091M4041H - Assignment 3

胡鹏飞 计控学院 201828007329009

1. Greedy Algorithm

Given a list of n natural numbers d1, d2,...,dn, show how to decide in polynomial time whether there exists an undirected graph G = (V; E) whose node degrees are precisely the numbers d1; d2;...; dn. G should not contain multiple edges between the same pair of nodes, or “loop" edges with both endpoints equal to the same node.

* 1. **Describe the basic idea in natural language AND greedy-choice property**

首先，图G的边数为e，那么图G所有顶点的度数之和为2e，即必然为偶数，这是构成简单图的充要条件，所以先看d1,d2,……dn之和是否为偶数。若为偶数，则继续判断，使用贪心算法的话，很容易可以想到将d1,d2……dn按照从大到小的顺序进行排列，若排序之后第一个为dk，若dk大于你，则必然存在环或者重复边，若dk小于n则继续判断，此时可以认为这个顶点与其后dk个顶点有连接，所以其后dk个顶点每个都需要减1，减完之后重新排序，重复以上操作，若中间某个顶点度数小于0，则必然存在孤立顶点，无法构成无向图，直至最后一个顶点，若为0，则可以构成无向图。

* 1. **Pseudo-code**

**ExistGraph(**d**[**1……n**])**

**if((**d1**+**d2**+**…**+**dn**)%**2**=**1**)** **then**

**return** **False**

**end** **if**

**while(**n**)**

sort**(**d**)**

d1 **=** d**[**0**]**

**if** d1 **>** n **then**

**return** **False**

**end** **if**

**for** i **=** 1 to d1 **then**

d**[**i**]** **-=** 1

**if** d**[**i**]** **<** 0 **then**

**return** **False**

**end if**

**end** **for**

d**[**0**]** **=** 0

n **-=** 1

**end** **while**

**if(**d**[**0**]!=**0**)** **then**

**return** **False**

**else**

**return** **True**

* 1. **Prove the correctness**

首先，算法排除了三种情况，这三种情况都是不能构成简单图的情况，上述已经说明。其次，算法在循环中每次排序选取最大度数的点，这样就可以使后面有足够多的点可以抵消该点的度数，使该顶点连接其他d1个节点，之后就不再考虑该顶点，然后在剩下的顶点中找度数最大的顶点，给它分配第二大的度数，即与d2个顶点连接，如此类推，直到分配完所有度数，综上所述，算法无误。

* 1. **Analyse the complexity**

首先，需要循环的次数最多为n次，复杂度为O(n)，每次循环里需要排序，复杂度为O(nlogn),所以总的时间复杂度为：



1. Greedy Algorithm

Given two strings s and t, check if s is subsequence of t?

A subsequence of a string is a new string which is formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (ie, "ace" is a subsequence of "abcde" while "aec" is not).

* 1. **Optimal substructure and DP equation**

根据贪心，目标串前面的字符匹配的位置越靠前，源串后面的字符越多，匹配上的概率越大，所以定义s和t两个字符串的索引，如果发现t的索引值和s的索引值相等，移动子串s的索引，继续比对下一个目标索引值，如果最后s的索引和s的长度相等，那么s即为t的子串。

* 1. **Pseudo-code**

**IsSubsequence(**s**,**t**)**

sIndex **=** 0**;**

tIndex **=** 0**;**

**while** **(**sIndex **<** s**.**length**()** **&&** tIndex **<** t**.**length**())**

**if** t**[**tIndex**]** **==** s**[**sIndex**]** **then do**

sIndex**++**

**end** **if**

tIndex**++**

**end** **while**

**if** sIndex **!=** s**.**length**()** **then**

**return** **False**

**else**

**return** **True**

* 1. **Prove the correctness**

两个字符串的索引意味着将问题分解成了子问题，子问题中是新的源串和目标串，在源串中找到了目标串的第一个字符，那么在源串中继续找到目标串下一个字符的概率就会增大，就成了下一个子问题，后面的子问题中不需要考虑之前的子问题的内容，即后面的子问题不需看到之前的问题，算法无误。

* 1. **Analyse the complexity**

算法要遍历一次源串t，即



4 Greedy Algorithm

Given a rope whose length is n, please cut the rope to m parts to get the maximum product of the length of each part Πl1+l2+::+lm=n l1 \* l2 \* …\* lm.For example, if a rope with length 8, when we cut it to 2, 3, 3, we can get the maximum product 18.

**4.1 Optimal substructure and DP equation**

在剪绳子中，如果绳子的长度大于5，则每次剪出的长度为3的绳子。如果剩下的长度仍然大于5，则接着剪出一段长度为3的绳子，重复这个步骤，即尽可能多的剪出长度为3的绳子，直到剩下的长度小于5，当剩下的长度为4时，把绳子剪成2的绳子。

**4.2 Pseudo-code**

**FindMaxPreouct(**length**)**

**if** length **<** 2 **then**

**return** 0

**end** **if**

**if** length **==** 2 **then**

**return** 1

**end** **if**

**if** length **==** 3 **then**

**return** 2

**end** **if**

m **=** length **/** 3

**if** length **-** m **\*** 3 **==** 1

m **-=** 1

**end** **if**

n **=** **(**length **-** m **\*** 3**)** **/** 2**;**

**return** pow**(**3**,** m**)\***pow**(**2**,** n**);**

**4.3 Prove the correctness**

首先，当n=1,2,3时，问题很容易求解；当n=4时，剪出一个3和一个1不如两个2乘积大；当n>=5,3(n-3)>=2(n-2)>n，即分出3比分出2乘积要大，更大于不分，所以应该不断分出3，这样乘积最大，length整除3后余数如果是1，说明多剪了一个3，应该拿出来一个3（即m-1）和剩下的1组成4,算法无误。

* 1. **Analyse the complexity**

因为只对length进行简单运算，所以时间复杂度为：

