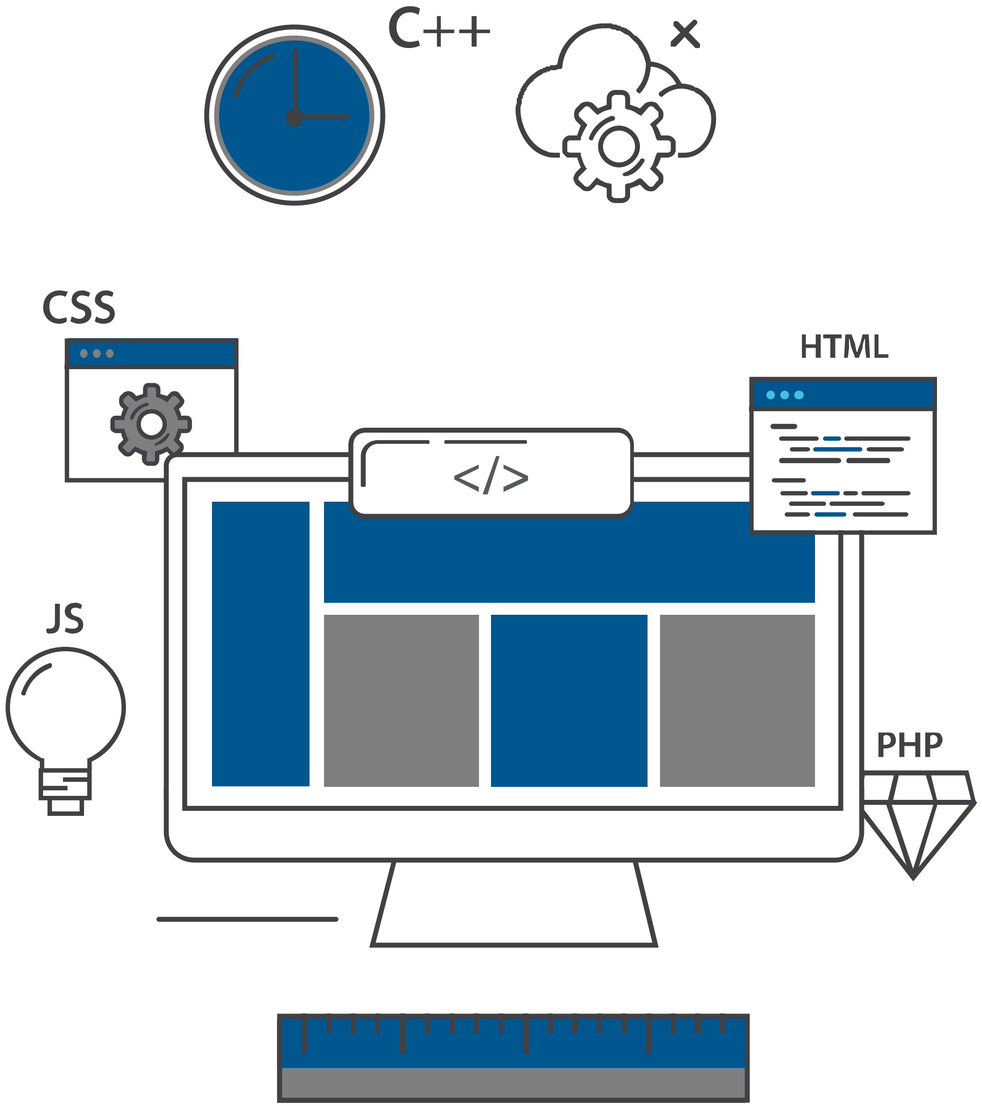
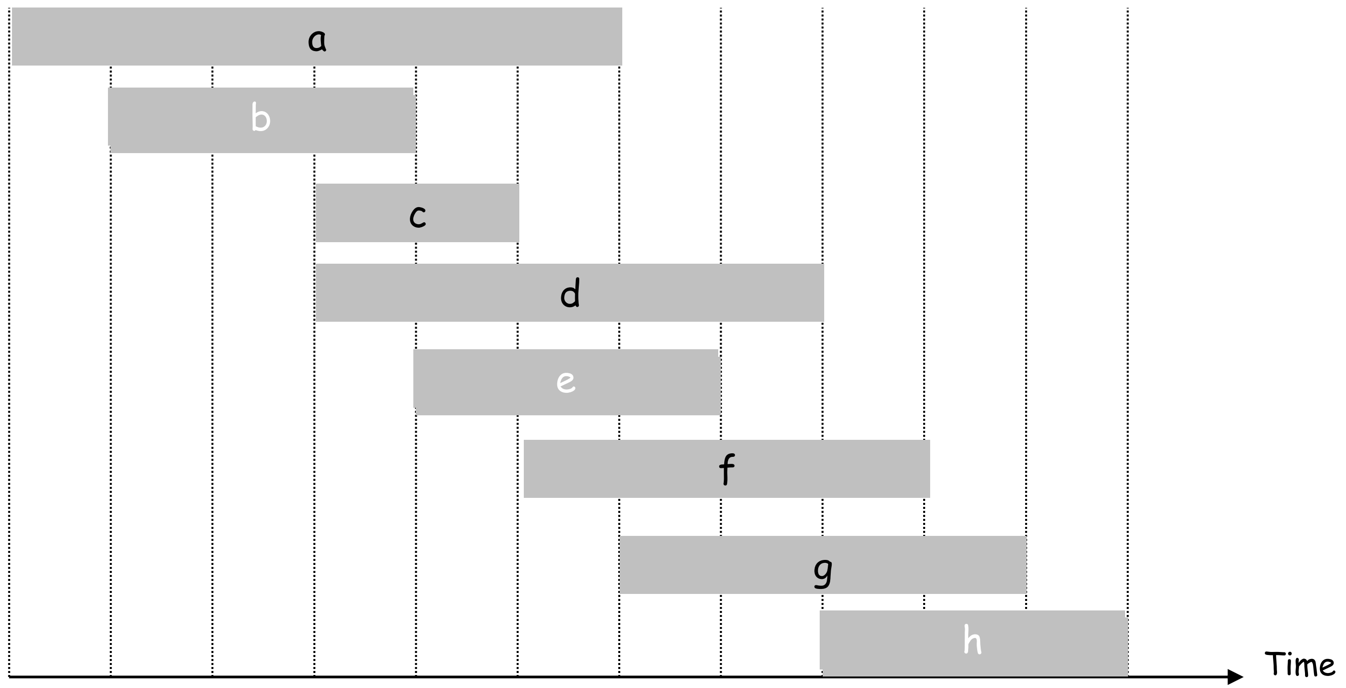
《算法导论》

李 翔
https://implus.github.io/



1.2.1 区间调度问题



1.2.1 区间调度问题

- 教材：可以用贪心算法来求解这个问题。

解题分析：

对这个问题，如果使用贪心算法的话，可能有以下几种考虑：

(1)、每次选取开始时间最早的；

(2)、每次选取结束时间最早的；

(3)、每次选取用时最短的；

(4)、在可选工作中，每次选取与最小可选工作有重叠的部分；

对于上面的四种算法，只有算法(2)是正确的，其它的三种都可以找到相应的反例。具体证明如下：

数轴上有n个区间，选出最多的区间，使得这些区间不互相重叠。

算法：

将所有区间按右端点坐标从小到大排序，顺序处理每个区间。如果它与当前已选的所有区间都没有重叠，则选择该区间，否则不选。

证明：

显然，该算法最后选出的区间不互相重叠，下面证明所选出区间的数量是最多的。设f_i为该算法所接受的第i个区间的右端点坐标，g_i为某最优解中的第i个区间的右端点坐标。

命题1.1 **当i>=1时，该算法所接受的第i个区间的右端点坐标f_i<=某最优解中的第i个区间的右端点坐标g_i。**

该命题可以运用数学归纳法来证明。对于i=1，命题显然为真，因为算法第一个选择的区间拥有最小右端点坐标。令i>1，假定论断对i-1为真，即f_{i-1}<=g_{i-1}。则最优解的第i个可选区间所组成的集合包含于执行该算法时第i个可选区间所组成的集合；而当算法选择第i个区间时，选的是在可选区间中右端点坐标最小的一个，所以有f_i<=g_i。证毕。

设该算法选出了k个区间，而最优解选出了m个区间。

命题1.2 **最优解选出的区间数量m=该算法选出的区间数量k。**

假设m>k，根据命题1.1，有f_k<=g_k。由于m>k，必然存在某区间，在g_k之后开始，故也在f_k之后开始。而该算法一定不会在选了第k个区间后停止，还会选择更多的区间，产生矛盾。所以m<=k,又因为m是最优解选出区间个数，所以m=k。

综上所述，算法选出的区间是最优解。

1.2.1 区间调度问题

- 教材：可以用贪心算法来求解这个问题。

解题分析：

对这个问题，如果使用贪心算法的话，可能有以下几种考虑：

(1)、每次选取开始时间最早的；

(2)、每次选取结束时间最早的；

(3)、每次选取用时最短的；

(4)、在可选工作中，每次选取与最小可选工作有重叠的部分；

对于上面的四种算法，只有算法(2)是正确的，其它的三种都可以找到相应的反例。具体证明如下：

前提不够自然，不够现实

数轴上有n个区间，选出最多的区间，使得这些区间不互相重叠。

算法：

将所有区间按右端点坐标从小到大排序，顺序处理每个区间。如果它与当前已选的所有区间都没有重叠，则选择该区间，否则不选。

证明：

显然，该算法最后选出的区间不互相重叠，下面证明所选出区间的数量是最多的。设f_i为该算法所接受的第i个区间的右端点坐标，g_i为某最优解中的第i个区间的右端点坐标。

命题1.1 **当i>=1时，该算法所接受的第i个区间的右端点坐标f_i<=某最优解中的第i个区间的右端点坐标g_i。**

该命题可以运用数学归纳法来证明。对于i=1，命题显然为真，因为算法第一个选择的区间拥有最小右端点坐标。令i>1，假定论断对i-1为真，即$f_{i-1}<=g_{i-1}$。则最优解的第i个可选区间所组成的集合包含于执行该算法时第i个可选区间所组成的集合；而当算法选择第i个区间时，选的是在可选区间中右端点坐标最小的一个，所以有$f_i<=g_i$。证毕。

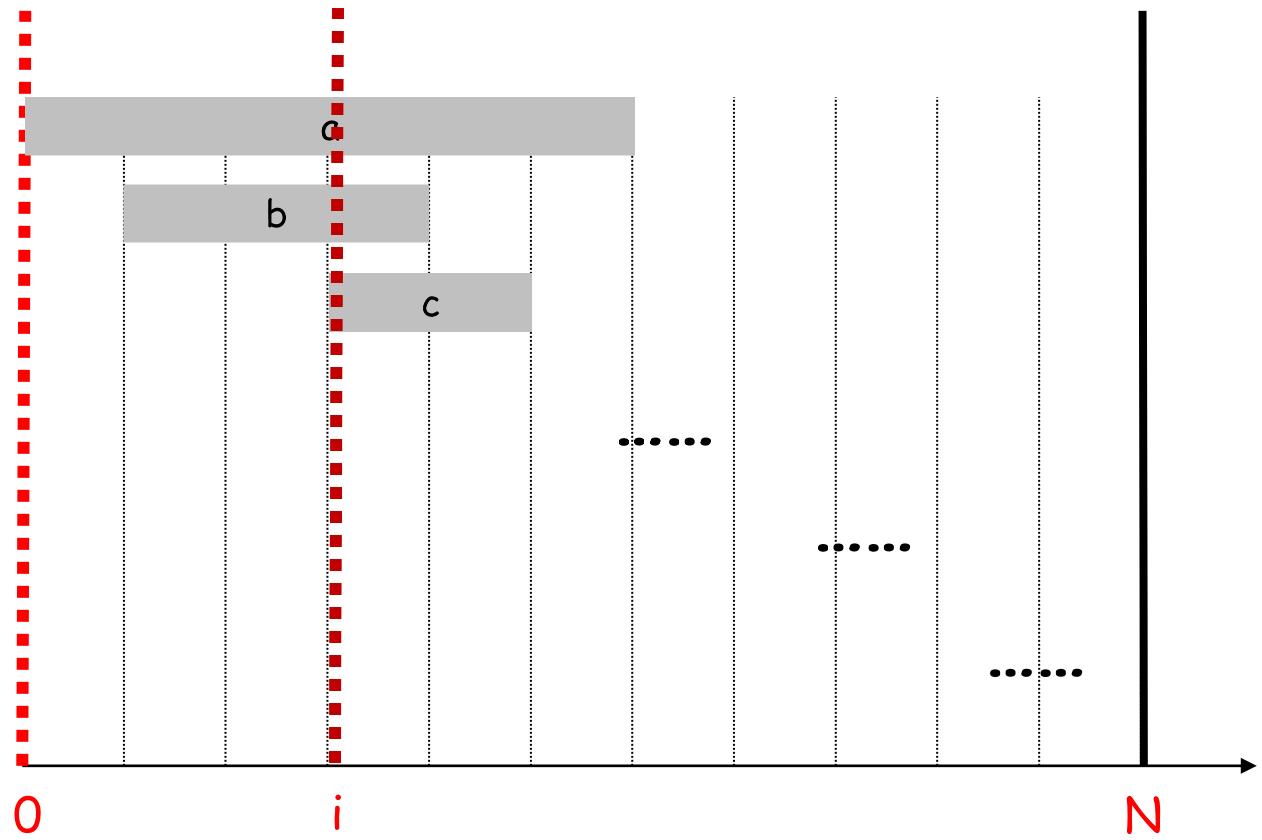
设该算法选出了k个区间，而最优解选出了m个区间。

命题1.2 **最优解选出的区间数量m=该算法选出的区间数量k。**

假设m>k，根据命题1.1，有$f_k<=g_k$。由于m>k，必然存在某区间，在g_k之后开始，故也在f_k之后开始。而该算法一定不会在选了第k个区间后停止，还会选择更多的区间，产生矛盾。所以m<=k,又因为m是最优解选出区间个数，所以m=k。

综上所述，算法选出的区间是最优解。

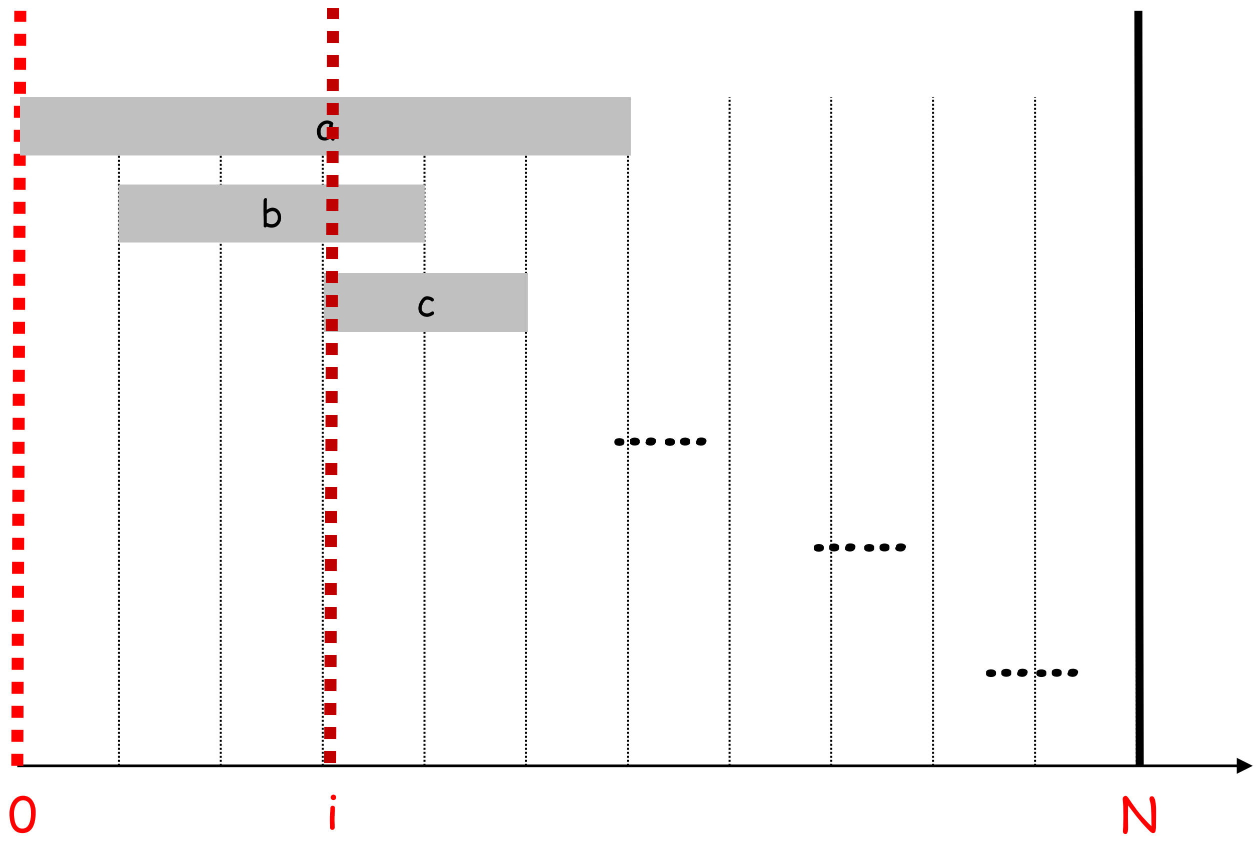
递归子问题：根据规模定义问题



F[i, N] 代表[i, N]范围的可行最优解　本问题的解就是F[0, N]

递归子问题：问题规模缩小

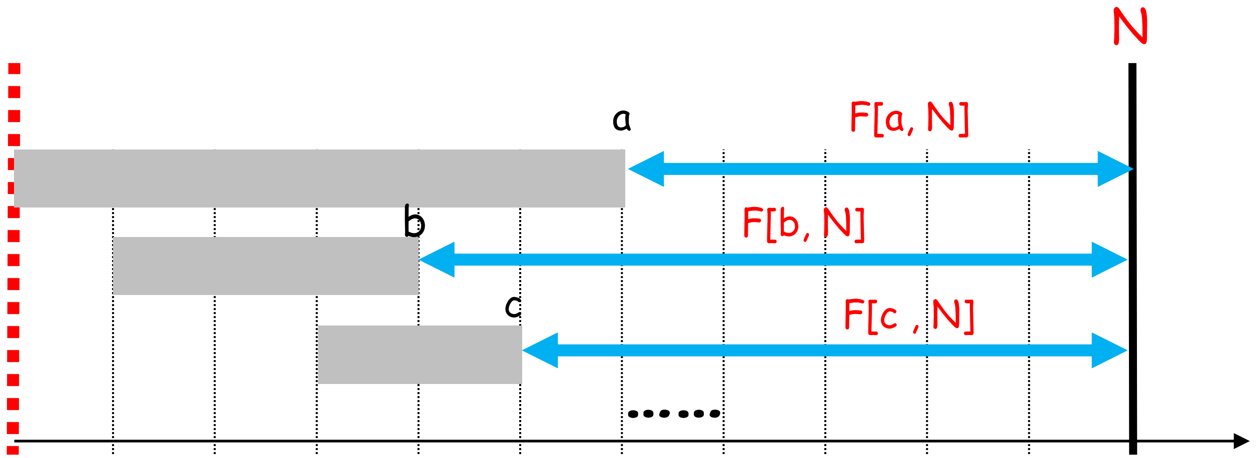
如何缩小规模？尝试纳入一个具体的选择



对于F[i, N]来说，枚举在它隶属范围内[i, N]的（有可能成为最终解的）第一个选择

递归子问题：问题规模缩小

枚举在它隶属范围内[0, N]的（有可能成为最终解的）第一个选择



F[0, N] = max{

1 + F[a, N],

1 + F[b, N],

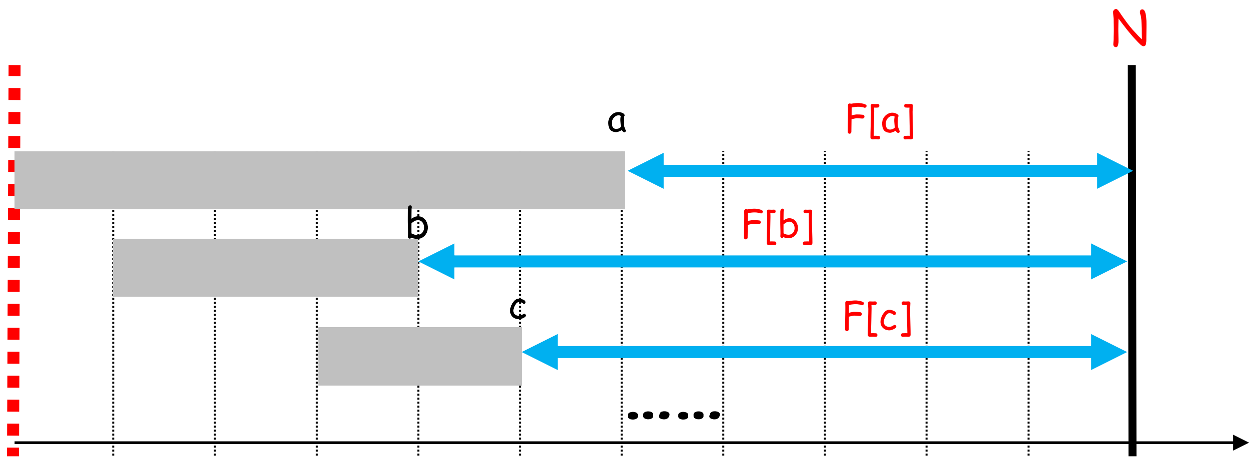
1 + F[c, N],

……

}

F[i, N] <==> F[i]

递归子问题：问题规模缩小



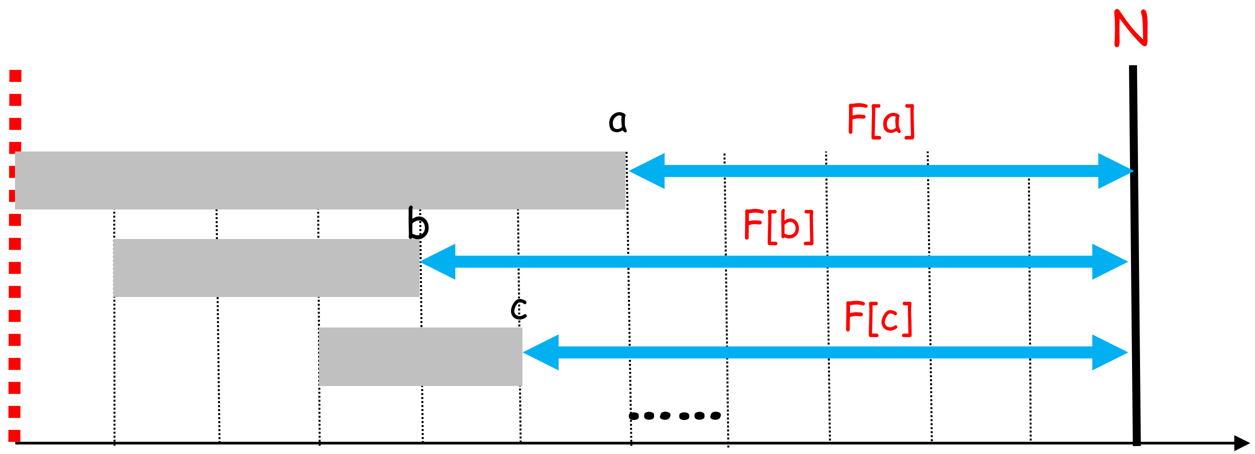
F[0] = max{
1 + F[a],
1 + F[b],
1 + F[c],
……
}

F[b] >= F[c] >= F[a]

发现一个规律!

1+F[b] >= 1+F[c] >= 1+F[a]

递归子问题：问题规模缩小

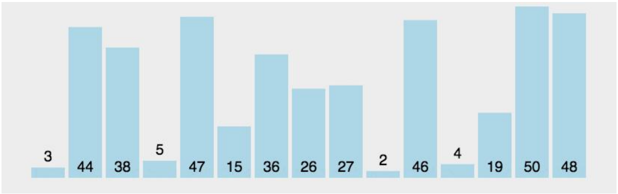


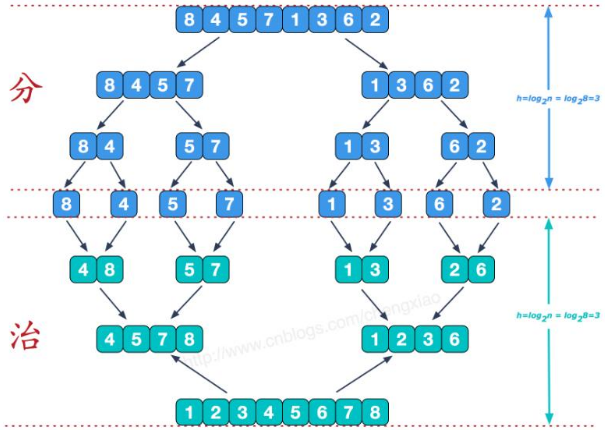
F[0] = max{
1 + F[a],
1 + F[b],
1 + F[c],
……
}

F[0] = 1 + F[b]，b为其隶属区间结束时间最早的一个区间解毕。因此本题可用贪心法求解。

两种排序对比

冒泡排序	归并排序





冒泡排序	归并排序
28次	16次
$O(n^2)$	$O(n\log n)$

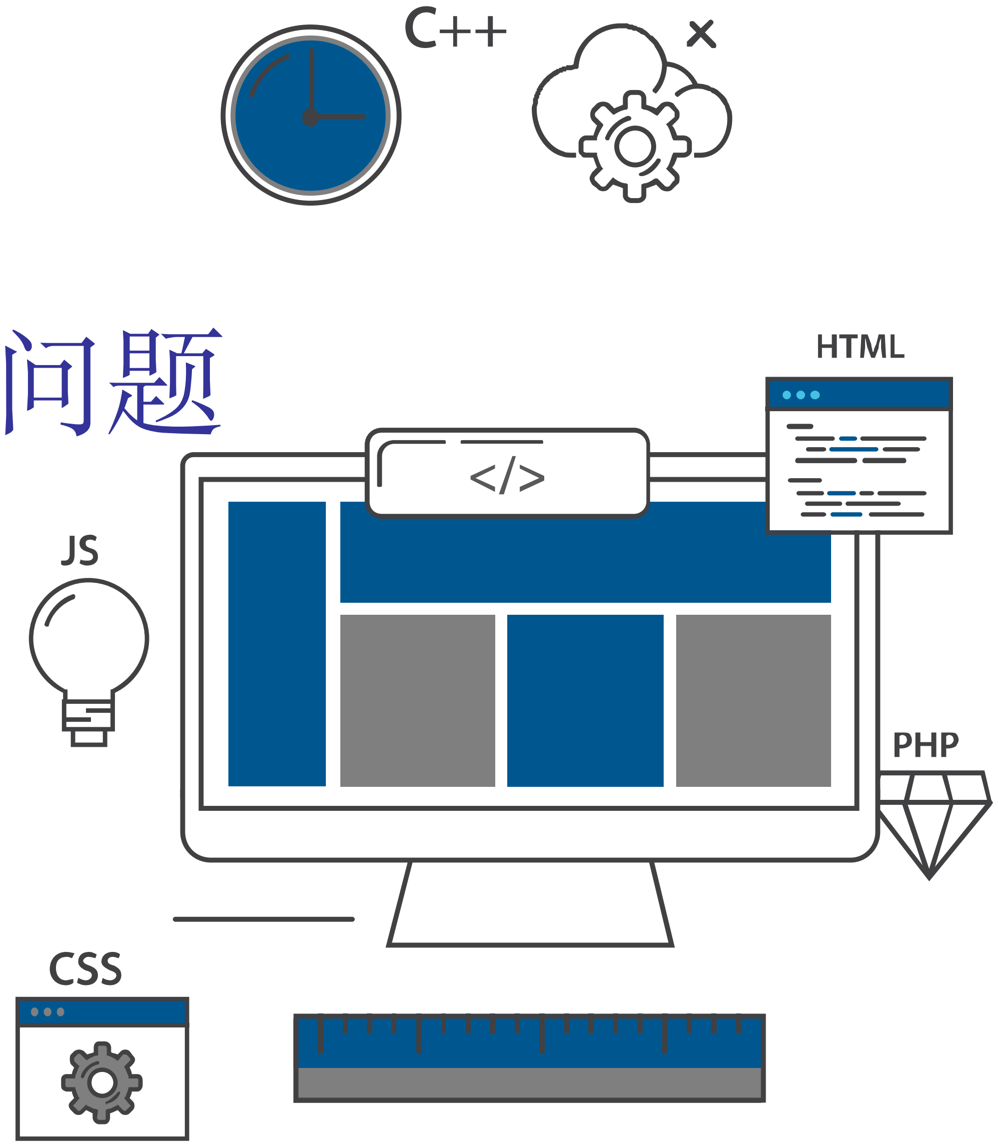
为什么以**递归子问题**为核重要

- 最小规模的子问题非常容易
  - 子问题子到一定程度（规模为1或2时）非常容易，相当于数学上1+1=2的难度；
- 由子问题扩大规模到原问题非常清晰
  - 递归子问题的思想有助于帮助我们理解和挖掘该问题的**结构与性质**；
- 统一的思考本质
  - 不需要死记硬背各种算法，大一统理论

第一章 引言：某些典型的问题

李 翔

https://implus.github.io/



问题的背景

- 1. 存在一些医院，以及一些即将从医学院毕业的学生。医院招学生的时候，会有自己的倾向性；同样的学生也有自己的喜好。
- 经过一系列的招聘选择以后，称申请者x和医院y是不稳定的，如果
  - ✓ x更喜欢y, 而不喜欢目前分配的医院。
  - ✓ y更喜欢x, 而不喜欢目前分配给它的学生。

问题的背景

- 稳定的分配方案

如果分配方案中没有前面所述的不稳定的(x,y)出现。

这样的定义比较符合真实，而且也能够保证尽可能的满足双方的选择。

1.1 问题

- 考虑n个男人的集合： $M=\{m_1,m_2,\ldots,m_n\}$,

  以及n个女人的集合： $W=\{w_1,w_2,\ldots,w_n\}$ 。

  令M*W 表示所有可能的形如(m,w)的有序对的集合，其中$m\in M, w\in W$. 一个匹配S是来自M*W的**有序对的集合**，并且有如下性质：每个M的成员和每个W的成员至多出现在S的一个有序对中。

1.1 问题

- 一个完美匹配S’是具有如下性质的匹配：M的每个成员和W的每个成员恰好出现在S’的一个队里。
- 优先的概念: 每个男人$m \in M$ 对所有的女人排名， 如果m给w的排名高于w’,称 m偏爱w超过w’.
- 于是每个男人对女人有一个排名->**优先表**； 类似的每个女人也有一个优先表。

举个例子理解概念

M*W = {(m1,w1),(m1,w2),(m2,w1),(m2,w2)}；

一个**匹配S**是指 = {(m1,w2)} 或 {(m1,w2),(m2,w1)};

一个**完美匹配S’**是指 = {(m1,w2),(m2,w1)};
或者 = {(m1,w1),(m2,w2)}

1.1 问题

- 一个具体的优先表例子： M={Xavier,Yancey,Zeus}; W={Army,Bertha,Clare}

	1st (favorite)	2nd	3rd (least favorite)
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Men's Preference Profile

	1st (favorite)	2nd	3rd (least favorite)
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

Women's Preference Profile

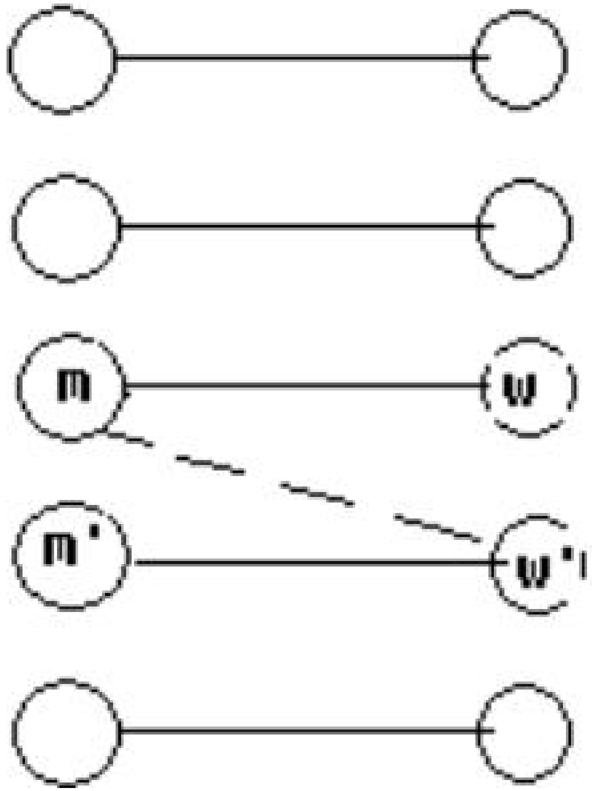
1.1 问题

- 稳定匹配
- 不稳定因素
- 给定一个完美匹配S, 在S中存在两个对(m,w)和(m’,w’), 如果m更偏爱w’而不爱w, <u>而且</u>w’更偏爱m而不爱m’.

---称(m,w’)是一个相对于S的***不稳定因素***：(m,w’)不属于S, 但是m和w’双方都偏爱另一方而不爱他们在S中的伴侣。

1.1 问题

- Figure 1.1 具有不稳定元素(m,w’)的完美匹配S



1.1 问题

- **目标**就是一个不含有不稳定因素的匹配(舞会，婚姻)集合
- 我们说一个匹配S是稳定的，如果
  - ✓ 匹配S是完美的
  - ✓ 不存在相对于S的不稳定因素

举个例子理解概念

M*W = {(m1,w1),(m1,w2),(m2,w1),(m2,w2)};

一个**匹配S**是指 = {(m1,w2)} 或 {(m1,w2),(m2,w1)};

一个**完美匹配S’**是指 = {(m1,w2),(m2,w1)};
或者 = {(m1,w1),(m2,w2)}

一个**稳定匹配S’’**是指不存在不稳定因素的**完美匹配S’**

例子

- 考虑n=2, M={m,m'};W={w,w'}
  - ✓ m更偏爱w而不爱w'
  - ✓ m'更偏爱w而不爱w'
  - ✓ w更偏爱m而不爱m'
  - ✓ w'更偏爱m而不爱m'

那么(m,w), (m',w')构成了唯一的稳定匹配。

---(m,w'), (m',w) 不是

例子

- 考虑n=2, M={m,m'};W={w,w'}
  - ✓ m更偏爱w而不爱w'
  - ✓ m'更偏爱w'而不爱w
  - ✓ w更偏爱m'而不爱m
  - ✓ w'更偏爱m而不爱m'

稳定匹配是什么?

(m,w),(m',w');

(m,w'),(m',w)

例子

- n=3,如前所述的优先表
- X-C, Y-B, Z-A 是稳定匹配吗?

	1st (favorite)	2nd	3rd (least favorite)
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Men's Preference Profile

	1st (favorite)	2nd	3rd (least favorite)
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

Women's Preference Profile

例子

- 不是，Xavier-Bertha是更好的配对

	1st (favorite)	2nd	3rd (least favorite)
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Men's Preference Profile

	1st (favorite)	2nd	3rd (least favorite)
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

Women's Preference Profile

例子

- X-A, Y-B, Z-C 是稳定匹配吗?

	1st	2nd	3rd
	favorite		least favorite
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Men's Preference Profile

	1st	2nd	3rd
	favorite		least favorite
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

Women's Preference Profile

Yes!!!

算法

- 初始，每个人都是自由的。一个**自由**的男人m选择他的优先表上排名最高的女人w,发起邀请，那么 (m,w)进入**中间状态：约会**。
- 如果又有另一个男人m’发起邀请，那么女人w决定,选择m,还是m’. 如果m优先，那么约会状态不变。否则(m’,w)变成约会状态，m变成自由状态。
- 循环往复；最后，当没有人处于自由状态，那么所有的约会被定为最后的结果，返回最终的匹配。

算法

- 邀请-拒绝算法. [Gale-Shapley 1962] 找到稳定匹配**符合直觉**的算法

```
Initialize each person to be free.
while (some man is free and hasn't proposed to every woman) {
    Choose such a man m
    w = 1st woman on m's list to whom m has not yet proposed
    if (w is free)
        assign m and w to be engaged
    else if (w prefers m to m')
        assign m and w to be engaged, and m' to be free
    else
        w rejects m
}
```

分析算法

- 命题4 如果男人m在算法执行的某点是自由的，那么存在一个他还没有发出过邀请的女人。【反证法】

- 命题5 终止时，G-S算法返回的集合S是一个完美匹配。【反证法】

分析算法

- 命题6 考虑G-S算法的一次执行，它返回一个集合S, 那么S是一个稳定匹配。
- 证明：假设S中存在一个不稳定因素，(m,w),(m',w'); 但是m偏爱w', w'偏爱m.

那么m最后一次邀请向w发出；在此之前, m向w'一定发出过邀请，但是被w'拒绝，意味着w' 对m'的偏好要高于m，与假设（w'偏爱m）不符。

分析算法

观察

- 命题1 w从接受第一次邀请开始保持约会状态，与她约会的一系列伴侣(依照w的优先表)越来越好。
- 命题2 m提出邀请的一系列女人(按照m的优先表)变得越来越差。

分析算法

- 命题3 G-S算法在至多n^2次While循环的迭代后终止。
- 证明： 需要定义一个**逐步进展**的度量。

单个自由人的数目不合适；

参与约会的对数也不合适；

定义P(t): 迭代t结束时，m已经向w发出过邀请的那些(m,w)的集合。

可知P(t)大小严格递增。且(m,w)只存在n^2种可能。

实现算法

- 女人如何判断接收/拒绝邀请?
  女人对自己的优先表做预处理，反向变换；
  这样以后判别的时候就是**常数阶**的代价；

Amy	1st	2nd	3rd	4th	5th	6th	7th	8th
Pref	8	3	7	1	4	5	6	2

Amy	1	2	3	4	5	6	7	8
Inverse	4th	8th	2nd	5th	6th	7th	3rd	1st

```
for i = 1 to n
   inverse[pref[i]] = i
```

Amy prefers man 3 to 6
since `inverse[3]` < `inverse[6]`

2 7

推广

- 具有多个稳定匹配的实例.
  - A-X, B-Y, C-Z.
  - A-Y, B-X, C-Z.

	1st	2nd	3rd
Xavier	A	B	C
Yancey	B	A	C
Zeus	A	B	C

	1st	2nd	3rd
Amy	Y	X	Z
Bertha	X	Y	Z
Clare	X	Y	Z

- G-S算法生成的是哪一个?

A-X, B-Y, C-Z

推广

- 关注的问题：
- G-S算法的执行步骤与自由的男人的选择有关，如果**选择不同**，那么G-S算法所有的执行会得到同样的匹配吗？
- 对！

推广

- 所有的执行得到**同样的匹配**!
- *寻找匹配的唯一特征*
- 如果存在一个**稳定匹配**包含了(m,w)对，我们就说女人w是男人m的有效伴侣。如果w是m的有效伴侣，且没有别的在m的排名中比w更高的女人是他的有效伴侣，那么w就是m的最佳有效伴侣，记为best(m)。

推广

- 对男人而言，G-S算法是理想的。
- 那么，是不是对女人就不是那么有利了呢?
- 类似的，如果存在一个稳定匹配包含了(m,w)对，我们就说男人m是女人w的有效伴侣。如果w是m的有效伴侣，且没有别的在w排名中比m更低的男人是她的有效伴侣，那么m就是w的最差有效伴侣，记为worst(w)。

推广

命题1.8 在稳定匹配S*中每个女人与她最差的有效伴侣配对.

证明：用反证法，假设在S*中存在一对匹配(m, w)使得m不是w的最差有效伴侣，则存在稳定匹配S，w的优先表上比m排名靠后的男生匹配.在S中，m与另一个女生匹配，但该女生并不是最佳有效伴侣，即m更偏爱w，但此时的匹配有不稳定的因素,m会去追求w，而 w会去接受m。因此S是不稳定的，与假设矛盾。所以在稳定匹配S*中每个女人与她最差的有效伴侣配对。

- 命题8 在稳定匹配S*中每个女人与她最差的有效伴侣配对。

https://blog.csdn.net/qq_43210583/article/details/107329257

- 暗示了一种现象：

对于任何输入，G-S算法中发出邀请的一方(根据他们的优先表)以**最佳可能**的稳定匹配结束；而另外一方却以**最差可能**的稳定匹配结束。

Proof：假设在匹配S中存在一对情侣m，w，其中m不是w的最差有效伴侣，假设m'才是w的最差有效伴侣

那么存在S*使得在S*中，w和一个她更不喜欢的男孩m'匹配在了一起——**w更喜欢m而不是m'**

此时m和谁匹配在一起了呢？我们继续假设和某个女孩w'匹配在了一起

由第六点的证明可知，因为在S中m，w被匹配在了一起，所以w是m的最佳有效伴侣——**m更喜欢w而不是w'**

这又在S*中产生了一个不稳定因素——w可能和m发生婚外情。

因此和假设相悖，反证结束。

推广

- 但是不要忘了一种平衡:
- ✓ w有主动选择的权利
- ✓ 命题一： w越来越好
- ✓ 命题二： m越来越差