Efficient Algorithm ofSplitting and Merging Classes onElective Course of College Education

Jian Liu1,*

1College of higher vocational education, Sichuan Radio and TV University, Chengdu, China
1Information center, Sichuan Huaxin Modern Vocational College, Chengdu, China

*Corresponding author e-mail: Carefree_Lee19@sohu.com

Abstract. It should take human intervention to adjust the number of each class after the students select elective courses on web by educational administration system in college, because the numbers are often less than the lower limit. This paper puts out an efficient scheduling algorithm to get the resolution of splitting and merging classes by linear time complexity computing on mathematical model. It introduces in detail the principle, the precondition, the procedure and the operation of splitting and merging classes, which illustrated with several charts, diagrams, pseudo-codes in key steps and some notes, and at last validated it by example. Then it could extend to similar issues, for example enrolling and distributing issue.

1. Introduction

There’s always an issue that the student number often exceeds the upper limit or is less than the lower limit on elective courses [1], while they are selecting those courses by educational administration management system [2]. This issue is different from the deployment on employees in plants, so the usual scheduling method [4-17] could not be put to application.

When it’s below the lower limit, it needs human intervention to adjust the numbers of these classes [3], which called the operation of splitting and merging classes. The results probably don’t match the request of the teachers and the students simultaneously and don’t make balance on teaching resources, if operated by hand which leads high error rates.

An efficient algorithm on this operation that resolved the issue by establishing mathematical model and computing accordingly is presented in this paper to save the many aspects of resources, for example human resource and time, etc.

2. Information of elective course

Elective courses, which include numerous subjects in numerous majors, for example Appreciation of English movies and songs, Interpersonal communication, Roller skating and roller ball, Free style grappling, Training of dance and form, Practical Writing, etc., orient to all students on campus, while these persons make the decision in individual units.

The information between the students, who make selection by wish and make the unknown numbers of the students of those classes, and office of academic administration, who release the subjects of elective courses and make the known numbers of those classes, don’t match each other’s that result in the situation that the number maybe exceeds the upper limit or be less than the lower limit, because it would not be known in advance. If the numbers are less than the lower limit on some
courses which result canceling and splitting of classes, the students have to make decision second time or are merged into other classes that the numbers keep normal level. So the solution of merging classes should be adopted, for avoiding selection second time inconveniently, which called operation of splitting and merging classes.

![Flowchart of splitting and merging classes]

**Figure 1.** Procedure of splitting and merging classes.

3. **Principle of splitting and merging classes**

   The operation of splitting and merging classes must comply with the principle that it should respect the students' wish at first, take advantage of the resources of teachers and classrooms, and make the times of operation as less as possible.

   The details of the principle are listed below.

   1. The student who has selected a certain course may be transferred from one class to the others if the class numbers of this course are more than one. If the class number of this course is only one, the student may be transferred from this course to other similar courses. If the class number of all similar courses is only one and the number of the students of this class is less than the lower limit, this class
should be canceled, which called splitting class, and all of the students of this class are put into the queue that has not selected any elective course yet and wait to select.

② The student should be transferred among the class of the elective courses which only offer to the grade of them, if the courses claim request for the grade.

③ If the course claims request of gender, the transfer must comply the request. For example, Free style grappling only offers to male, Training of dance and form only offers to female.

④ The staff should assign the class of the course for those students who have not made the selection this term.

4. Procedure of splitting and merging classes

Please refer to Figure 1.

5. Operation of splitting and merging classes

5.1. Precondition of splitting and merging classes

One elective course that the upper limit is assumed as ‘max’ and the lower limit is assumed as ‘min’, is picked out randomly from the several ones.

In the interval between after the selection and before the merging, the ‘max’ increases to a number big enough to receive all of the students who have selected this class. Ideally it increases to infinity, ‘ max = ∞’, which could be comprehended as no upper limit. At this moment there’s no class that has fulfilled the quota, so the all classes will not exceed the upper limit.

The number of the classes ‘c’ of this course is assumed as ‘(m+n)’, and the number of the students in each class is assumed as ‘x’, i=1, 2, ⋯, m, m+1, ⋯, m+n, that the number of class which matches ‘ min ≤ x < max ’ is ‘m’ and the number of class which matches ‘ x < min ’ is ‘n’. The ‘x’ is sorted by ascending, that it matches ‘ x ≤ x_i+1 ’.

5.2. Algorithm of splitting classes

The ‘ n ’ which matches ‘ x < min ’ becomes ‘ n ’ after the splitting, that there must be ‘ n ≤ n ’. Now, it applies circular probing that ‘ n ’ decreases from ‘ n ’ to ‘0’ or stops once it matches the formula (1) below.

\[
\sum_{i=m+1}^{m+n} (\min - x_i) \leq \sum_{i=m+n+1}^{m+n} x_i
\]  

(1)

The actual value of ‘ n ’ is computed out at this moment. Then the formulas below could be deduced from formula (1).

\[
\Rightarrow \sum_{i=m+1}^{m+n} \min - \sum_{i=m+1}^{m+n} x_i \leq \sum_{i=m+n+1}^{m+n} x_i
\]

\[\Rightarrow \sum_{i=m+1}^{m+n} \min \leq \sum_{i=m+1}^{m+n} x_i + \sum_{i=m+n+1}^{m+n} x_i \]  

(2)

\[\Rightarrow n \cdot \min \leq \sum_{i=m+1}^{m+n} x_i \]

The pseudo codes of splitting classes are listed below.

\[
tmp1 = \sum_{i=m+1}^{m+n} x_i ;
\]

\[n' = n;\]

while( n' ≥ 0 && n' × min > tmp1 )

(3) 

(4) 

(5)
\[ n \text{; at last the ‘} n \text{’ could be 0 which means the ‘} n \text{’ classes all should be split} \] (6)

Obviously, the time complexity of ‘\( \sum_{i=m+1}^{m+n} x_i \)’ is \( O(n) \), ‘\( n \)’ min ‘\( \text{‘} \)’ is \( O(1) \) and circulation of ‘while’ is \( O(1(n+1)) = O(n+1) \), then the time complexity of splitting classes is the sum of ‘\( \sum_{i=m+1}^{m+n} x_i \)’ and circulation of ‘while’ [18, 19], which means \( O(n + n + 1) = O(2n + 1) \).

5.3. Algorithm of merging classes

Presumed that there are ‘\( m \)’ classes, which match ‘\( \min \leq x_i < \max \)’ and there are ‘\( m' \)’ classes that have received the students, then there must exists ‘\( m' \leq m \)’. The method of circular probing that ‘\( m' \)’ decreases from ‘\( m \)’ to 0 or stops once the formula (2) below is matched could be put to apply for the same reason.

\[
\sum_{i=m-m'+1}^{m+n} (x_{m-m'+1} - x_i) \leq \sum_{i=m+n+1}^{m+n} x_i
\] (7)

The actual value of ‘\( m' \)’ is computed out at this moment. Then the formulas below could be deduced by formula (7).

\[
\Rightarrow \sum_{i=m-m'+2}^{m+n} x_{m-m'+1} - \sum_{i=m-m'+1}^{m+n} x_i \leq \sum_{i=m+n+1}^{m+n} x_i
\]

\[
\Rightarrow \sum_{i=m-m'+2}^{m+n} x_{m-m'+1} \leq \sum_{i=m-m'+1}^{m+n} x_i + \sum_{i=m+n+1}^{m+n} x_i
\] (8)

\[
\Rightarrow (m' + n' - 1)x_{m-m'+1} \leq \sum_{i=m-m'+2}^{m+n} x_i
\]

The pseudo codes of merging classes are listed below.

\[
\text{for(} j = 0; j <= m; ++j) \text{) (9)}
\]

\[
\text{tmp2}[j] = \sum_{i=m-m'+2}^{m+n} x_i \text{; (10)}
\]

\[
m' = m \text{; (11)}
\]

\[
\text{while(} m' \geq 0 \text{ & & } (m' + n' - 1)x_{m-m'+1} > \text{tmp2}[m']) \text{) (12)}
\]

\[
--m' \text{; (13)}
\]

// at last the ‘\( m' \)’ could be 0 which means none of the ‘\( m \)’ classes has received the students

The two lines, which are (9) and (10), of the codes above equal the five ones below.

\[
\text{tmp2}[0] = 0; \text{ (14)}
\]

\[
\text{for(} i = n; i > 1; --i) \text{) (15)}
\]

\[
\text{tmp2}[0] += x_{m-i} \text{; (16)}
\]

\[
\text{for(} i = 1; i <= m; ++i) \text{) (17)}
\]

\[
\text{tmp2}[i] = \text{tmp2}[i-1] + x_{m-i+2} \text{; (18)}
\]

Obviously, the time complexities of two circulation of ‘\( \text{for} \)’ are \( O(n-1) \) and \( O(m) \) separately, ‘\( (m' + n' - 1)x_{m-m'+1} \)’ is \( O(1) \), and ‘while’ is \( O(1(m+1)) = O(m+1) \), then the time complexity of merging classes is the sum of three circulations [18, 19], which means \( O(n-1 + m + m + 1) = O(2m + n) \).
5.4. Algorithm of adjusting number of the students

Finally, there actually is only \((m^i+n^i)\) classes to increase the number of the students, which means the number keeps invariant while \(1 \leq i \leq m-m^i\), the number increases while \(m-m^i+1 \leq i \leq m+n^i\), and the classes should be split while \(m+n^i+1 \leq i \leq m+n\), just as the Table 1.

| Class \(c_i\) | Result          |
|--------------|----------------|
| \(1 \leq i \leq m-m^i\) | Invariant number |
| \(m-m^i+1 \leq i \leq m+n^i\) | Increased the number |
| \(m+n^i+1 \leq i \leq m+n\) | Split the class |

Presumed that the number of the students is \(x_i + x'_i\) that \(x'_i\) stands for the increasing volume of this class, while \(m-m^i+1 \leq i \leq m+n^i\), then there must exists \(\sum_{i=m-m^i+1}^{m+n^i} x'_i = \sum_{i=m+n^i+1}^{m+n} x_i\). The \(x'_i\) is formed with three parts, which are filling \(y_{i1}\), dividing \(y_{i2}\) and remaining \(y_{i3}\), \(x'_i = y_{i1} + y_{i2} + y_{i3}\). The three of them are detailed in the three formulas below.

\[
y_{i1} = x_{m-m'} - x_i
\]

\[
y_{i2} = \text{INT}(f(x_i, m', n'))
\]

\[
y_{i3} = \begin{cases} 1, & \text{while } m-m^i+1 \leq i < m-m^i+1+\text{MOD}(f(x_i, m', n')) \\ 0, & \text{while } m-m^i+1+\text{MOD}(f(x_i, m', n')) \leq i \leq m+n' \end{cases}
\]

Among them, ‘\text{INT()}’ stands for rounding up function, ‘\text{MOD()}’ stands for computing modulo function, and ‘\(f(x_i, m', n')\)’ is presented below in formula (22).

\[
f(x_i, m', n') = \left( \sum_{i=m-m^i+2}^{m+n} x_i - (m'+n'-1)x_{m-m'+1} \right) / (m'+n')
\]

Because the actual value of \('m', 'n'\) and \(\sum_{i=m-m^i+2}^{m+n} x_i\) have been computed out previously, the time complexities of formula (22), ‘\text{INT}(f(x_i, m', n'))’ and ‘\text{MOD}(f(x_i, m', n'))’ are all are ‘\(O(1)\)’, which means that the time complexities of formula (19), (20), (21) are also ‘\(O(1)\)’. So the time complexity of adjusting number of students of each class is ‘\(O(1+1+1) = O(3)\)’, then the time complexities of adjusting number of students of \((m'+n')\) classes is ‘\(O(3(m'+n'))\’ [18, 19].

Now, the operation of splitting and merging of classes of this course has been finished. After it, pick out another course randomly to do the same operation till all courses have been gone through out.

According to the simple rule of \(O\) representation [20], the time complexity of whole operation is the sum of each step, which means the formula (23).
\[O(2n + 1 + 2m + n + 3(m' + n')) = O(2m + 3n + 3m' + 3n' + 1) = O(2m) + O(3n) + O(3m') + O(3n') + O(1) = 2O(m) + 3O(n) + 3O(m') + 3O(n') + O(1) = O(m) + O(n) + O(m') + O(n') = O(m + n + m' + n') \]  
(23)

From formula (23), the conclusion of the time complexity of the algorithm is linear.

6. Example

The upper limit of Practical Writing is 60, the lower is 30. There are 8 classes of this course that the numbers of the students are 52, 16, 34, 20, 24, 60, 19, 36. The process of splitting and merging is detailed in Table 2.

**Table 2. The process of splitting and merging.**

| Class \( c_i \) | \( c_1 \) | \( c_2 \) | \( c_3 \) | \( c_4 \) | \( c_5 \) | \( c_6 \) | \( c_7 \) | \( c_8 \) |
|-----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| \( x_i \)      | 60      | 52      | 36      | 34      | 24      | 20      | 19      | 16      |
| \( n \)         |         |         |         |         |         |         |         | 4       |
| Probable value of \( n' \) | 0 | 1 | 2 | 3 | 4 |
| \( n' \): min  |         |         |         |         | 0 | 30 | 60 | 90 | 120 |
| \( \sum_{j=m+1}^{m+n} x_j \) |         |         |         |         | 79 | 79 | 79 | 79 | 79 |
| Actual value of \( n' \) |         |         |         |         | 2 |
| \( m \)         |         |         |         |         | 4 |
| Probable value of \( m' \) | 4 | 3 | 2 | 1 | 0 |
| \( (m' + n' - 1)x_{m-m'+1} \) |         |         |         |         | 300 | 208 | 108 | 68 | 24 |
| \( \sum_{j=m-m'+1}^{m+n} x_j \) |         |         |         |         | 201 | 149 | 113 | 79 | 55 |
| Actual value of \( m' \) |         |         |         |         | 2 |
| Filling \( y'_1 \) |         |         |         |         | 0 | 2 | 12 | 16 |
| \( f(x_i, m', n') \) |         |         |         |         | 5/4 | 5/4 | 5/4 | 5/4 | 5/4 |
| Dividing \( y'_2 \) |         |         |         |         | 1 | 1 | 1 | 1 |
| Remaining \( y'_{i+3} \) |         |         |         |         | 1 | 0 | 0 | 0 |
| \( x'_i \) |         |         |         |         | 2 | 3 | 13 | 17 |
| \( x_i + x'_i \) |         |         |         |         | 38 | 37 | 37 | 37 | 37 |
| final number | 60 | 52 | 38 | 37 | 37 | 37 | 37 | 0 | 0 |
The numbers of all classes are detailed in Figure 2, after split and merged.

**Figure 2.** Adjusted numbers.

7. Notes

1. If there are several students who have not made their decision before merging, divide the number of them equally into those classes which matches \( \min \leq x_i < \max \) and pick out one course randomly next.

2. Usually, \( m' = 0 \) and \( n' = 0 \) will not both reasonable simultaneously, which means \( m' + n' \neq 0 \). If not so, the \( m' \) classes which matches \( \min \leq x_i < \max \) all don’t receive any student after split the \( n' \) classes which matches \( x_i < \min \). This situation could not be accepted.

Yet there must be \( m' = 0 \) while \( m = 0 \). At the same time there must be \( n' = 0 \) if exists \( \min > \sum_{i=m+1}^{m+n} x_i \). Then there must be \( m' + n' = 0 \), which indicates the sum of the numbers of the students of the \( n' \) classes which matches \( x_i < \min \) is less than the lower limit. After split them all, there is none of the classes which matches \( \min \leq x_i < \max \) could receive the students. The result is to either cancel this course or to remain only one class which is less than the lower limit by special case.

Under this condition, this course and similarities should be taken care of first, then pick out one course randomly next. The reason is to avoid the situation that it makes some students who have made selection unselected after cancelled the course.
The upper limit ‘max’ and the lower limit ‘min’ are usually set by experience, which says that they would be gotten from the practices in long term. The computing and operation of splitting and merging will decrease in a large number theoretically, if they have been set as rational values.

8. Conclusion

The operation of splitting and merging should be cut down as less as possible, but it will always be inevitable in reality. Then applying the algorithm in this paper can save the recourses of time and labor, instead of operating by hand, to upgrade the efficiency and accuracy.

The algorithm in this paper was abstracted from the type of these issues, for example enrolling into several classes in training or allocating the players into several groups in competition. This algorithm will be suitable for them, if some numbers are less than lower limit that the classes or groups should be split and merged.

9. Acknowledgments

This work was financially supported by Project of Department of education, Sichuan Province, China (13ZB0367).

References

[1] Wang Feng, Lin Jie, Model of college time-table problem based on graph theory. Computer Engineering and Applications, 2009, 45(27): 240-242.
[2] Ma Yan, Wang Wenfa, Li Hongda. Backtracking algorithm for assigning dormitory problem based on matrix storage. Computer Engineering and Design, 2009, 30(12): 2022-2025.
[3] Zhang Xiaoli, Wang Wenjie, Zhao Geng. Establishing courses section of timetabling system with cluster algorithm. Journal of Beijing Electronic Science and Technology Institute, 2008, 16(4): 4-8.
[4] Heng Hongjun, Jia Wenqi, Zheng Bingzhong. Study on crew scheduling method based on algorithm combination. Computer Applications and Software, 2011, 28(2): 279-301.
[5] Zhang Bixi, Zhang Wanli, Song Jing. Ant colony-based labor allocation method for small-and-medium-size manufacturing workshops. Industrial Engineering Journal, 2011, 14(4): 119-123.
[6] Wang Aiping, Zhu Yongjun, Zhang Gongying, et al. Call center labor resources allocation based on ant colony algorithm. Computer Technology and Development, 2009, 19(3): 202-207.
[7] Li Jufang, Tan yuejin. The guided local search method for a class of workforce scheduling problems. Systems Engineering, 2004, 22(9): 87-91.
[8] Fukunaga A, Hamilton E, Fama J, et al. Staff Scheduling for inbound call centers and customer contact centers // Proceedings of the National Conference on Artificial Intelligence. Edmonton, Alberta, Canada: [s.n.], 2002, 23: 822-829.
[9] Baptista L, Marques-Silva J P. Using randomization and learning to solve hard real-world instances of satisfiability // Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming Lecture Notes In Computer Science[s. n.]: Springer Berlin, 2000: 489-494.
[10] Marques-Silva J P. Algebraic simplification techniques for propositional satisfiability // Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming Lecture Notes In Computer Science[s. n.]: Springer Berlin, 2000: 537-542.
[11] Ertay T, Ruan D. Data envelopment analysis based decision model for optimal operator allocation in CMS. European Journal of Operational Research, 2005, 164(3): 800-810.
[12] Yang T, Chen M C, Hung C C. Multiple attribute decision-making methods for the dynamic operator allocation problem. Mathematics and Computers in Simulation, 2007, 73(5): 285-299.
[13] Wang L, Zhen D Z. An effective hybrid heuristic for flow-shop scheduling. International Journal of Advanced Manufacturing Technology, 2003, 21(1): 38-44.
[14] Ponnambalam S G, Mohan R M. A GASA multi-objective hybrid search algorithm for integrating lot sizing and sequencing in flow-line scheduling. International Journal of Advanced Manufacturing Technology, 2003, 21(6): 126-137.
[15] Sawik T. Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers. Mathematical and Computer Modeling, 2000, 31(13): 39-52.
[16] Kuo Y, Yang T. Optimization of mixed-skill multi-line operator allocation problem. Computers and Industrial Engineering, 2007, 53(3): 386-393.
[17] Sen C G, Cinar G. Evaluation and pre-allocation of operators with multiple skills: A combined fuzzy AHP and max-min approach Expert Systems with Applications, 2010, 37(3): 2043-2053.
[18] Weimin Yan, Weimin Wu, Data structure: C Language, Beijing, Tsinghua University Press, 2007:14-17.
[19] Thomas H, Cormen Charles E, Leiserson Ronald L, et al. Edited by Jianping Yin, Yun Xv, Gang Wang, et al. Introduction to Algorithms, third ed., Beijing, China Machine Press, 2013:27-30.
[20] Kyle Loudon. Edited by Xiang Xiao, Ge Chen. Mastering Algorithms with C, Beijing, China Machine Press, 2012:42-43.