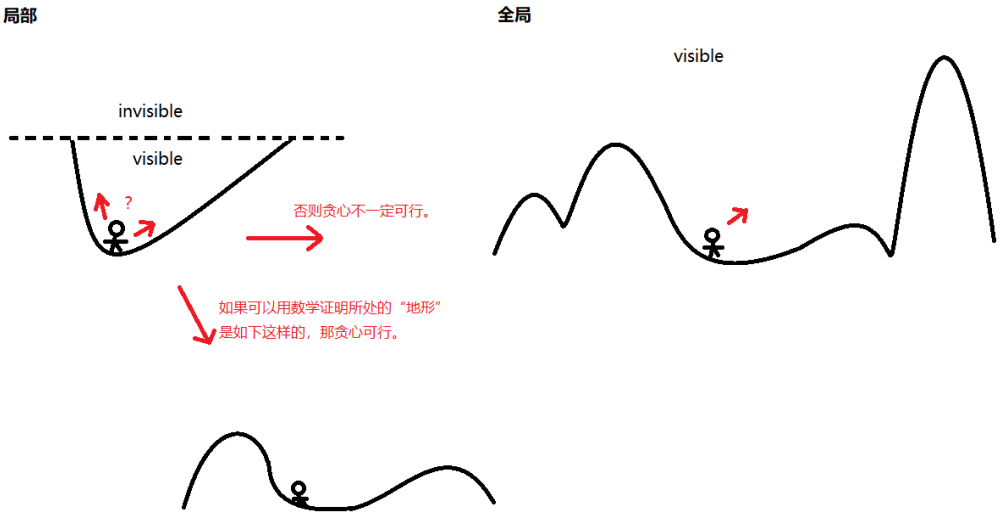
STUACM 第五次集训

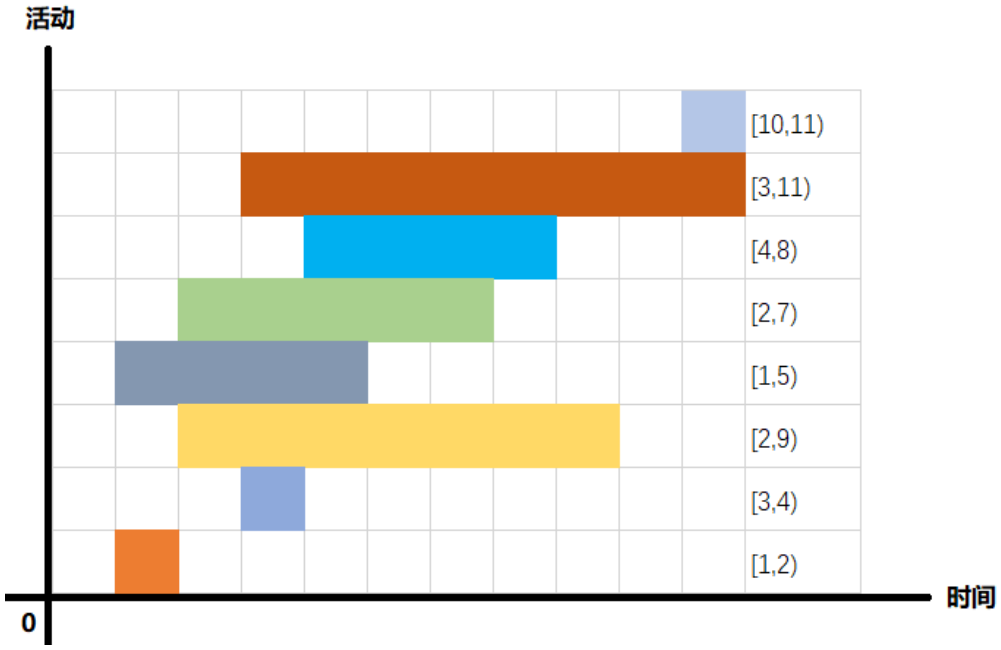
2019/11/23

- 初窥贪心思想
  - 贪心选择

    贪心选择是指所求问题的**整体最优**解可以通过一系列**局部最优**的选择，即贪心选择来达到。这是贪心算法可行的第一个基本要素，也是贪心算法与动态规划算法的主要区别。贪心选择是采用从顶向下、以迭代的方法做出相继选择，每做一次贪心选择就将所求问题简化为一个规模更小的子问题。对于一个具体问题，要确定它是否具有贪心选择的性质，我们*必须证明每一步所作的贪心选择最终能得到问题的最优解*。通常可以首先证明问题的一个整体最优解，是从贪心选择开始的，而且作了贪心选择后，原问题简化为一个规模更小的类似子问题。然后，用*数学归纳法*证明，通过每一步贪心选择，最终可得到问题的一个整体最优解。
  - 形象理解



  - 例子
    - 贪心可行：背包容量为V，从n个物品中做选取，每个物品的重量均为w，拥有各自的价值v_i，求可以获得的最大价值。
    - 贪心不可行：背包容量为V，从n个物品中做选取，每个物品拥有各自的重量w_i和各自的价值v_i，求可以获得的最大价值。
  - 总结步骤：
    - 建立数学模型来描述问题；
    - 把求解的问题分成若干个子问题及确定贪心策略；
    - 证明用该贪心策略能保证局部最优一定能得到全局最优；若无法证明：
      1. 更改步骤二的子问题划分方式或贪心策略；
      2. 该问题不适用贪心思想，改用其他思路；
    - 对每一子问题求解，得到子问题的局部最优解；
    - 把子问题的解局部最优解合成原来解问题的一个解。
  - 经典例题
    - 活动选择问题
    - 建立数学模型：



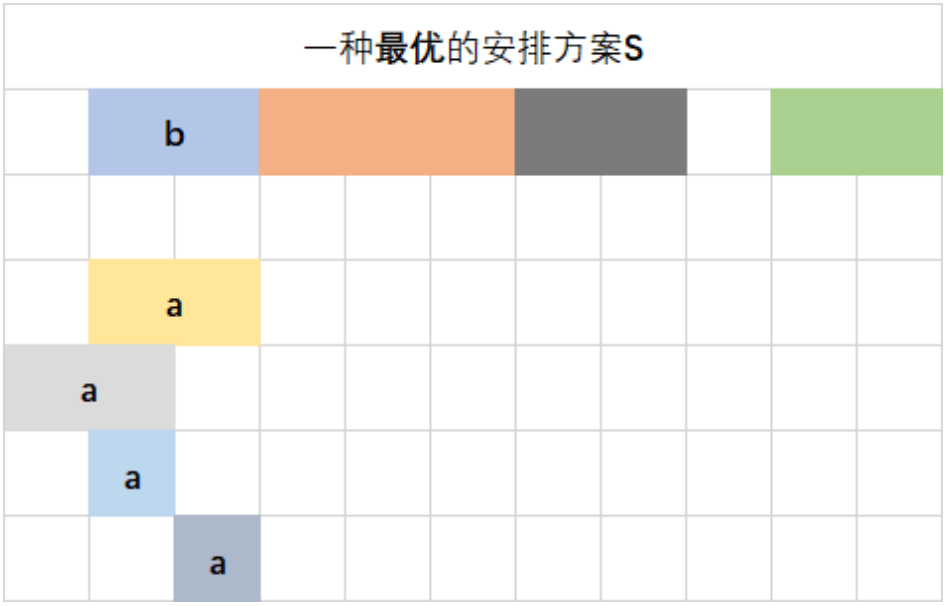
求区间[1,R]上，不相交区间的个数的最大值。

- 确定贪心策略及化简为子问题的方式
  - 贪心策略：优先选择最早结束的活动(若有活动同时结束，优先选择最早开始的)。
  - 化简为子问题的方式：假设当前问题为“求区间[L,R]上，不相交区间的个数的最大值”，当前贪心选择的活动为[l,r)，则子问题为“求区间[r,R]上，不相交区间的个数的最大值”
- 证明贪心策略的正确性

1. 证明**第一步**贪心选择总是正确的：

   假设对于给定的所有活动，其中最早结束的活动(若有活动同时结束，优先选择最早开始的)是a $[l_0, r_0]$；已知一种**最优**的安排方案S，使得区间$[1, R]$上，不相交区间的个数的最大值为M。

   假设该方案S中，最早结束的活动(若有活动同时结束，优先选择最早开始的)是b $[l_1, r_1)$，



   1. $a == b$，a**被包含**在最优方案中，则第一步贪心选择是对的；
   2. $a != b$，a**不被包含**在最优方案中，由假设知a与b存在如下关系：
      1. $l_0 < l_1, r_0 == r_1$，那么方案S中把b换成a，形成新的方案S'，结果不会变差；

2. $l_0 == l_1, r_0 < r_1$，那么方案S中把b换成a，形成新的方案S'结果不会变差；
3. $l_0 < l_1, r_0 < r_1$，那么方案S中把b换成a，形成新的方案S'结果不会变差；

综上，第一步贪心选择总是对的。

2. 数学归纳法证明，在子问题中贪心选择是正确的

每次贪心选择会把问题化简为子问题，而在区间变小的子问题中，同样变成了证明“第一步贪心选择总是正确的”。与1证明同理。

- 对每一子问题求解，得到子问题的局部最优解；把子问题的解局部最优解合成原来解问题的一个解。

```
#include <iostream>
#include <algorithm>
using namespace std;
const int mxn = 100 * 1000 + 10;
struct activity {    // 表示一个活动占用的区间[l,r)
    int l, r;
};
activity As[mxn];    // 存储输入的所有活动
bool cmp(activity & a, activity & b) {    // 按贪心策略的标准进行比较
    if (a.r == b.r) return a.l < b.l;
    return a.r < b.r;
}
int main()
{
    int n;
    while (cin >> n) {    // 多组输入
        for (int i = 0; i < n; ++i) {    // 输入
            cin >> As[i].l >> As[i].r;
        }
        // 对n个活动按贪心选择标准排序
        // 排序复杂度为O(nlogn)
        sort(As, As + n, cmp);
        int ans = 0;
        int l = 0;
        for (int i = 0; i < n; ++i) {    // 按贪心策略逐个选择
            // 当选择了一个活动后，其他冲突的活动不再考虑
            if (As[i].l >= l) {    // 压缩区间，变成子问题
                ++ans;
                l = As[i].r;
            }
        }
        cout << ans << endl;
    }
}
```

- 贪心例题　　参考博客
  - 分发饼干
    - 题意：已知一些孩子和一些糖果，每个孩子有需求因子g，每个糖果有大小s，当某个糖果的大小s>=某个孩子的需求因子g时，代表该糖果可以满足该孩子，求使用这些糖果，最多能满足多少孩子（注意，某个孩子最多只能用1个糖果满足）
  - Remove K Digits

        - 题意：已知一个使用字符串表示非负整数num，将num中的k个数字移除，求移除k个数字后，可以获得的最小的可能的新数字(num不会以0开头，num长度小于10002)
    - 摆动序列