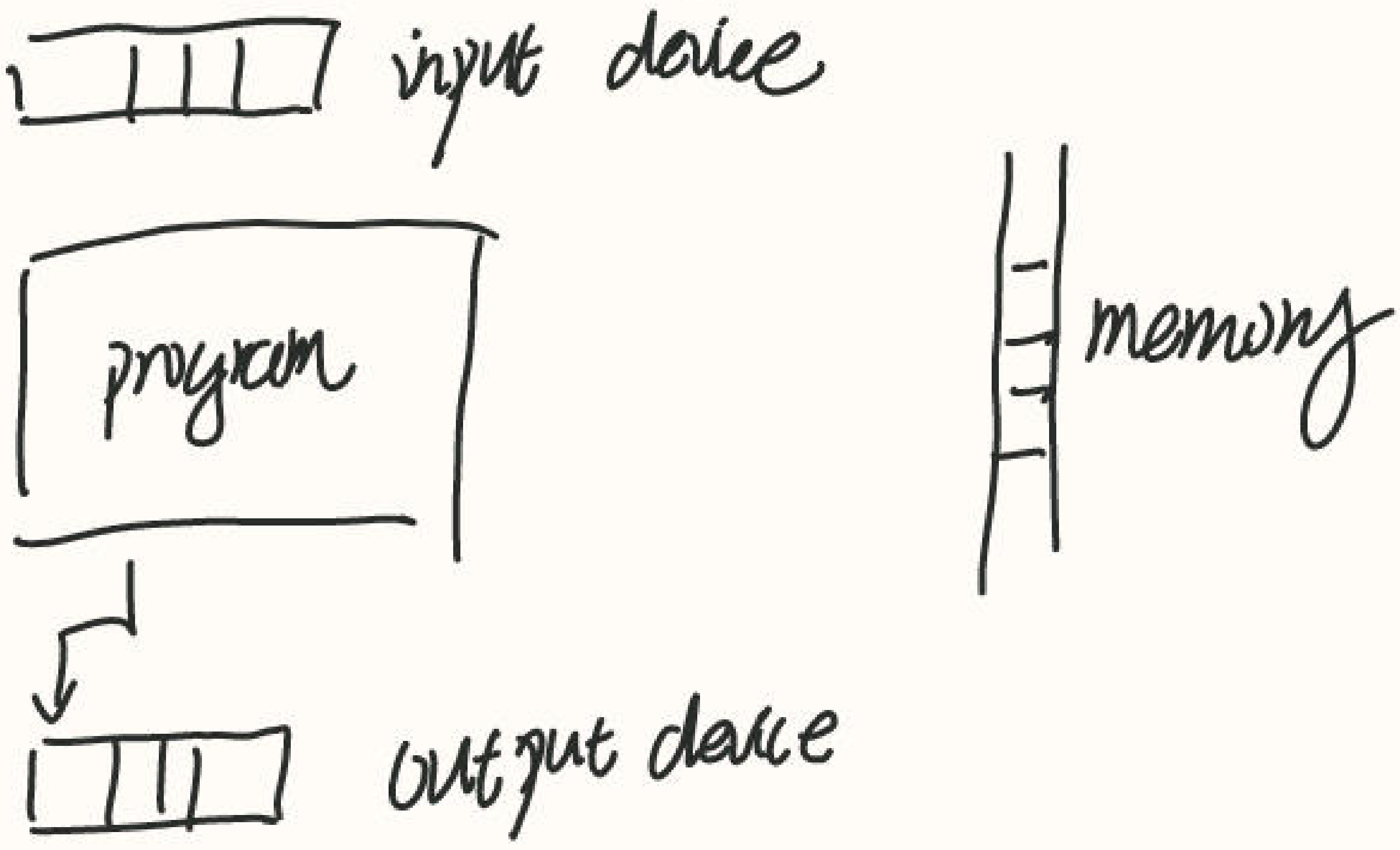
RAM模型:

抽象模型 → 计算模型



排序

```
MERGE-SORT(A, p, r)
  if p < r
     q = ⌊(p+r)/2⌋
     MERGE-SORT(A, p, q)
     MERGE-SORT(A, q+1, r)
     MERGE(A, p, q, r)
```

分而治之 $(n\lg n)$

动态规划

DFS

for each vertex $u \in V[G]$

do color[u] ← white

π[u] ← NULL

for each vertex $u \in V[G]$

do if color[u] = White

then DFS-VISIT(u)

color[u] = GRAY

for each $v \in$ Adj[u]

do if color[v] = white

then π[v] ← u.

DFS-VISIT(v)

color[u] = Black.

日期:　　　/

一

（1）解释算法时间复杂度的三个符号——Θ、Ω、O

（2）T（n）=T（3n/4）+nlogn，计算T（n）的时间复杂度 最大上界

（3）证明顶点覆盖问题是NP完全问题

NPC 证明

二

（1）寻找强连通分支，证明其正确性，时间复杂度

（2）在已有最小生成树的基础上，将随机一条边的权重变大，求新的最小生成树的更新算法。写出伪代码，证明其正确性。

三

（1）寻找最短路径的三个相关证明

1.最短路径的子路径也是最短路径

2.对于任意两个点间的所有最短路径，总有一条为有限长度

3.证明三角不等式 $\delta(s,v)<=\delta(s,u)+w(u,v)$

（大概是这三个，诶嘿）

（2）Floyd-Warshall算法（多源最短路径问题）说明其相关思想，补全（更新）邻接矩阵，并写出最短路径

四

（1）从左到右有n堆棋子，每堆棋子若干个，只能合并相邻堆的棋子，合并的花费为被合并棋子堆的棋子数。

现要把这n堆棋子合并为一堆，给出一个花费最小的算法。写出伪代码，证明其正确性。

（2）有一个区间集N，中间有若干段闭区间。有一个点集P（数集合），如果P含有的任意数字包含在一段闭区间内，则称该区间被击中。

最小击中集：如果对于N中的任意一个区间，P中都存在一个点可以击中该集合，且要求P中的点尽可能少。

给出一个算法，求P中最小有多少个点，写出伪代码，并证明其正确性。

（这里其实是还有一张图来帮忙理解的，可惜我没找到一样的问题，下面补个类似的）

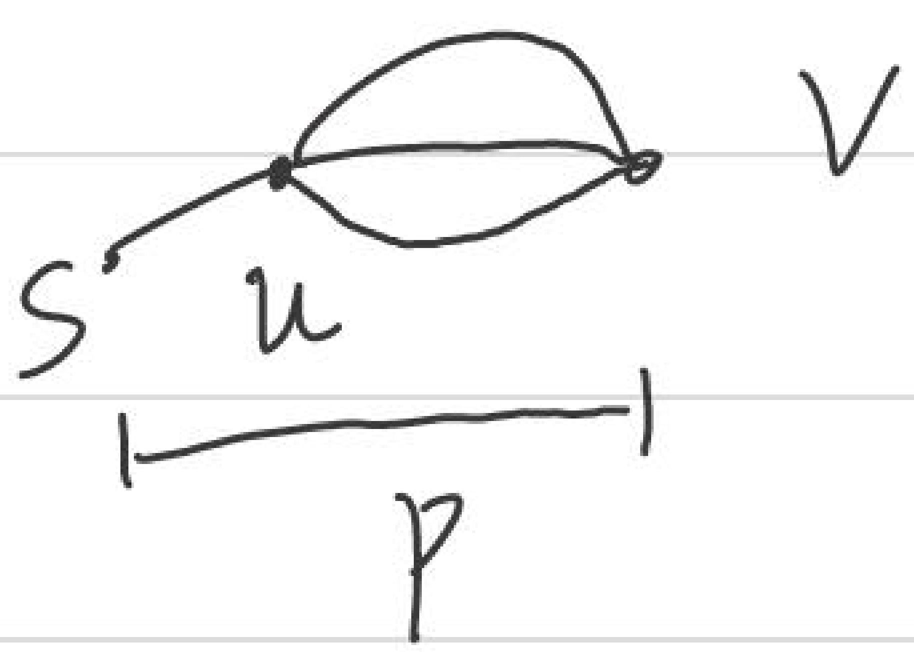
类似问题：假设存在下面需要付费的广播台，以及广播台信号可以覆盖的地区。 如何选择最少的广播台，让所有的地区都可以接收到信号

日期:

三角不等式: $\delta(s,v) \le \delta(s,u) + w(u,v)$

证明 ①假定p是从源结点s到结点v的一条最短路径，则p的权重不会比任何从s到v的其它路径的权重大，因此路径p的权重也不会比这样的一条路径的权重更大：从源结点s到结点u的一条最短路径再加上边(u,v)而到达结点v的这条路径

②如果s到v没有最短路径，则不可能存在s到v的路径。



Bellman-Ford算法

Initialize-single-source(G,S)

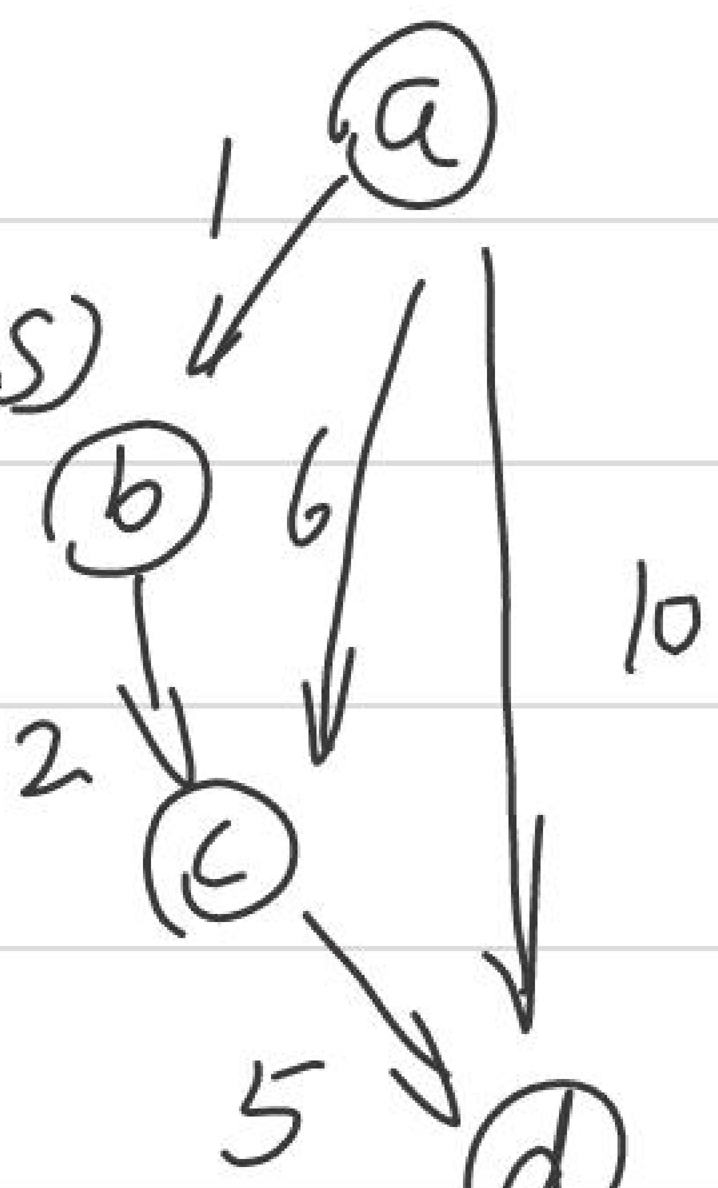
for i=1 to |G.V|-1

for each edge (u,v)∈E

Relax(u,v,w)

for each edge (u,v)∈E if v.d>u.d+w(u,v) return F

else return T



日期: /

渐近精确界 Θ 西塔 $=$ $f(n)=n^2+n$ $\Theta(n^2)$

渐近上界 O 大欧上界 $\le$ $0 \le f(n) \le cg(n)$

o 小欧 $<$

渐近下界 Ω 下界 $\ge$

ω 小欧 $>$

O $\le$

Θ

Ω $\ge$

$\left(\frac{3}{4}\right)^i$ $\left(\frac{3}{4}\right)^{\log_4 n}$ $n^{\log_4 \frac{3}{4}}$ $\boxed{n^{\log_4 3}}$

$T_n = n + 3T[\lfloor n/4 \rfloor]$

$T(n) = n + \frac{3}{4}n + \frac{9}{16}n + \frac{27}{64}n + \cdots + 3^i * \left(\frac{1}{4^i}n\right)$

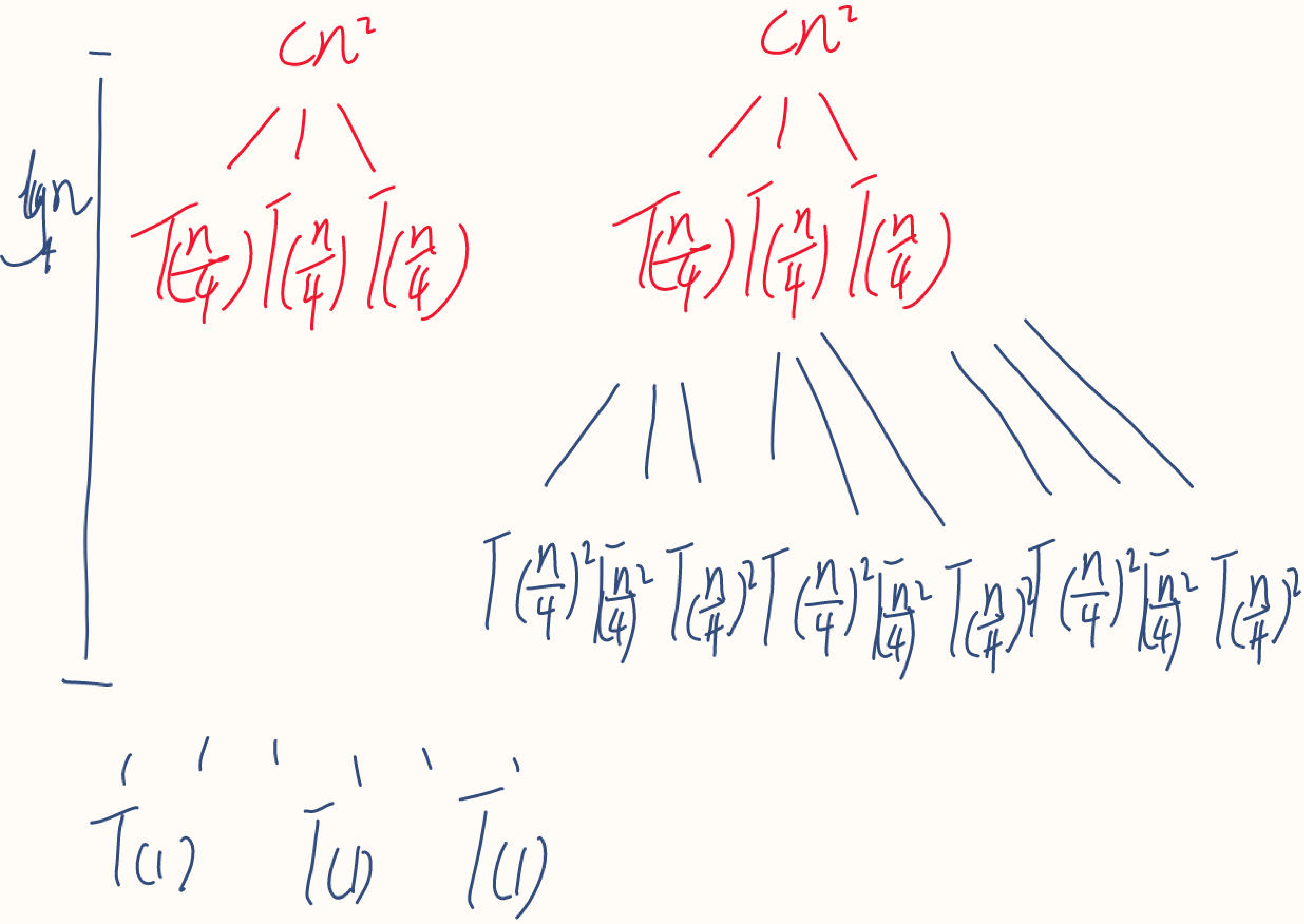
$\frac{1}{4^i}n = 1$ $n = 4^i$ $i = \log_4 n$. $3^i * \frac{1}{4^i}n$

$T_n \le n\left(1 + \frac{3}{4} + \frac{9}{16} + \cdots + \left(\frac{3}{4}\right)^n\right) + \Theta(n^{\log_4 3}) = 3^{\log_4 n}\, n$

$= n^{\log_4 3}\, n$ 换底公式

$\left(\frac{1}{1-\frac{3}{4}}\right)n = 4n.$ $= \Theta(n^{\log_4 3})$

$T_n \le 4n + \Theta(n^{\log_4 3}) \le O(n) + 4n \le O(n)$



$n^{\log_4 3}$

$$3^{\log_4 n} = n^{\log_4 3}$$

$$\max\{(i-1), (w-w_i)+v_i, [i-1, w)]$$

$$\max\{c[i-1, j], c[i-1, j-w_i]+v_i\}$$

① 刻画一个最优解的结构特征

② 递归定义最优解的值

③ 计算最优解的值 采用自底向上

④ lead to 最优解

```
BTUCR(p.n)

create a new array[0,...n]
  r[0]=0
for i=1 to n
    q = p[i]
    for j=i-1 to 1
        q = max(q, p[j]+r[i-j])
    r[i]=q
return r[n]
```

```
CR(p.n)
  if n==0
      return 0
  q=-∞
  for i=1 to n
      q = max(q, CR(p.n-i))
  return q.
```

21

证明f'是一个流（容量约束 流守恒约束）。（课本证明和上课讲的证明方法不同，两者都可，但个人倾向于课本证明，理解以后证明思路很清晰）

强连通分支的证明

设计最小生成树算法（通过安全边），算法正确性证明，时间复杂度分析

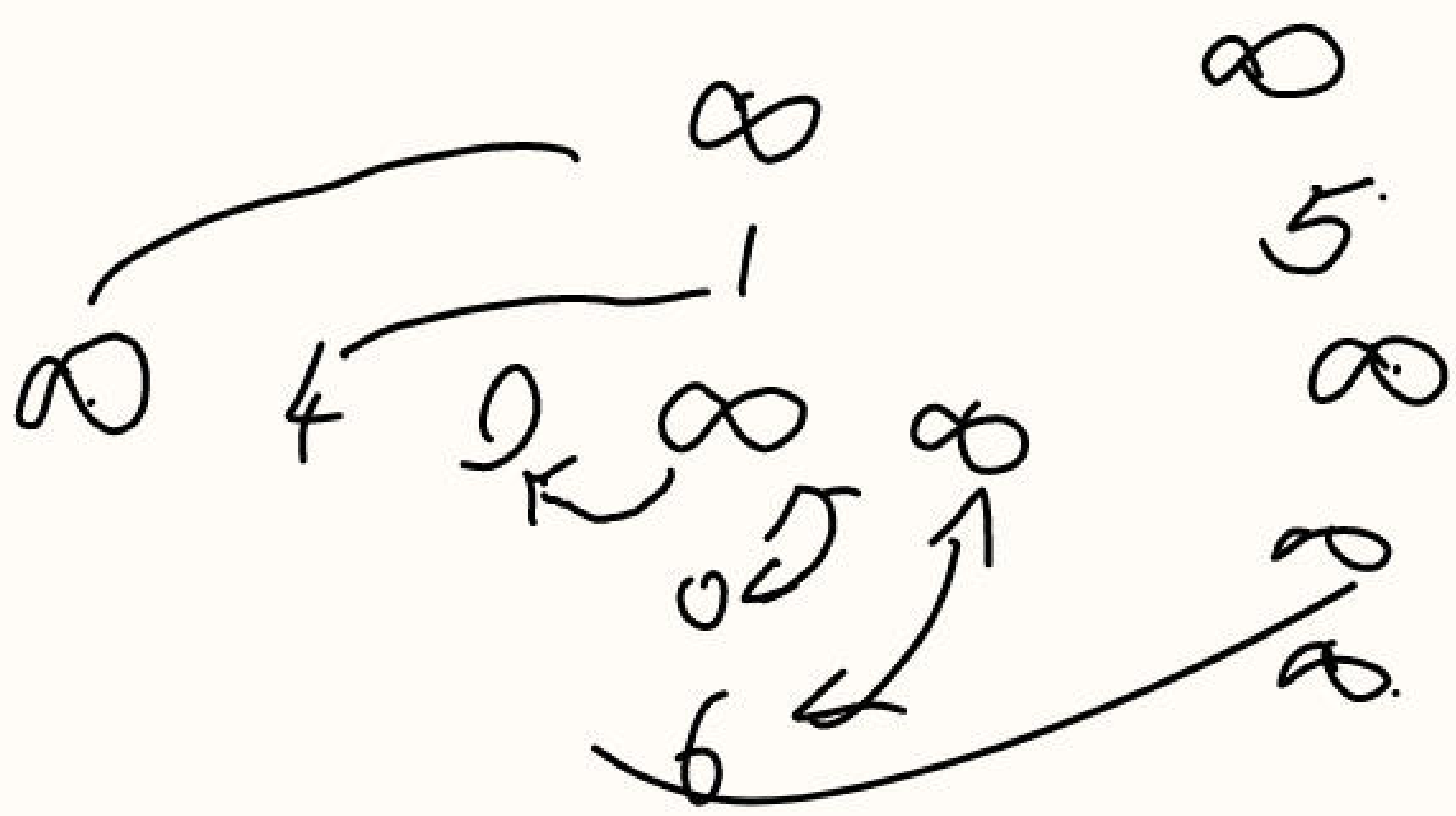
DAG中最长路径的算法设计，bellman方程，时间复杂度分析

迭代次数与所求点到源点s边数相等证明

类似于最短路径的算法设计，给予每条边一个宽度，计算出每条路径的最小宽度，设计算法并证明正确性，时间复杂度分析。

最大流有关的问题，证明路径条数和最大流容量的关系

1.算法的下界为(),非紧下界为(),上界为(),非紧上界为()$
A.O B.o C.ω \omegaω D.Ω \OmegaΩ
2.渐进复杂度排序问题(选择),包含对数函数、幂函数、指数函数等【简单】
3.遗忘
4.Strassen矩阵乘法使用的具体思想为:
A.分而治之 B.贪心算法 C.动态规划 D.回溯随机
5.是一道关于计算复杂度的问题,具体内容实在背不下来,内容包括NPC、NP-hard、3SAT问题等定义,主要是基础知识的考查,



日期: /

21

NP问题 非确定性多项式问题.

验证 猜测在多项式时间内完成

二、 指问题只能通过验证求解

2.1 计算 T (n) = 2 T (n / 2) + n 2 l o g n , T (1) = 1 T(n)=2T(n/2)+n^2logn,T(1)=1T(n)=2T(n/2)+n
2
 logn,T(1)=1

则求解 可以直接求解

① 在多项式时间内求解的问题称为P问题

2.3 写出P、NP、NPC的定义

② 指问题只能通过验证给出的猜测是否正确来求解

2.4 证明 P ⊆ N P P\subseteq NPP⊆NP

NPC问题

① 它是一个NP问题

② 所有的NP问题都可以用多项式时间约化它.

三、
3.1【分治】
给定数组T[1...n],如果对于1 ≤ i ≤ n 1\leq i\leq n1≤i≤n,存在T(i)=i,则称i为不动点,请你设计一个算法,如果存在不动点输出不动点的一个下标,如果不存在则输出0
3.2 【贪心】
在一个银行内,如果有n个人需要处理业务,他们需要的处理时间是t i t_it
i

 ,只有一个银行柜员,请你设计一个算法,使得处理业务数量得到最大
3.3 【动态规划】
本质为0-1背包问题，就不再写题目描述了

2.2 假设$W_1(n)=7W(n/2)+n^2, W_2(n)=aW(n/4)+n^2$,且$W_2(n)$的阶小于$W_1(n)$的阶，求a的最大整

一（30分）
某个点的最短边是否属于某课最小生成树，如果是，请证明。

要求你给出一个求强连通分量的算法，并且证明正确性

网络流的一个割，其中一条边去掉之后，最大流是多少

二（20分）
给出3SAT问题和Independent Set问题的判定形式，并证明：Independent Set问题是NP-hard问题

网络流算法中，找到增广路径后，证明新的流是合法的（流量守恒and容量限制）

三（15分）
求单源最长路径，要求使用动态规划算法（bellman-ford算法即可）

伪代码，时间复杂度，思想，动态规划方程

四（15分）
描述Floyd算法思想，给矩阵D0D2 求D1D3（理解Floyd算法就会做）

五（20分）
动态规划算法设计：
n个题目，每个题目有mi的做题时间和vi的分数
问得到V分数的前提下，最少需要多少时间做题（假设题目全对）
写动规方程，算法思路，伪代码，设计一个时间复杂度为mV的算法

一、
（1）一个英文题是关于安全边定理的改编
（2）强连通分量的伪代码，时间复杂度，正确性证明
（3）最大流<=最小割的变式
二、
（1）将3SAT问题规约为Indenpent problem（独立集问题）
（2）证明增广后得到的流仍然是合法的流（容量限制和流量守恒）
三、动态规划最长路径改编
四、所有顶点对之间的最短路径-基本思想，填表（Ford-wall算法）
五、关于背包问题的改编（能达到某个价值的最小重量-大体就是这个意思）

18

一.

（1）强连通分量正确性证明。

（2）DAG中最长路径的算法设计，写出bellman方程和伪代码，并分析时间复杂度。

二.

（1）白色路径定理的证明。

（2）假设最短路径含有K条边，证明迭代K次可以产生最短路径。

三.

T是图G中的一棵最小生成树，现将G中一条边的权重改为w’，设计算法实现对最小生成树T的更新。简述思想，写出伪代码，分析正确性。

四.

给了一个图，计算出最大流和最小割，要给出详细过程。

五.

每条边赋予一个宽度，一条路径的宽度为这条路径上边的最小宽度。借鉴Dijkstra算法思想，计算出从源点S到其他每个顶点的路径的最大宽度。

简述基本思想，写出伪代码，证明正确性，分析最坏时间复杂度。

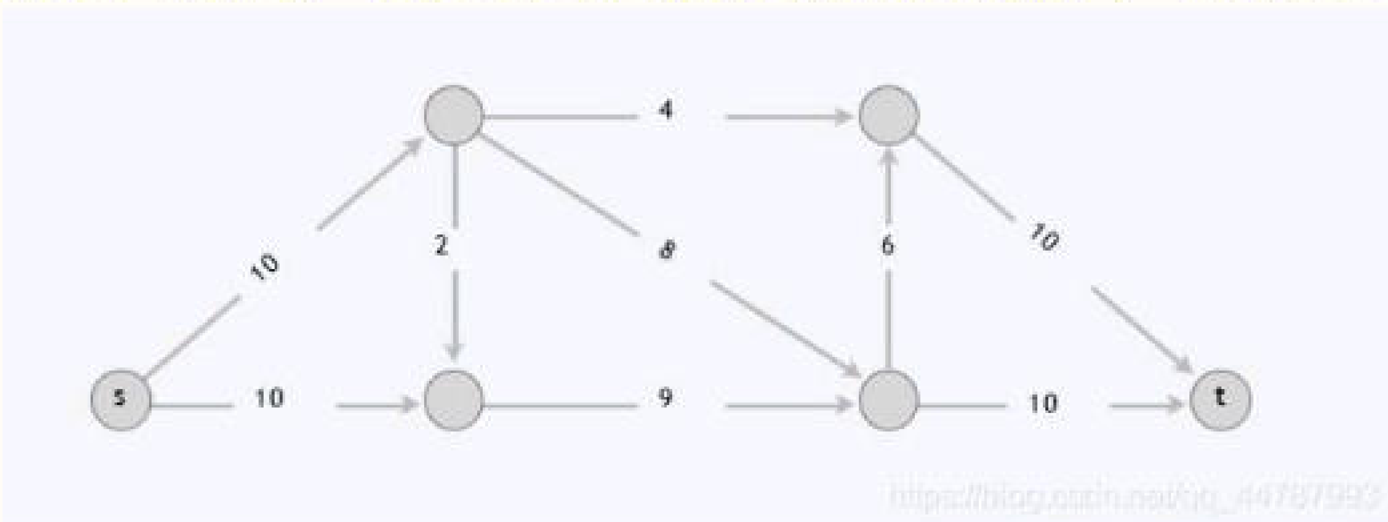
说明算法的时间复杂度为O(n2)与问题的时间复杂度为O(n2)含义与区别
简述贪婪算法的基本思想

对于一个有向无圈图DAG，其中顶点s入度为0，t出度为0，设计算法求s到t的最长路径的长度，简述算法的基本思想，写出伪代码并分析其时间复杂度

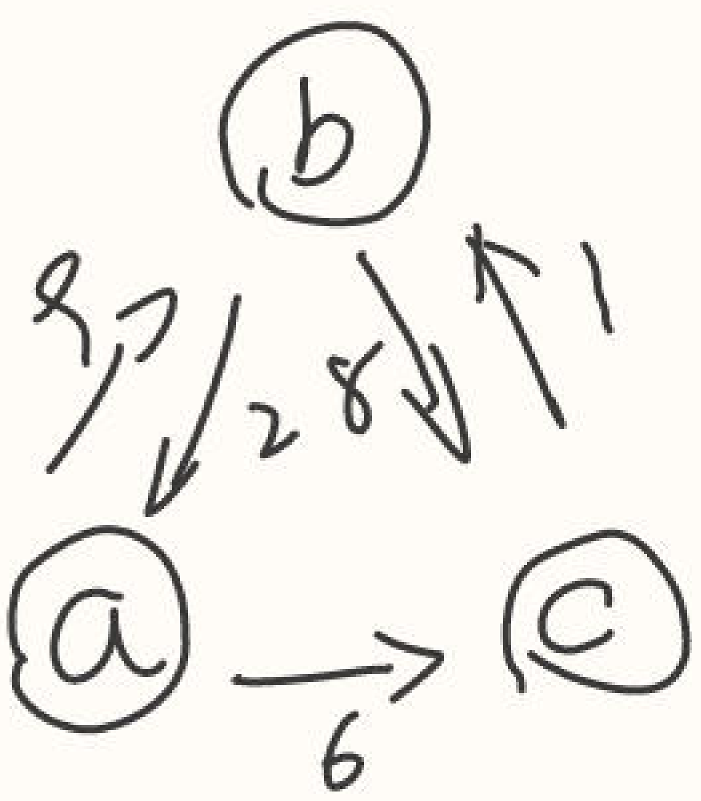
证明
安全边定理
最大流最小割定理
求单源点最短路径中，设源点s到顶点v的最短路径包含的边数为k,证明在Bellmanford算法中，经过第k次循环后，得到s到顶点v的最短距离

写出Ford-Fulkerson算法的伪代码，假设流网络中容量均为整数，且最大流量为C，试分析算法的时间复杂度，求出如下流网络中最大流和最小割（此流网络是从ppt上摘抄地，和试题上的流网络做法是一样的）

写出求有向图的强连通分支的伪代码，并给出证明



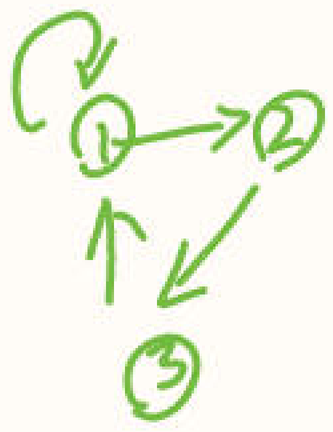
简述Floyd-Warshall算法的基本思想，对如下有向图，已知D(0)矩阵如下图所示，求出其D(1),D(2),D(3)矩阵



$$\begin{bmatrix} 0 & 2 & 8 \\ 2 & 0 & 6 \\ \infty & 1 & 0 \end{bmatrix}$$

算法 → 邻接链表法 / 邻接矩阵 (↓① →②)

$\Theta(V+E)$



$$
\begin{array}{c|ccc}
 & 1 & 2 & 3 \\
\hline
1 & 1 & 1 & 0 \\
2 & 0 & 0 & 1 \\
3 & 1 & 0 & 0
\end{array}
$$

u.d 距离

u.π 前驱的值 A/B/C

← 根结点,

```
BFS (G, s)
for each vertex u ∈ G.V-{s}
        u.color = white;
        u.d = ∞;
        u.π = nil;
    s.color = gray [redacted]
      s.d = 0
      s.π = nil
      Q ≠ ∅
      ENQUEUE(Q, s)
      while Q ≠ ∅
           u = DEQUEUE(Q)
           for each v ∈ G.Adj[u]
               if v.color == white
                   v.color = GRAY
                   v.d = u.d + 1
                   v.π = u
                   ENQUEUE(Q, v)
           u.color = Black
```

$$\max\{\ c[i-1, j-1] + v \quad c[i-1, j]\ \}$$

$$\begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 3 & 8 & 2, & -4 \\ 3 & 0 & -4, & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

Floyd-Warshall 算法