EE 3980 Algorithms

Homework 8. Selecting Courses

105061110 周柏宇

2020/5/10

1. Introduction

In this homework, we want to select courses using only greedy method to maximize the credits we get. Meanwhile, the selected courses should not overlap on the schedule. We will discuss the time and space complexity. Furthermore, we will try to examine the optimality and uniqueness of our solution.

2. Analysis & Implementation

In this homework, I try to establish the connection between course selection problem and maximization problem of independent systems. Hopefully, we can apply the general algorithm for the latter to solve our problem.

2.1 Independent systems, Matroid & Maximization

Let S be a finite set and $\mathcal{I} = \{X : X \subseteq S\}$, then the set system $(S, \mathcal{I})$ is an independent system if

(i) $\emptyset \subseteq \mathcal{I}$;

(ii) $If\ Y \in \mathcal{I}\ and\ X \subseteq Y\ then\ X \in \mathcal{I}.$

Furthermore, an independent system $(S, \mathcal{I})$ is a matroid if

(iii) $If\ X, Y \in \mathcal{I}\ and\ |X| > |Y|,$

$then\ there\ is\ an\ x \in X \backslash Y\ such\ that\ Y \cup \{x\} \in \mathcal{I}.$

Also, a maximization problem of independent systems is described as follows:

$Given\ an\ independent\ system\ (S, \mathcal{I})\ and\ the\ weight\ function$

$w: S \to \mathbb{R}^+, find\ an\ X \in \mathcal{I}\ such\ that\ w(X) = \sum_{x \in X} w(x)\ is\ maximized.$

Using the principle of the greedy method, that is,

"making the locally optimal choice with the intent of finding a global maximum", we can come up with the best-in greedy algorithm.

```
1. // Given (S,I) and w:S → R+
2. // find X ∈ I such that w(X) is maximum.
3. // Input: (S,I) and w
4. // Output: X
5. Algorithm BestInGreedy(S, I, w)
6. {
7.     Sort S into nonincreasing order by w;
8.     X := ∅; // initialize to empty set
9.     for each x ∈ S in order do { // try all elements
10.          // maintain independence then add
11.          if (X ∪ {x} ∈ I) then {
12.              X := X ∪ {x};
13.          }
14.      }
15.      return X;
16. }
```

However, the optimality of the solution is not guaranteed unless the

independent systems $(S, \mathcal{I})$ is also a matroid.

2.2 Selecting Courses

For this problem, we are given the information of courses, including class time, credits, etc. We try to select courses that does not occupy the same time on the schedule and maximize the total credits. For this homework, we are only allowed to use greedy method to obtain the solution.

First, if we let

$S = \{all\ courses\}$ and $\mathcal{I} = \{\text{all feasible selection of courses}\}$, by feasible selection we mean any course selected does not overlap, then $(S, \mathcal{I})$ is indeed an independent system, since

(i) Not selecting any course is allowed.

(ii) If Y is a feasible selection, then $X \subseteq Y$ will still be feasible.

Furthermore, the course selection problem is a maximization problem of independent system if we let the weight function be the credits of a course (all positive number). Therefore, the high-level operation of best-in greedy algorithm for our problem will be as below.

```
1. // Given course information, e.g. class time and credits
2. // find feasible selection of courses
3. // such that sum of credits is maximum.
4. // Input: array of course information
5. // output: array selected
6. Algorithm BestInGreedy(courses, selected) {
```

```
7.     Sort courses into nonincreasing order by their credits;
8.     selected := ∅;
9.       // try all courses
10.     for each course ∈ courses in order do {
11.          // add the course if the set stays feasible
12.          if (selected ∪ {course} is feasible) then {
13.              selected := selected ∪ {course};
14.          }
15.     }
16.     return selected;
17. }
```

In the *BestInGreedy* algorithm, line 7 is a sorting operation, it will take up to $\mathcal{O}(n^2)$ if using insertion sort, where n is the number of courses available. The loop will execute $\Theta(n)$ times and we need to check the feasibility of the union for every iteration. Checking the feasibility will be simply looking up the class time we are about to occupied. Therefore, the exact number will be the number of classes of the course, which can be bounded by $6 = \mathcal{O}(1)$, assuming that no course has classes more than 6.

To sum up, the sorting operation will not take less than $\mathcal{O}(n)$ and the whole loop are bounded by $\mathcal{O}(n)$. Thus, the total time complexity of *BestInGreedy* algorithm is determined by the time complexity of sorting operation. In my implementation, it is $\mathcal{O}(n^2)$.

As for the space complexity, assuming sorting does not take additional space, then we only need space for course information and the array *selected*.

Just to keep the necessary information: class time and credits, it will take $\Theta(n) \cdot \mathcal{O}(6+1) = \Theta(n)$ and $\mathcal{O}(n)$ for array *selected*. Therefore, the overall space complexity is $\mathcal{O}(n)$, where n is the number of courses available.

Unfortunately, our independent system is not a matroid. Because two feasible selection of courses X, Y and $|X| > |Y|$, say X has a course x that Y does not select, then $Y \cup \{x\}$ may not be feasible. This implies the algorithm does not guaranteed the optimality.

3. Result and Observation

The result we found using *BestInGreedy* algorithm is following.

```
Total credits: 37
Number of courses selected: 12
  1: MATH102006 4 T3T4R3R4 Calculus (II)
  2: MATH202001 4 T1T2F1F2 Advanced Calculus II
  3: EE211000 3 T7T8R7 Modern Physics
  4: EE214001 3 M3M4W2 Electromagnetism
  5: EE231000 3 M1M2R1R2 Introduction to Programming
  6: EE313000 3 W5W6R8R9 Optics and Photonics
  7: EE335000 3 W3W4F4 Introduction to Solid-State Electronic
Devices
  8: EE336000 3 M7M8M9 Opto-electronic Devices
  9: EE364000 3 M5M6RnR5 Communication Systems (I)
  10: EE413500 3 T5T6F3 Principle of Lasers
  11: EECS340000 3 RaRbRc Satellite Electrical System Design
  12: EE240500 2 W7W8W9 Embedded System Laboratory
Weekly schedule:
    1 2 3 4 n 5 6 7 8 9 a b c
  M V V V V . V V V V V . . .
  T V V V V . V V V V . . . .
  W . V V V . V V V V V . . .
  R V V V V V V . V V V V V V
  F V V V V . . . . . . . . .
```

Although we cannot proof its optimality, it has been found that the solution achieving total credits 37 is not unique. Here's another solution.

Total credits: 37
Number of courses selected: 12
1: EE465000 2 W7W8W9 Communications System Laboratory
2: EE231000 3 M1M2R1R2 Introduction to Programming
3: EE364000 3 M5M6RnR5 Communication Systems (I)
4: EE413500 3 T5T6F3 Principle of Lasers
5: EECS202002 3 W5W6R8R9 Signals and Systems
6: EECS302000 3 M7M8R6 Introduction to Computer Networks
7: EECS303002 3 W3W4F4 Probability
8: EECS340000 3 RaRbRc Satellite Electrical System Design
9: ENE553000 3 T7T8T9 Terahertz Science and Technology
10: MATH242000 3 M3M4W2 Algebra II
11: MATH102007 4 T3T4R3R4 Calculus (II)
12: MATH202002 4 T1T2F1F2 Advanced Calculus II
Weekly schedule:

	1	2	3	4	n	5	6	7	8	9	a	b	c
M	V	V	V	V	.	V	V	V	V	.	.	.	.
T	V	V	V	V	.	V	V	V	V	V	.	.	.
W	.	V	V	V	.	V	V	V	V	V	.	.	.
R	V	V	V	V	V	V	V	.	V	V	V	V	V
F	V	V	V	V	.	.	.	.	.	.	.	.	.