transformation can effectively solve the problem of curriculum arrangement, and has higher efficiency and higher learning ability than genetic algorithm.

1. Introduction

With the popularization of education, colleges and universities have expanded their enrollment year after year, the number of people has been increasing, and teaching resources have become increasingly tight. However, due to many objective factors such as limited investment costs and tight school land, the problem of curriculum arrangement has become more and more difficult, and there may even be problems with arranged courses but no classrooms. With the continuous development of many algorithms, many schemes have emerged to solve the problem of scheduling using algorithms: some have solved the problem of scheduling under the coexistence of shift system and administrative classes through the idea of combining optimization algorithms [1]; Genetic algorithm optimizes class scheduling [2]; there are also solutions to class scheduling problems through intuitionistic fuzzy decision-making algorithms. This article will use deep reinforcement learning to model the problem of class scheduling, mainly considering the changes in the class location during the class scheduling process, the uneven distribution of class locations, etc., to solve the problems of different class hours, different locations, and uneven use of classrooms. The purpose is to construct a course arrangement model that can adapt to the current shortage of classroom resources.

2. Deep reinforcement learning

The main indicators to measure the level of intelligence in the field of advanced artificial intelligence are perception ability and decision-making ability. Deep reinforcement learning (DRL) combines the
strong decision-making ability of reinforcement learning (RL) and the feature extraction ability of deep learning (DL) to form an end-to-end perception and control system. The algorithm can be described as three steps:

1) The subject's effect on the environment, obtain observations, use DL methods to perceive observations, and obtain specific state characteristics;
2) Evaluate the value function of each action in the United States according to the expected benefits, and map the current state to the corresponding action through a certain strategy;
3) The environment reacts to the action and obtains the next observation.

Cycle through three steps, and finally obtain the best strategy to achieve the goal. This method has been officially used in games, robot control, parameter optimization and other fields [3].

3. Analysis and Modeling of Course Arrangement

3.1. Analysis of Course Arrangement

Solving the problem of course arrangement is actually a process of arranging courses, teachers, and students in a classroom in a time shop. First, we must analyze and determine the necessary and non-essential conditions in the course arrangement process. Necessary conditions refer to the conditions that must be met during the course arrangement process. Failure to satisfy such conditions will cause the course arrangement results to be unusable.

The necessary conditions include the following:
1) At the same time, the same teacher can only teach one course;
2) At the same time, only one course can be arranged in the same classroom;
3) At the same time, only one course can be arranged in the same class;
4) The number of seats in the classroom cannot be less than the number of people in the class;

Non-essential conditions refer to the conditions that are expected to be met in the course of scheduling. The scheduling results that meet these conditions are requirements that can be recognized and are also the main problems that this article solves.

Non-essential conditions include the following:
1) The same course (classes, teachers, and courses are the same), with the same schedule every week;
2) For the same course (classes, teachers, and courses are the same), the location of each lesson is the same;
3) The usage frequency of each classroom is evenly distributed with two weeks as a single temperature.

Due to the limitation of calculation cost and processing cost, this article mainly solves the above problems as the test results to verify the feasibility of the algorithm.

3.2. Course scheduling problem modeling

To optimize the course arrangement through deep reinforcement learning, it is necessary to establish the course arrangement problem as a model that can be trained by deep reinforcement learning.

On the issue of curriculum arrangement, there are two main bodies: curriculum and classroom. These two have their own attributes: the number of people in the classroom, and considering that the number of hours in the course may vary, the state attribute is added to determine whether the classroom can be used at this point in time; the course is a broad definition, and its attributes including the class, teachers, and courses taught. The class also includes the attributes of the number of people. The courses taught include the attributes of the number of class hours. Instead, they themselves are not valued in the course scheduling process. These attributes are defined as broad courses. Therefore, the curriculum includes the attributes of class, teacher, number of hours, and number of students.

The time dimension is set to four times a day, that is, two classes in the morning and two classes in the afternoon. Considering the actual situation, a classroom may be unavailable for a certain period of time. Lessons, so the week is divided into 19 time periods.
4. Design of Deep Reinforcement Learning Algorithm for Course Arrangement

4.1. State model
Let the classroom attribute be the formula:
\[ c \equiv \{\text{classNo}, \text{siteNum}, \text{isAvailable}\} \]
Let the classroom set be the formula:
\[ \text{classrooms} \equiv \{c1, c2, c3, c4, \ldots \} \]
Let the course attribute be the formula:
\[ l \equiv \{\text{lessonNo}, \text{classNo}, \text{teacher}, \text{stuNum}, \text{classHour}\} \]
Set the course collection formula:
\[ \text{lessons} \equiv \{l1, l2, l3, l4, \ldots \} \]
Set the time set formula:
\[ \text{time} \equiv \{T1, T2, T3, T4, \ldots \} \]
Arranging courses in a classroom at a time point is expressed as a formula:
\[ \text{schedule} \equiv \{\text{classNo}, \text{siteNo}, \text{isAvailable}, \text{lessonNo, classNo}, \text{teacher, stuNum, classHour}\} \]
The state modeling in the course scheduling process builds a two-dimensional table with the collection of each attribute, and the initial state is the formula:
\[ \text{state} = \{\} \]
The next state after one action is expressed as:
\[ \text{state}' = \{\text{schedule}\} \]
By analogy, each time an arrangement is made, a state is added to state.

4.2. Action space model
A point in time is a unit, and all classrooms at that point in time are scheduled to be an action.
However, if the result of permutation and combination is regarded as an action, it may lead to an excessively large action space. In particular, there are hundreds of courses in each semester, and all the possibilities of permutation and combination with the classroom will cause the model to be unable to optimize at all, or the optimization is extremely slow. Therefore, this article designs a function as an action, and the function corresponds to each attribute. Reinforcement learning algorithm is used to optimize the weight, the function result is as a score, and the best course is selected as an option among the remaining courses.
The function when arranging courses in a classroom is:
\[ \text{choiceScore} = \sum_{i=1}^{m} w_i \times \text{attr}_i \]
Among them, \( m \) is the number of attributes, \( w_i \) is the weight value corresponding to the \( i \)-th attribute, and \( \text{attr}_i \) is the value of the \( i \)-th attribute.

4.3. Reward modeling
After each action operation, the return of the current result needs to be calculated. This return is a cumulative return, that is, it needs to be calculated through all the previous results. The calculation method is as follows:
1) The scoring reference model design is the goal that needs to be achieved, that is, necessary and non-essential conditions. Points will be added when the conditions are met, and points will be deducted if the conditions are not met;
2) According to experience, since the necessary conditions must be met, when the necessary conditions are not met, -100 points, if the necessary conditions are met, +20 points;
3) -10 minutes when non-essential conditions are not met, +15 minutes when met;
4) Record the current status for each condition score as \( R = \{r_i | i = 1, 2, \ldots, m\} \)
5) Record the result status for each condition score as \( R' = \{r'_i | i = 1, 2, \ldots, m\} \)
6) Calculate each requirement according to the current state, the formula is: \( \text{reward} = \sum_{i=1}^{m} r'_i - r_i \)
After an action is performed, the next action needs to cover the previous state \(1\) with the current state \(2\). The state after the next action is completed is recorded as \(2\), but in order to exclude the occurrence of necessary conditions are not met, the state affects the calculation result, so artificial It is stipulated that when the necessary conditions are not met in the current state \(2\), no overwriting operation is performed, and the previous state \(1\) is used as the state of the next action.

5. Algorithm flow

The flow of the algorithm designed in this paper is as follows:

- Set the initialization state to \{\}, the action determined according to the initial weight is \(A\), the discount rate is \(\gamma\), and the learning rate is \(\alpha\)
- Initialize the experience pool with a capacity of \(N\)
- Initialize the Q network, the initial weight is the set weight \(\omega\)
- Initialize target-Q network, let \(\omega' = \omega\)
- For episode = 1, \(M\) do
  - For \(t=1, T\) do
    - Calculate the return \(r\) on the input data \((s, \omega', r', s')\)
    - Store the sample \((s, \omega', r', s')\) in \(D\)
    - Sampling \((s'', \omega'', r'', s''')\) from \(D\)
    - \(y = \begin{cases} r'' & \text{is End} \\ r'' + \gamma \max_{\omega''} Q_{\omega''}(s'', \omega'') , x \geq 0 \end{cases}\)
    - take \(y = Q_{\omega''}(s'', \omega'')^2\) as a loss function, train the Q network
    - If the necessary conditions are not met, let \(s \leftarrow s'\)
    - Update \(\omega'\) every \(C\) steps, \(\omega' \leftarrow \omega\)
  - End of the cycle or until \(s''\) is the final state
- End of cycle or until \(\forall s, \omega, Q_{\omega}(s, \omega)\) become convergent
- Output Q-network: \(Q_{\omega}(s, \omega)\)

Since the setting of the action space in this article is different from the general deep Q network, the weight of the Q network is directly used as the course fitness scoring and selection when selecting courses.

6. Experiment and analysis

In this paper, a comparative experiment is conducted through Python. The experimental data is 200 courses (100 courses with a weekly course of 1, 50 courses with a weekly course of 2, and 50 courses in a single week). There are 19 time periods per week. 20 classrooms; set the initial weight used to select courses as \{1,1,1,1,1\}. The manual arrangement, genetic algorithm-based course arrangement and deep-reinforced learning-based course arrangement were carried out respectively for comparative experiments, and the time cost, calculation cost and course scheduling results of the three were compared. The comparison results are as follows:

|                  | Time  | Memory Usage | CPU Usage | Necessary conditions | Non-essential conditions |
|------------------|-------|--------------|-----------|----------------------|-------------------------|
| Labor            | 12 Hours | 0%          | 0%        | Fully satisfied      | Partially satisfied     |
| Genetic algorithm| 3 Hours | 15%         | 30%       | Partially satisfied  | Partially satisfied     |
| Deep reinforcement learning | 3 Hours | 80%         | 100%      | Fully satisfied      | Partially satisfied     |
It can be seen from the results that compared with manual operation, the solution in this paper can achieve the same effect in a shorter time; and compared with genetic algorithm, the result obtained by genetic algorithm cannot meet the necessary conditions in the same time, so it cannot be used. It takes more time to iterate and optimize. After continuing to optimize the above model, it is found that the model tends to converge in about 8 hours.

7. Summary
This paper designs a course arrangement model design based on deep reinforcement learning. After modeling the course arrangement problem, the deep Q neural network is transformed. After comparative experiments, it is found that the model designed in this paper can deal with the problems of uniform use of classrooms and non-uniform curriculum arrangements encountered in the course arrangement.

At the same time, the model in this paper still has the possibility of further optimization. For example, the reusability of the model is not enough to verify the data, and the scalability is considered. These problems require targeted transformation of the algorithm during the model design process, which is also the future research direction.

References
[1] Hao X.X. (2019) Research on the Representation of Combinatorial Optimization Problem and Evolutionary Optimization Algorithm, Xidian University.
[2] Ma X.S, Li F.R. (2020) Research on the Application of the Course Arrangement System Based on Improved Genetic Algorithm. Journal of Qingdao University (Engineering Technology Edition), 35(03): 15-20+55.
[3] Zhu Z.M. (2020) Design and implementation of intelligent classroom control model based on deep reinforcement learning. Donghua University.