EDA 软件设计 I

Lecture 20

EDA软件设计 I

贪心算法

Greedy Algorithm

- **基本概念：什么是贪心策略（Review)**
- **活动选择问题**
- **贪心算法「可找到最优解」的条件**

EDA软件设计 I

Review:贪心策略

- **贪心算法/策略：**“总是做出在当前时刻最好的选择，无论后续出现什么情况，它都不会重新考虑这一决策”
- **期望：**局部最优选择将导致全局最优解
- **优势：**简单、高效
- **局限：**
  - 不是对于所有问题，方法都能保证有效（可以找到全局最优解）

EDA软件设计1

Review:找零问题

- **找零问题:**
  - 假设要找零27元，手头上有的零钱面值有20元、10元、5元和1元
  - **目标**：使用最少的纸币来满足27元的找零需求

- **贪心策略:**
  1. 选择当前能够找的最大面额的纸币：从20元开始，因为这是小于（或等于）27元且面额最大的零钱
  2. 更新总额：已找20元，还需要找零7元
  3. 重复此过程直到需要找零的总额为0

能够得到最优解

EDA软件设计

Review:摘果子问题

- **水果收集问题**
  - 假设你去果园摘苹果、橙子、葡萄。每种水果的价值不同，苹果每斤5元，橙子每斤6元，葡萄每斤4元，你带的篮子只能装10斤的水果
  - **目标**：在不超过10斤的前提下，收集尽可能多的总价值

- **贪心策略：按照价值密度从高到低选择水果，直到篮子装满为止**
  - 第一步：先选价值密度最高的橙子（6元/斤）装
  - 如果篮子还有容量，装完橙子后，再依次选择密度第二高的苹果（5元/斤），最后选择葡萄

通常不能得到最优解

活动选择问题

- **问题描述：**给定一组活动，每个活动都有一个开始时间 s[i] 和结束时间 f[i]（其中 s[i] < f[i]）。活动选择问题要求从这些活动中选择一个子集，使得：
  1. 选出的活动互相之间不重叠（即：一个活动的结束时间不得晚于另一个活动的开始时间）
  2. 选出的活动数量最多

435. 无重叠区间

活动编号	开始时间 s[i]	结束时间 f[i]
1	0	6
2	1	2
3	3	4
4	5	7
5	5	9
6	8	9

贪心设计的关键?

Greedy on：“每次选择结束时间最早且与之前已选择的活动不冲突的活动”

ED软件设计1

活动选择问题

- **贪心算法步骤：**
  1. **对活动进行排序**
     - 按活动的结束时间 f[i] **从小到大排序**，确保优先选择结束时间更早的活动
  2. **初始化选择**
     - 选择排序后的第一个活动（即结束时间最早的活动），将其加入结果集合
     - 记录下当前活动的结束时间 last_end_time，来判断后续活动是否与当前活动冲突（时间重叠）
  3. **遍历活动**
     - 从排序后的第二个活动开始，逐一检查每个活动：
       - 如果当前活动的开始时间 s[i] 大于或等于 last_end_time，则选择该活动
       - 更新last_end_time为当前活动的结束时间 f[i]
  4. **返回结果集合**

EDA软件设计1

活动选择问题

1．排序活动：按照结束时间从小到大排序后，活动编号顺序为2,3,1,4,5,6

活动编号	开始时间 s[i]	结束时间 f[i]
1	0	6
2	1	2
3	3	4
4	5	7
5	5	9
6	8	9

活动编号	开始时间 s[i]	结束时间 f[i]
2	1	2
3	3	4
1	0	6
4	5	7
5	5	9
6	8	9

EDA软件设计

活动选择问题

活动编号	开始时间 s[i]	结束时间 f[i]
活动 2	1	2
活动 3	3	4
活动 1	0	6
活动 4	5	7
活动 5	5	9
活动 6	8	9

2. 初始化选择

- 选择第一个活动（活动2），将其加入结果集合
- 设置last_end_time = 2 (f[2] = 2)

3. 遍历其余活动：

- 活动 3：s[3] = 3 >= last_end_time = 2，选择活动 3，更新 last_end_time = 4
- 活动 1： s[1] = 0 < last_end_time = 4，跳过
- 活动 4：s[4] = 5 >= last_end_time = 4，选择活动 4，更新 last_end_time = 7
- 活动 5：s[5] = 5 < last_end_time = 7，跳过
- 活动 6：s[6] = 8 >= last_end_time = 7，选择活动 6，更新 last_end_time = 9

4. 返回结果： set（2,3,4,6)

EDA软件设计

活动选择问题：贪心策略是最优解吗？

- **时间轴: 0---1---2---3---4---5---6---7---8---9**
- **活动 2: [1---2]**
- **活动 3: [3---4]**
- **活动 1: [0----------6]**
- **活动 4: [5-------7]**
- **活动 6: [8---9]**
- **活动 5: [5----------9]**

如何证明贪心算法为最优解?
① 暴力求解
② 数学归纳法

EDA软件设计

贪心算法一定能找到最优解的核心条件

1. 贪心选择性质（Greedy Choice Property）

2. 最优子结构性质 （Optimal Substructure Property）

EDA软件设计

贪心选择性质

- **定义**：通过每一步的局部最优选择，可以逐步构造出全局最优
- **理解本质**：
  - “每次做出的局部选择（当前看起来最优的选择）一定是全局最优解的一部分”
  - “每一步的局部最优选择，能为后续问题留下最多的可能性，从而不会错过全局最优解”

EDA软件设计I

活动选择问题：贪心选择性质

- **问题描述**
  - 给定 n 个活动，每个活动有开始时间 s[i] 和结束时间 f[i]
  - 活动集合为 A = {a1, a2, ..., an}
  - 活动之间不能重叠（一个活动的开始时间必须晚于或等于前一个活动的结束时间）
  - 目标是选择尽可能多的活动，使得它们互不冲突
- **贪心策略：每次选择结束时间最早且与之前已选择的活动不冲突的活动**

贪心选择性质直观解释：

① **为后续活动留出更多时间**：

选择结束时间最早的活动，比选择其他活动能更早地释放时间段，从而为后续的活动安排预留更多的时间

② **不影响最优解：**

假设最优解中包含某个活动 b，但 b 的结束时间比 a1 晚，且 a1 不在最优解中，那么用 a1 替换 b，不会减少解的总活动数，且可能使后续的活动数量更多

EDA中心设计

严格证明：活动选择问题的贪心选择性质

证明方法：反证法

1. 假设：

- 令 S 是所有与当前已选择活动不冲突的活动集合
- a1 是 S 中结束时间最早的活动
- 假设存在某个最优解 O，它没有选择 a1，而是选择了另外一个活动 b （$b \in S$ 且 $b \neq a1$）

2. 比较 a1 和 b 的结束时间

- 因为 a1 是 S 中结束时间最早的活动，所以 $f[a1] \leq f[b]$

3. 替换操作：

- 因为 a1 和 b 都在集合 S 中，说明它们与当前已选择的活动集合不冲突
- 如果用 a1 替换最优解 O 中的活动 b，仍然可以得到一个合法的解（即这些活动仍然互不冲突）

4. 替换后的解仍然是最优解：

- 替换后得到的新解包含了 a1，而未减少解的活动总数（因为只是用 a1 替换了 b）
- 因此，替换后的解也是一个最优解

5. 矛盾：

- 假设中说 O 是最优解，但它不包含 a1。替换后得到的解也是最优解，这与“O 是最优解但不包含 a1”的假设矛盾

要证明：

- **每次选择最早结束的活动可以构造出全局最优解。即：全局最优解里面一定包含每一步骤中最早结束的活动**

EDA软件设计

最优子结构性性质

- **定义：一个问题的最优解包含其子问题的最优解**
- **理解本质：**
  - 将问题分解为若干子问题后，子问题的最优解可以直接组合成原问题的最优解
  - 这一性质是动态规划和贪心算法的共同基础，但动态规划不要求“局部最优选择”成立，而贪心算法则更严格

自顾设计