



第一章 引言：某些典型的问题

苏 明



1.1 第一个问题： 稳定匹配

- 双向选择，缺少强制干预的实例：
- ✓ Raj 接收到来自电信公司CluNet的暑期工作；
- ✓ 另一家公司WebExodus也给他一个Offer；
- ✓ Raj自己想去WebExodus而不是CluNet；



问题的提出

- ✓ 于是Raj就去了WebExodus;
- ✓ 原来的CluNet空出来一个暑期实习位置;
- ✓ CluNet招了一个新人, Bob; Bob 更喜欢CluNet, 因此取消了原先的公司Babelsoft的录用;
- ✓ 于是Babelsoft空出来一个位置
- ✓
- ✓ 情况连续失控



问题的提出

- 如果申请者，公司都有自己的**优先**选择列表，如何在申请者，公司之间找到一种选择的平衡？
- 最后是一种什么样的局面，如何让最后的选择“**稳定**”下来？



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



问题的背景

- 2. 1962, David Gale, Lloyd Shaply, 两位数理经济学家, 当时在《纽约人》读到一个关于学校入学处理错综复杂的事情, 于是引起了对稳定匹配研究的兴趣。



1.1 问题

- 问题的形式化

提出本质的、相对简单模型：

n 个申请人中的每个人对 n 个公司提出申请，
每个公司只要单一的申请人；

按照**Gale-Shapley**的思路，考虑等价的不同性别的匹配问题；



1.1 问题

- 考虑 n 个男人的集合： $M = \{m_1, m_2, \dots, m_n\}$ ，
以及 n 个女人的集合： $W = \{w_1, w_2, \dots, w_n\}$ 。
令 $M \times W$ 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 $M \times W$ 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。

1.1 问题

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

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

Men's Preference Profile

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

Women's Preference Profile

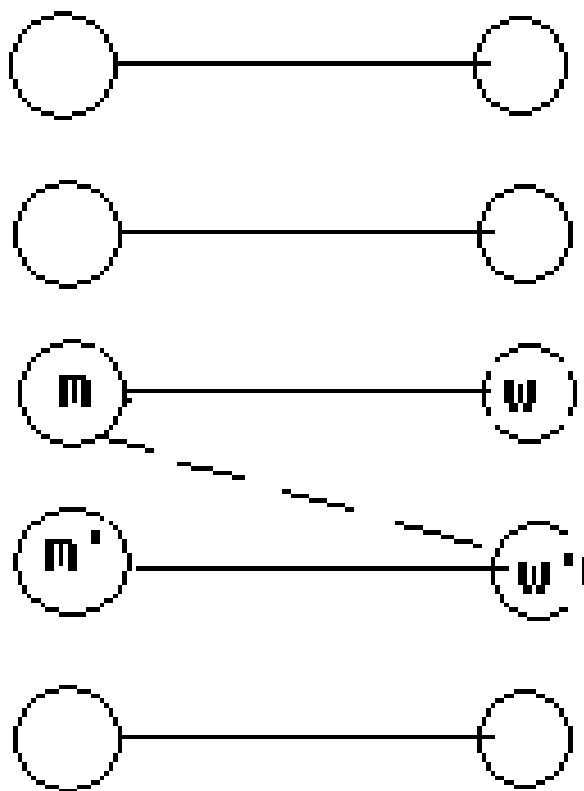


1.1 问题

- 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 , 而且 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



1.1 问题

- 对每组优先表是否**存在**一个稳定匹配？
- 给定一组优先表，如果存在稳定匹配，我们能够有效的**构造**出来吗？
- 如果存在稳定匹配，会有**很多**吗？



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?

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

Men's Preference Profile

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

Women's Preference Profile

例子

- 不是，Bertha-Xavier是更好的配对

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

Men's Preference Profile

	favorite ↓		least favorite ↓
	1st	2nd	3rd
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?

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

Men's Preference Profile

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

Women's Preference Profile

Yes!!!



算法

What is a solution?

下面我们按照 *优先选择* 的思路，尝试给出一个算法



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。



算法

- 循环往复；最后，当没有人处于自由状态，那么所有的约会将被定为最后的结果，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
}
```




算法

- 正确性？
- 有穷性？
- 输出？



分析算法

- G-S算法叙述比较简单
- 具体看一个算法的执行过程

G-S算法是否会输出一个正确结果？

- ✓ 是否G-S算法返回一个完美匹配？
- ✓ 是否G-S算法返回一个稳定匹配？



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。
- 命题5 终止时，**G-S**算法返回的集合**S**是一个完美匹配。



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，那么 w' 一定选择了比 m 更好的对象，无论如何 w' 都不可能与 m 配对。 矛盾。



分析算法

- G-S算法会在有限步内停止吗？
- 这是一个有效的算法吗？需要多少计算步骤？



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。



分析算法

- 如何能够有效的实现**G-S**算法？
- 若存在多个，**G-S**算法找到的是哪一个稳定匹配？



实现算法

- 要点：
- $M=\{1,\dots, n\}; W=\{1,\dots, n\};$
- 用两个数组来记录约会的对象；
- 用一个数组来记录每一个男人在自己优先表的位置（提出邀请的次数）；

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

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

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

Amy prefers man 3 to 6
since $\text{inverse}[3] < \text{inverse}[6]$
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算法生成的是那一个
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？



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 $\text{best}(m)$ 。



推广

- 现在定义 $S^* = \{m, \text{best}(m) : m \in M\}$
- 命题7 G-S算法的每次执行都得到集合 S^*



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 $\text{worst}(w)$ 。



推广

- 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。
- 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。



推广

- 但是不要忘记了一种平衡：
 - ✓ 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



扩展

- 对于一般情形，稳定匹配是否一定存在？
- 是否唯一？
- 如何生成稳定匹配？

扩展

■ 稳定室友匹配问题

- $2n$ 个人，每个人对其他人有个排序，从1 到 $2n-1$.
- 给出分配方案，使得没有不稳定的配对出现.

	<i>1st</i>	<i>2nd</i>	<i>3rd</i>
Adam	B	C	D
Bob	C	A	D
Chris	A	B	D
Doofus	A	B	C

$A-B, C-D \Rightarrow B-C$ unstable
 $A-C, B-D \Rightarrow A-B$ unstable
 $A-D, B-C \Rightarrow A-C$ unstable

- 观察可知，对于稳定室友匹配问题，可能会没有稳定匹配方案！



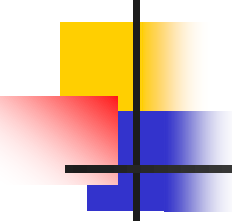
扩展

- Ex: Men \approx hospitals, Women \approx med school residents.
- 不同点 1. 事先有不愿意配对的情况
- 不同点 2. 两边的数目不一样
- 不同点 3. 有些医院可以接收一定数量的学生.
- Def. Matching S **unstable** if there is a hospital h and resident r such that:
 - h and r are acceptable to each other; and
 - either r is unmatched, or r prefers h to her assigned hospital; and
 - either h does not have all its places filled, or h prefers r to at least one of its assigned residents.



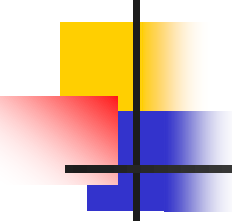
总结

- 如何从一个实际的问题中抽取出具有本质性的问题描述
- 能够设计有效的算法解决问题
- 分析算法的步骤及最后的结果
- 能够以小见大，用理论来解释一些现象



1.2 五个典型问题

- 回顾一下稳定匹配问题的思路
- 问题陈述中的微妙变化对问题的性质，问题的计算效率，有可能会有着巨大的影响。
- 图 G 的描述 (V, E) : V , 结点集合; E , 边的集合



1.2 五个典型问题

- 区间调度
- 带权的区间调度
- 二分匹配
- 独立集
- 竞争的便利店选址问题



1.2.1 区间调度问题

- 问题的描述：
- 你有某种资源（报告厅，超级计算机，电子显微镜），许多人需要在某个时间段使用这个资源。一个**需求**是从时刻**s**开始，到时刻**f**结束。假设每个时刻至多一个人使用这个资源。一个调度员接收了一系列的需求，他需要做出决定：目的是使得**被接收的需求数目最大**。



1.2.1 区间调度问题

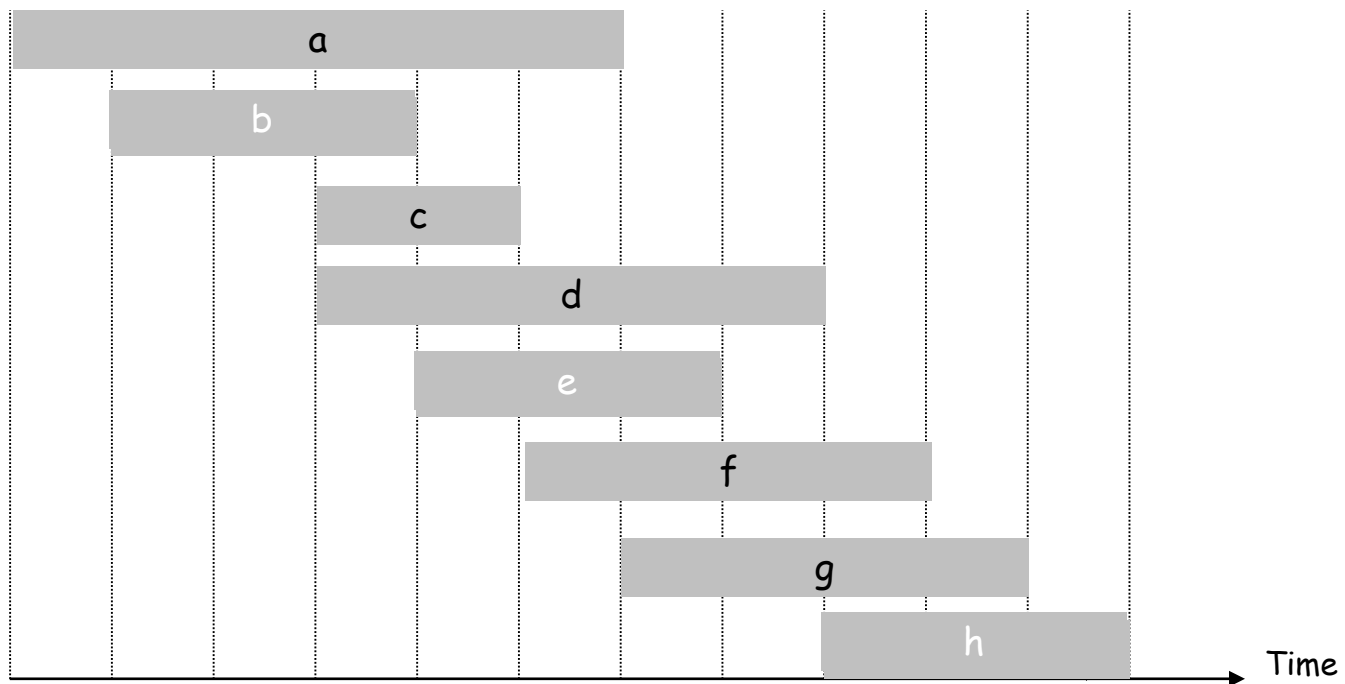
- 形式的说法:

有标记为 $1, 2, \dots, n$ 的 n 个需求, 每个需求 i 从时刻 s_i 开始, 到 f_i 结束($s_i < f_i$)。目标就是选择一个最大的相容需求子集。

(**相容**: 两个需求 i, j 所要求的区间不重叠)



1.2.1 区间调度问题





1.2.1 区间调度问题

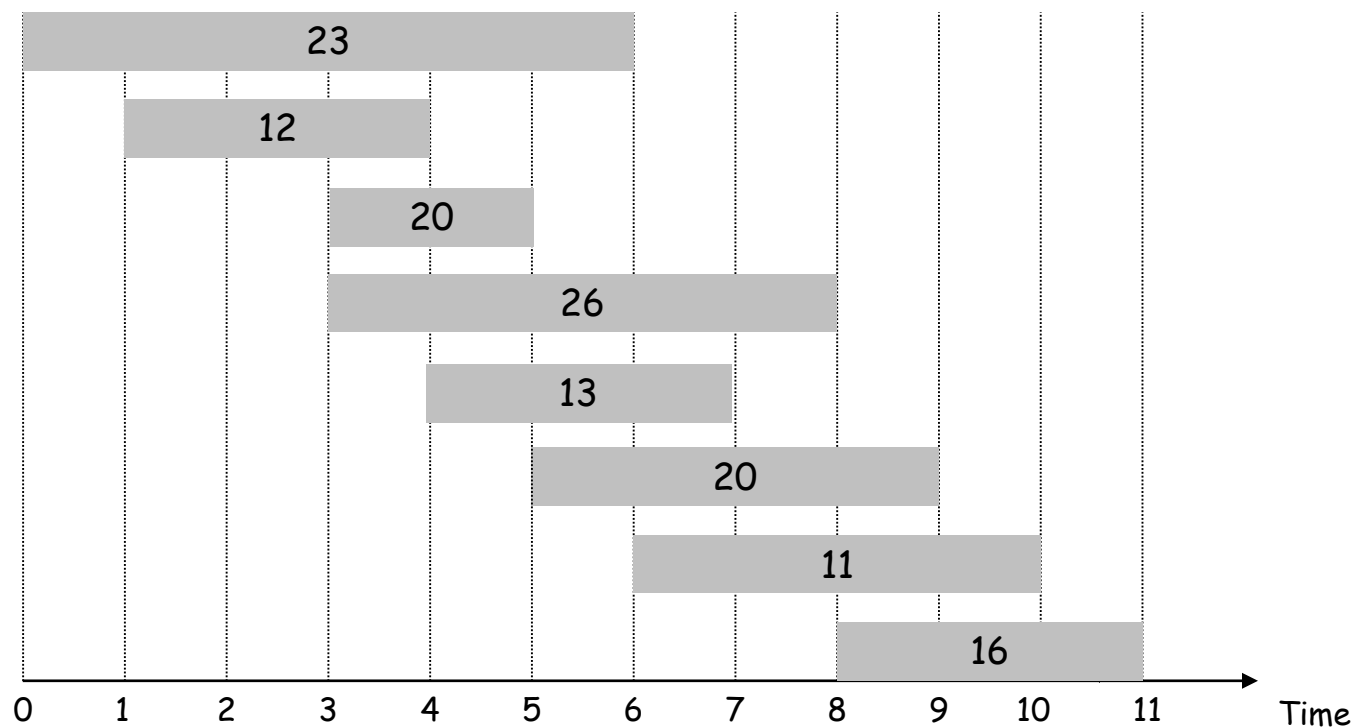
- 可以用贪心算法来求解这个问题。



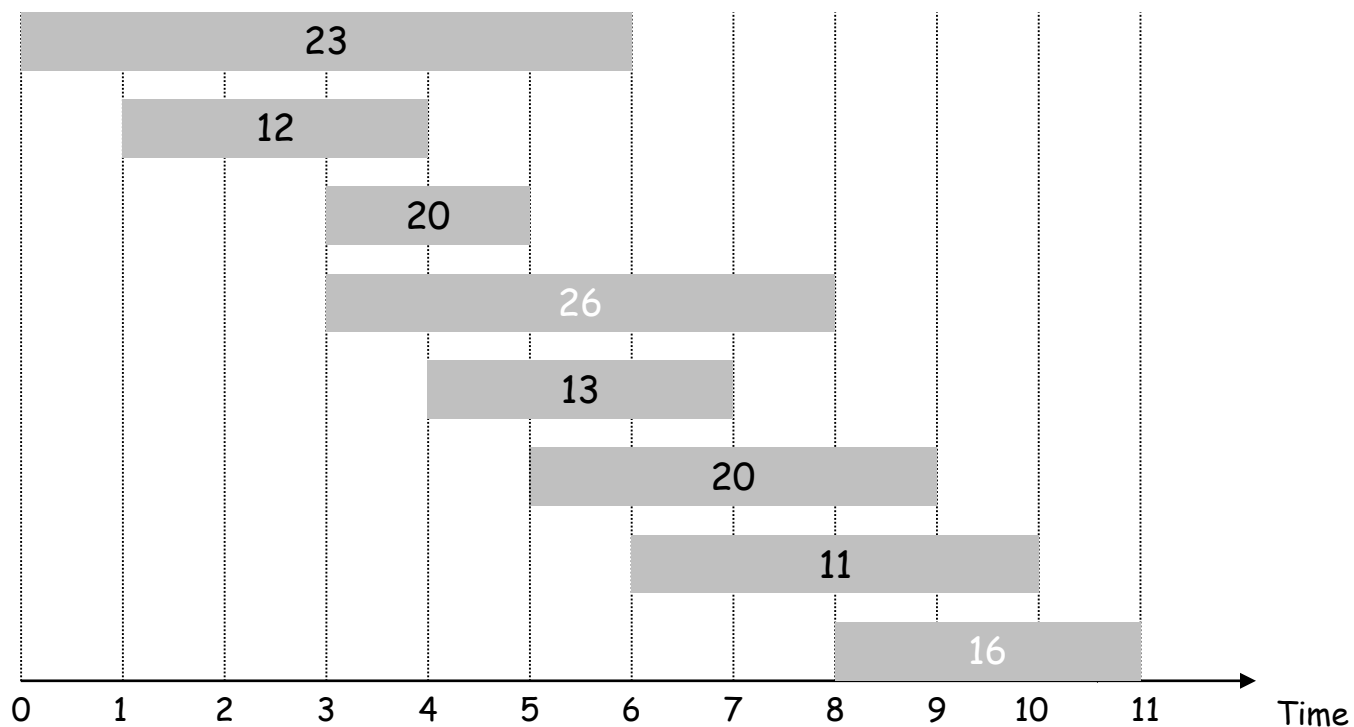
1.2.2 带权的区间调度

- 问题比1.2.1区间调度问题更一般化：
- 假设调度员对第 i 个需求做出安排，可以挣到钱数 $v_i > 0$ ，目标是找一个总价值最大的相容的区间子集。
- 这里每个区间 i 有一个权 v_i 的概念。

1.2.2 带权的区间调度



1.2.2 带权的区间调度





1.2.2 带权的区间调度

- $v_i = 1$, 就是基本的调度问题
- 解决这个问题涉及到动态规划技术:
导致一种非常紧凑的, 表格式的方法得到所有可行解最优值的有效算法。



1.2.3 二分匹配

- 问题的提出：把某些个体分配给其他个体
- 比如：
- 有一些任务，以及一些机器，把每项任务分配给可处理它的机器，使得每台机器恰好分配一项任务。
- 系里面有一些教授，需要开设一些课程，安排课程的分配方案，使得每个教授能教一门课程，课程都能开全。



1.2.3 二分匹配

- **匹配**：每个男人和每个女人至多属于一个有序对
- **完美匹配**：每个男人和每个女人都属于其中某个对的匹配。
- $G=(V,E)$ 是**二部图**，如果他的结点集 V 可以如下划分成集合 X 和 Y , 每条边有一个端点在 X 中，另一个端点在 Y 中。



1.2.3 二分匹配

- 与稳定匹配的不同:
- ✓ 没有优先表的概念
- ✓ 从每个 $x \in X$ 到每个 $y \in Y$ 不一定存在一条边

可能的匹配集有很复杂的结构



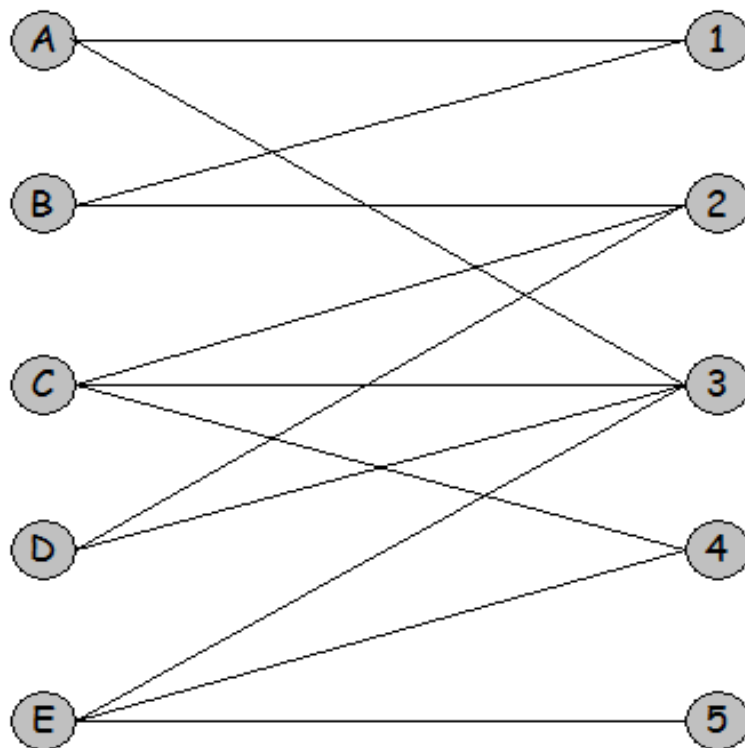
1.2.3 二分匹配

二分匹配问题:

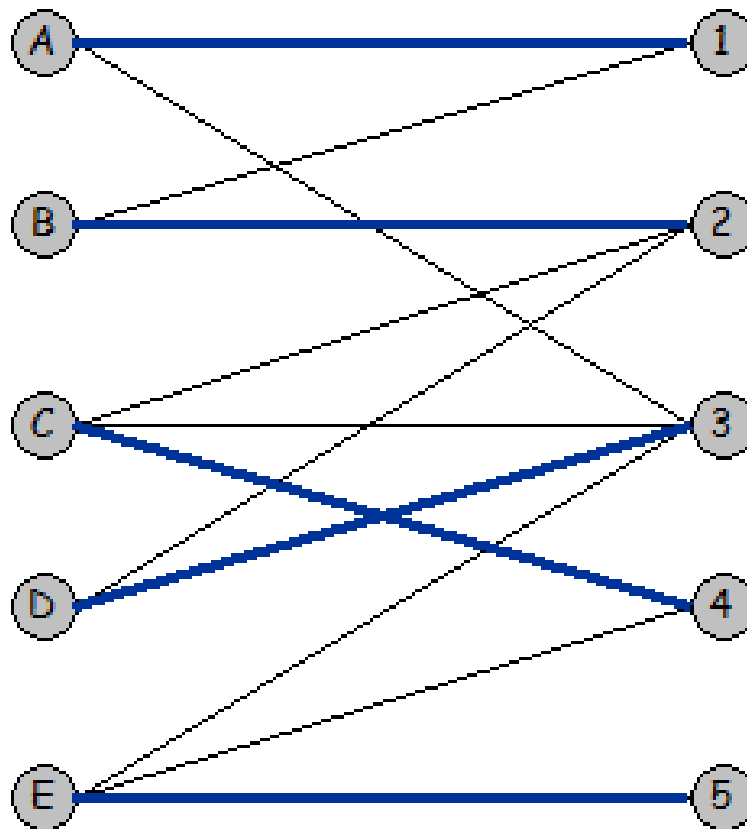
- 给定一个任意的二部图 G ,求一个最大的匹配。
- 如果 $|X|=|Y|=n$,那么存在一个完美匹配当且仅当最大匹配的大小为 n .

1.2.3 二分匹配

例：寻找二分匹配



1.2.3 二分匹配





1.2.3 二分匹配

- 选择性的回溯，归纳的建立越来越大的匹配。这种处理叫做**增广**。网络流问题中，增广构成了其中的核心要素。



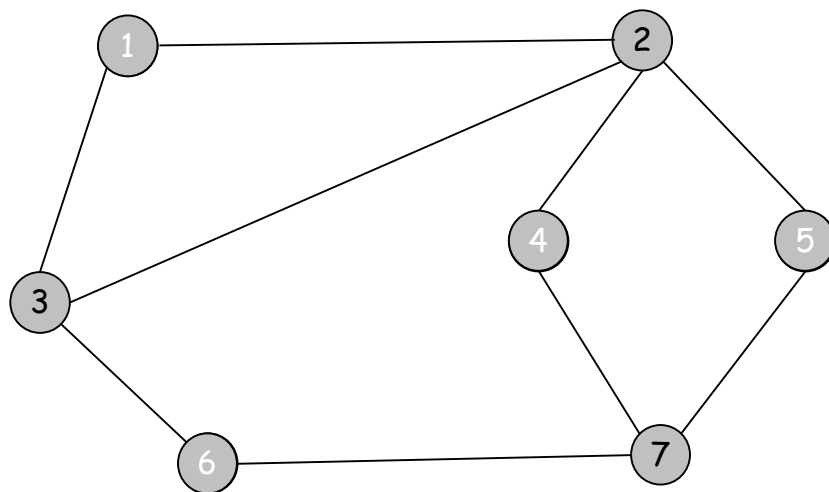
1.2.4 独立集

- 给定图 $G=(V,E)$, 我们说结点集合 $S \subseteq V$ 是**独立**的, 如果在 S 中没有两个节点与同一条边相交。

独立集问题: 给定 G , 找一个最大的独立集。

1.2.4 独立集

- 例：寻找最大的独立集



{1, 4, 5, 6}



1.2.4 独立集

- 独立集问题称为**NP**完全这一大类问题中的一个
- 检查某个集合是大的独立集(**验证**)；真正找到一个大的独立集(**求解**)；难度上看起来存在很大的差别。



1.2.5 竞争的便利店选址问题

- 问题的提出
- 竞争策略：

两家咖啡公司：JavaPlanet, Queequeg 竞争某个地区的市场份额，交替开咖啡馆。假设他们必须遵从分区规章(**1.相邻地区禁止开咖啡店；2.一个地区只开一家**)，要求两家咖啡馆不允许位置太接近，每个公司都想使布局尽可能方便，谁是赢家？



1.2.5 竞争的便利店选址问题

- 问题中的地区被划分为 n 个小区，标记为 $1, 2, \dots, n$. 每个小区 i 有一个值 b_i , 是在那里开一家咖啡馆的公司所得的收入.
- 通过图 $G=(V,E)$ 建模, V 对应小区的集合, 分区要求就是, 所开的全部咖啡馆的集合构成 G 的一个独立集.

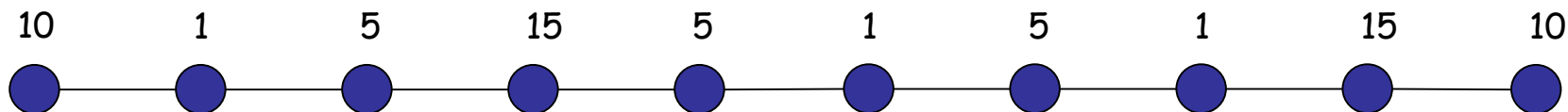


1.2.5 竞争的便利店选址问题

- **竞争的便利店选址问题**：对策由两个对手 P_1, P_2 构成. 它们交替从 G 中选择结点, P_1 走第一步. 假设对手 P_2 有一个界为 B 的目标, 我们关心, 对 P_2 是否存在一种策略不管 P_1 如何走, P_2 都能选出一个总值至少是 B 的结点的集合?

1.2.5 竞争的便利店选址问题

■ 例子：



■ $B=20$, P_2 有赢的策略； $B=25$, P_2 就不再有赢的策略。



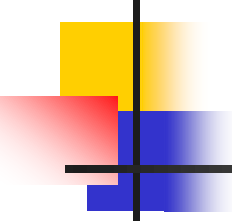
1.2.5 竞争的便利店选址问题

- 从例子可以看出，确信 P_2 有一个赢的策略也是困难的。简短证明不明显，为了验证需要对可能的交替过程一个个状态进行冗长分析。
- 这里验证一个解也是很困难的



1.2.5竞争的便利店选址问题

- PSPACE完全问题类
- PSPACE完全问题严格难于NP完全问题
- PSPACE完全的概念涉及到对策策略，规划，人工智能领域的基本课题



小结

- Interval scheduling: $n \log n$ greedy algorithm.
- Weighted interval scheduling: $n \log n$ dynamic programming algorithm.
- Bipartite matching: n^k max-flow based algorithm.
- Independent set: NP-complete.
- Competitive facility location: PSPACE-complete.