An Experience-Computer-Based University Course Arrangement System

Chun Yu, Yanxia Li, Xin Wang, Chunyu Miao, Qian Wang
Computer and Information Management Center
Tsinghua University
Beijing 100084, China
yuchun@cic.tsinghua.edu.cn; liyx@cic.tsinghua.edu.cn; wxin@cic.tsinghua.edu.cn; mcy@cic.tsinghua.edu.cn;
wq@cic.tsinghua.edu.cn

Abstract—Course arrangement is an important part of educational administration in university. With the improvement in information system construction and the development in education reform, it brings up higher request for course arrangement. This paper proposed an intelligent course arrangement system based on experience-computer, which solves the problem by designing appropriate time pattern and priority model, as well as using the heuristic algorithm, backtracking algorithm and tabu search algorithm. It is proven that the system embodies the concept of education management reform, and improves the efficiency of course arrangement in colleges and universities.

Keywords—Course arrangement; experience-based; heuristic algorithm; backtrack algorithm; tabu search algorithm

I. PREFACE

University timetable reflects teaching and extra-curricular activities in a semester, a week or even every day. It influences students’ learning effect and courses’ teaching quality. Indeed, timetable is the schedule of whole teaching work of university, which constitutes an important guarantee for stable teaching order. It reflects the universities’ capacity and level of teaching support and teaching management, as well as the leader’s guiding ideology. With the rapid expansion of universities and colleges, there is a severe shortage of teaching resources. The rationality of curriculum arrangement has influenced teaching order’s normalization and stability directly.

II. PROBLEM DESCRIPTION OF COURSE ARRANGEMENT

Course arrangement is a process to find the combination of teachers, students, courses, time and classrooms, which makes the best use of teaching resources on the premise of ensuring normal teaching activities and filling individual requirements. By mathematical methods, it resolves resource conflicts and satisfies all constraints. Constraints in course arrangement consist of mandatory conditions, neutral conditions and soft conditions. It’s a valid solution only when mandatory condition is satisfied, while neutral conditions and soft conditions are used to evaluate the solution.

A. Mandatory condition

The mandatory conditions are absolutely necessary conditions without which might lead to the teaching working disability.

1) Only one course can be appointed to a teacher at the same time.
2) Only one course can be appointed to a class at the same time.
3) Only one course can be appointed to a classroom at the same time.
4) The number of a classroom’s available seats should not be less than that of students during scheduling.

B. Neutral condition

The neutral conditions are situated between absolutely mandatory conditions and extremely soft conditions and are generally decided by universities themselves, because teaching effect, learning efficiency, teachers and students’ convenient should also be considered besides normal teaching activities.

1) Satisfy course’s requirement on class time as far as possible.
2) Satisfy course’s requirement on classroom as far as possible.

C. Soft condition

The soft conditions are higher requests on the basis of above conditions. They can be added when needed, and did not influence the range but the superior of feasible solution.

1) Use the same classroom or neighbourhood for students’ adjacent classes to avoid large flow of students moving to another classroom during the break.
2) Hold obligatory course in the morning to improve learning efficiency, because students and teachers are more vibrant in this period of day.
3) Do not arrange too many classes for students and teachers, better less than six successive classes one day.

III. DESIGN AND IMPLEMENTATION OF COURSE ARRANGEMENT

Problem of course arrangement should be solved in 2 ways, experience and computer together. Experience means to sum up working experience, which aims to define an appropriate time pattern and place model, to establish a reasonable sequence, and to arrange courses group after group. Computer means to utilize the advantage of computer - fast speed, to find the best (maybe almost best) schedule in case of large amount data as well as reduce human working greatly.
A. Experience model

1) Time pattern

Some irrational arrangements, such as two classes of one course arranged in one day between which students are unable to do homework and digest knowledge, have to be settled by defining rational time pattern basing on experience.

Time pattern consists of class hours per week, class day of week, period of day and other items. According to time intervals between classes and their feathers, we define time mode to improve the efficiency, such as common mode, unit mode, night mode, whole day, and etc. Each mode is composed of one or many time split, each time split is composed of one or many time record, and each time record appoints class day of week and period of day. Sum of time records hour should equals to class hours per week. Time pattern composition are shown in Fig. 1.

![Time Pattern](image)

Figure 1. Time Pattern.

Time mode was used to select class time in course arrangement system. The time records with higher priority should be selected first from those of same time mode and same hours per week. Highest priority of time record is 10, least is 1. Higher priority means more welcomed by teachers and students. Time patterns are defined in advance by administrators upon their experience.

2) Place model

Building and classroom are 2 components of place model. Classroom consists of floor, room number, capacity, classroom type, multimedia device, and etc. Place selection in course arrangement is determined by its priority which composes of building priority and classroom priority too. The building with higher priority should be selected first, and then the classroom in the building with higher priority. This model is groped and revised by experts according to their working experience and teachers’ and students’ common reflection.

3) Arranging priority

Arranging courses in a right order is the key to success, less confliction and higher efficiency. After many years of research, practice, and analyze, we summarized a set of numerical method to calculate arranging priority, so as to ensure the course with more classes, higher demand can be arranged first. The calculation formula of arranging priority is shown in (1).

$$
\Pi = 3 \sum_{i=1}^{X_i} (X_i \times \Omega_i) + 3 \sum_{j=1}^{A_j} (A_j \times \Theta_j)
$$

(1)

$\Pi$: arranging priority.

$X_i$: number of courses holding in the same time, often used to describe a group of courses from which student can choose only one.

$\Omega_i$: weight of courses holding in the same time.

$A_j$: capacity of course.

$\Theta_j$: weight of course capacity.

$X_i$: number of classes in the course.

$\Omega_j$: weight of classes in the course.

$A_i$: 1 when time and place both appointed, else 0.

$\Theta_i$: weight of appointed time and place.

$A_i$: 1 when time appointed, else 0.

$\Theta_i$: weight of appointed time.

$A_i$: 1 when place appointed, else 0.

$\Theta_i$: weight of appointed place.

Weights are usually set in accordance with resource to establish a good schedule, the more important and shorter resource the bigger weight.

B. Solution design

Timetable problem with large scale and numerous constraints is a NP-hard problem which many researchers in the operational research and computer fields have been dedicated to resolve but have not found a best solution. The methods to solve timetable problem in use nowadays are random optimization algorithms, such as simulated annealing algorithm, genetic algorithm, and etc. But these random search algorithms have to check a great many feasible solutions which means huge workload, computational expense, low convergence speed. So we proposed a composite method based on heuristic algorithm, backtracking algorithm and tabu research algorithm. The effectiveness and practicability of the method have been proved by applying it to the Tsinghua University course arrangement that we can find a more satisfying solution within a shorter period.

1) Heuristic algorithm

Heuristic refers to experience-based techniques for problem solving, learning, and discovery. Heuristic methods are used to speed up the process of finding a satisfactory solution, where an exhaustive search is impractical. In more precise terms, heuristics are strategies using readily accessible, though loosely applicable, information to control problem solving in human beings and machines.

2) Tabu search

Tabu search is a metaheuristic algorithm that can be used for solving combinatorial optimization problems, such as the traveling salesman problem (TSP). Tabu search uses a local or neighborhood search procedure to iteratively move from a solution x to a solution x’ in the neighborhood of x, until some stopping criterion has been satisfied. To explore regions of the search space that would be left unexplored by the local search procedure, tabu search modifies the neighborhood structure of each solution as the search progresses. The solutions admitted to N + (x), the new neighborhood, are determined through the use of memory structures. The search then progresses by iteratively moving from a solution x to a solution x’ in N + (x).

3) Backtracking
Backtracking is an important tool for solving constraint satisfaction problems, for finding all (or some) solutions to some computational problem, that incrementally builds candidates to the solutions, and abandons each partial candidate (“backtracks”) as soon as it determines that a candidate cannot possibly be completed to a valid solution.

Backtracking depends on user-given “black box procedures” that define the problem to be solved, the nature of the partial candidates, and how they are extended into complete candidates. It is therefore a metaheuristic rather than a specific algorithm - although, unlike many other metaheuristics, it is guaranteed to find all solutions to a finite problem in a bounded amount of time. When it is applicable, however, backtracking is often much faster than brute force enumeration of all complete candidates, since it can eliminate a large number of candidates with a single test.

The backtracking algorithm enumerates a set of partial candidates that, in principle, could be completed in various ways to give all the possible solutions to the given problem. The completion is done incrementally, by a sequence of candidate extension steps.

Conceptually, the partial candidates are the nodes of a tree structure, the potential search tree. Each partial candidate is the parent of the candidates that differ from it by a single extension step; the leaves of the tree are the partial candidates that cannot be extended any further. The backtracking algorithm traverses this search tree recursively, from the root down, in depth-first order. At each node c, the algorithm checks whether c can be completed to a valid solution. If it cannot, the whole sub-tree rooted at c is skipped. Otherwise, the algorithm checks whether c itself is a valid solution, and if so reports it to the user; and recursively enumerates all sub-trees of c. The two tests and the children of each node are defined by user-given procedures.

Therefore, the actual search tree that is traversed by the algorithm is only a part of the potential tree. The total cost of the algorithm is the number of nodes of the actual tree times the cost of obtaining and processing each node. This fact should be considered when choosing the potential search tree and implementing the pruning test.

In order to apply backtracking to a specific class of problems, one must provide the data P for the particular instance of the problem that is to be solved, and six procedural parameters, root, reject, accept, first, next, and output. These procedures should take the instance data P as a parameter and should do the following:

- root(P): return the partial candidate at the root of the search tree.
- reject(P,c): return true only if the partial candidate c is not worth completing.
- accept(P,c): return true if c is a solution of P, and false otherwise.
- first(P,c): generate the first extension of candidate c.
- next(P,s): generate the next alternative extension of a candidate, after the extension s.
- output(P,c): use the solution c of P, as appropriate to the application.

The backtracking algorithm reduces then to the call bt(root(P)), where bt is the following recursive procedure:

```
procedure bt(c)
  if reject(P,c) then return
  if accept(P,c) then output(P,c)
  s ← first(P,c)
  while s ≠ Λ do
    bt(s)
    s ← next(P,s)
```

C. System implementation

1) Backtracking and tabu search

Backtracking and tabu search are used when a task cannot be arranged successfully. There are two key points in the backtracking and tabu search. First, we do not backtrack to the task former arranged, since it perhaps has nothing to do with the task at the front and maybe we cannot find any available schedule after back and back. Second, we should use limited backtracking to avoid snow slide phenomenon caused by backtracking too deep. Suppose A is the task cannot be arranged successfully, backtrack as follows:

- Rearrange task A’, with which A has an edge connected in conflict graph.
- If success, then try to arrange A again.
- If A is successfully arranged, go on to arrange the next task.
- Else rearrange task A”, with which A’ has an edge connected.
- If success, then arrange A’ and try to arrange A again when A’ successes.

To avoid the unlimited expansion in tabu search, we have to define the number of back step in advance. In the procedure described above, we set l as back step number. It means only A-connected A’ and A’-connected A” will be rearranged at most.

2) Conflict graph

To deal with the iterating judge among the teaching resources, we use conflict graph to describe and avoid the conflict connection between courses.

The following is definition of conflict graph:

- Graph={ vertex, edge }  
- Vertex={ (c_i, f_i, t_i), c_i∈course, f_i∈class, t_i∈teacher }  
- Edge={ line between vertices }  
- DenyList: pairs of unavailable time window and classroom. Each c_i has a DenyList_i.

Conceptually, conflict graph is an undirected graph, each vertex stands for an arrangement task consisted of course to be arranged and its demand on class, teacher, time, place and etc. There is an edge between two vertices when these two arrangement tasks are conflicted, they cannot be hold at the same time window. When c_i is arranged into a time window, course arrangement system will update DenyList_i (DenyList of each connected vertex c_j ) and mark all classrooms unavailable at that time window.

Two mandatory conditions, teacher-time and class-time, are satisfied through conflict graph. Since only the available resources have to be considered during conflict detecting, the
complexity of resource search reduced greatly. Construction of conflict graph are shown in Fig. 2.

3) Searching time and place
There are 2 ways during searching time and place resource, time first or place first. If the university's requirement on time is higher than on place, that is the buildings and classrooms are relatively resourceful, time first should be used. Otherwise, place first may be more helpful.

Time first search algorithm takes highest priority time of available time resources, and then finds if there is an available classroom. Time first search algorithm processes are shown in Fig. 3 and Fig. 4.

a) Search for a time resource with highest priority.

b) Search for a place resource with highest priority at this time.

c) Check on the availability of time and place.

d) If success, then mark the task arranged, else go back to step b, choose the place with lower priority, and repeat step c and step d.

e) If cannot find a place at this time, go back to step a, choose the time with lower priority, and repeat all steps.
IV. CONCLUSIONS

Course arrangement is a system engineering for unified scheduling of teaching resources in university and is a key standard to evaluate whether the operation mechanism works well.

The experience-computer-based course arrangement system has been completed and is in operation in Tsinghua University. There are more than 6500 courses in autumn semester, more than 3000 courses in spring semester and more than 560 courses in summer semester are arranged by this system. The arranged courses in each semester far exceeds that of the corresponding period in history, and students generally reflect that timetable arranged by experience-computer-based system is more satisfied. The system embodies the concept of education management reform, improves the efficiency of course arrangement in colleges and universities.

REFERENCES