CS 341 - Algorithms **Winter 2018**

Lecture 7: January 25, 2018

Lecturer: Bin Ma *Notes By: Harsh Mistry*

7.1 Greedy Algorithms Continued

7.1.1 Interval Scheduling

Given a input list of intervals $I_k = [S(I_k), f(I_k)]$. Find a subset of intervals with the maximum cardinality, such that no two intervals in the subset intersect

An optimized solution to select the interval that ends the earlies, as by choosing the one that finishes the earliest, it resserves more room for future events. Additionally, it combines the earliest start and shortest strategies into one.

Algorithm FinishTimeFirst :L Greedily choose the next non-conflicting interval

Proof : To prove this we can use an "Exchange Proof". You start with an optimal solution that is not the greedy solution. then you compare them and identify the difference, Then you modify the optimal solution to make it increasingly "greedier", without increasing the cost.

7.1.2 Interval Colouring

Given n intervals $[s_i, f_i]$. Colour them in k different colors, so that no two intervals with the same color overlap. Minimize k.

Algorithm :
For each interval in the start time order :
Color it with the smallest non-conflicting colour ID

7.1.3 Fractional Knapsack

Input: n items, each has weight w_i and per unit weight value v_i; a capacity W.
Output : an amount $0 \leq x_i \leq w_i$ for each i, such that $\sum_{i=1}^{n} x_i \leq W$ and $\sum_{i=1}^{n} x_i v_i$ us maximized.

since we can allow to take a fraction of each item, so it makes sense to take the most valuable item first. Let us assume that the items are sorted according to per unit weight value.

Greedy Algorithm

```
• Remaining $W \leftarrow W$
• For i from 1 to n
    – If (remaining $W > 0$)
        * $x_i \rightarrow \min\{remainingW, w_i\}$
```

    - remaining$W \rightarrow w - x_i$

  else : $x_i \rightarrow 0$

7.1.4 Stable Matching

Gale Shapely Algorithm

- While there is an unpaired manager M_i
  - Let M_i propose to his most favourite student whom he never tried before
  - if student is not paired then : Pair them
  - Else if student prefers the new manager over his old manager
    - Pair the student with the new manager, and turn old manager unpaired.