蛮力法

就像宝剑不是撬棍一样，科学也很少使用蛮力。

——Edward Lytton

把事情做“好‘常常是浪费时间。

——Robert Byrne

(撞球大师，台球选手和作家)

思考题

- 某地刑侦大队对涉及六个嫌疑人的一桩疑案进行分析：（1）A、B至少有一人作案；（2）A、E、F三人中至少有两人参与作案；（3）A、D不可能是同案犯；（4）B、C或同时作案，或与本案无关；（5）C、D中有且仅有一人作案；（6）如果D没有参与作案，则E也不可能参与作案。试用蛮力法设计算法将作案人找出来。

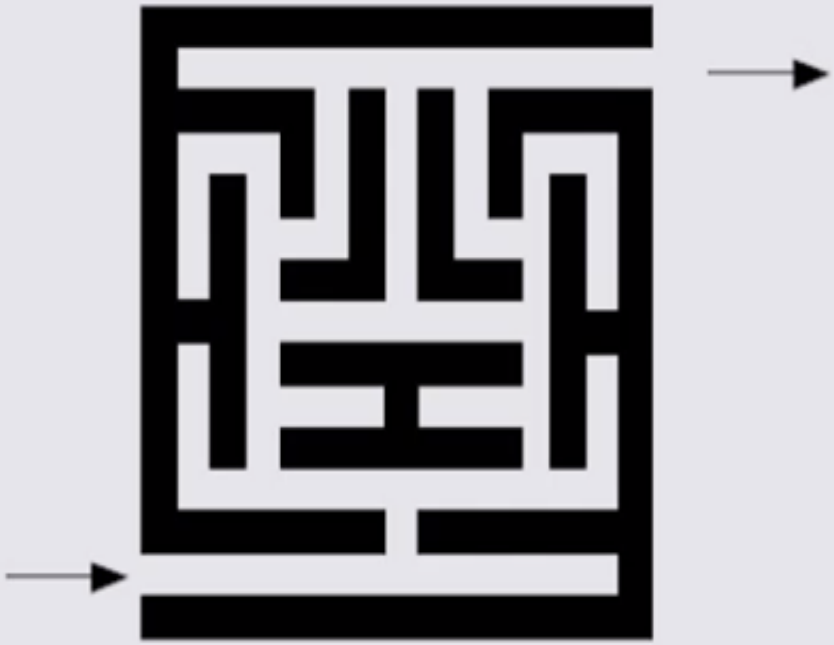
找字游戏

ANTI
ARSON
ASCEND
BAMBOO
BOYCOTTING
COCOA
COUNTERSIGN
DIGITALIZING
DUELLIST
EDGY
EVERLASTING
GREETINGS
GUINEA
GULAG
HUSTLED
JAUNT
MEDIOCRITY
MESTIZO
MOGUL
MOSAIC
NEAT

T	D	D	G	G	Y	C	A	M	S	R	A	C	K	E	T
A	I	E	A	N	T	I	G	R	E	E	T	I	N	G	S
E	G	L	L	I	I	P	E	R	S	O	N	A	E	W	I
N	I	T	U	L	R	T	S	Y	B	O	O	S	V	H	L
I	T	S	G	E	C	T	T	H	D	A	N	O	E	I	L
U	A	U	O	V	O	N	U	O	S	B	C	M	R	T	E
G	L	H	M	A	I	U	Y	N	C	K	E	S	L	E	U
N	I	R	T	R	D	A	S	P	S	Y	C	H	A	W	D
O	Z	O	N	E	E	J	S	P	E	W	O	I	S	A	N
L	I	C	B	A	M	B	O	O	U	C	H	B	T	S	E
B	N	Y	O	Z	I	T	S	E	M	S	K	I	I	H	C
O	G	O	N	C	S	A	L	V	A	T	I	O	N	I	S
A	R	G	Q	C	O	U	N	T	E	R	S	I	G	N	A
G	N	I	Z	I	N	A	M	O	W	A	G	E	D	G	Y

任意选取一个单词，然后在字母堆中间按照某个顺序去对比，直到单词被找到。

迷宫



每到迷宫的一个转角处，就选择一个方向，直到无法穿越迷宫，然后我们回过头，再去选择另外一个转角，直到发现了出口为止。

水杯

如何通过这几个杯子互相倒水，
最后能够分出4升的水？

水杯

把水杯间互相倒水的可能性列举出来，直到发现4升水为止。

Step#	3-pint jug	5-pint jug	8-pint jug
	0	0	8
1	0	5	3
2	3	2	3
3	0	2	6
4	2	0	6
5	2	5	1
6	3	4	1

北京郵電大學

蛮力法

蛮力法在解决问题时采取的一种“懒惰”的策略。这种策略不经过（或者说是经过很少的）思考，把问题的所有情况或所有过程交给计算机去一一尝试，也即是检查搜索空间中的每一个解直至找到问题的解。

【例】百钱百鸡问题。中国古代数学家张丘建在他的《算经》中提出了著名的“百钱百鸡问题”：鸡翁一，值钱五；鸡母一，值钱三；鸡雏三，值钱一；百钱买百鸡，翁、母、雏各几何?

【例1】百钱百鸡问题。中国古代数学家张丘建在他的《算经》中提出了著名的“百钱百鸡问题”：鸡翁一，值钱五；鸡母一，值钱三；鸡雏三，值钱一；百钱买百鸡，翁、母、雏各几何?

算法设计1：

设x，y，z分别为公鸡、母鸡、小鸡的数量。

尝试范围：由题意给定共100钱要买百鸡，若全买公鸡最多买100/5=20只，显然x的取值范围1~20之间；同理，y的取值范围在1~33之间，z的取值范围在1~100之间。

约束条件： x+y+z=100 且 5*x+3*y+z/3=100

算法1如下：

```
main( )
{ int x,y,z;
for(x=1;x<=20;x=x+1)
   for(y=1;y<=34;y=y+1)
     for(z=1;z<=100;z=z+1)
        if(100=x+y+z and 100=5*x+3*y+z/3)
            {print(the cock number is″,x);
              print(the hen number is″, y);
      print(the chick number is ″z);}
}
```

算法分析：以上算法需要枚举尝试20*34*100=68000次。算法的效率显然太低

算法设计2：

在公鸡（x）、母鸡（y）的数量确定后，小鸡 的数量 z就固定为100-x-y，无需再进行枚举了

此时约束条件只有一个： 5*x+3*y+z/3=100

算法2如下：

```
main(   )
{  int x,y,z;
   for(x=1;x<=20;x=x+1)
      for(y=1;y<=33;y=y+1)
       { z=100-x-y;
       if(z mod 3=0 and 5*x+3*y+z/3=100)
          {print(the cock number is″,x);
           print(the hen number is″, y);
print(the chick number is ″z);}
      }
}
```

算法分析：以上算法只需要枚举尝试20*33=660次。实现时约束条件又限定z能被3整除，进一步提高了算法的效率。

蛮力法的设计思想

蛮力法的设计思想：是一种简单直接地解决问题的方法，常常直接基于问题的描述。

例：计算a^n

$$a^n=\underbrace{a\times a\times\ldots\times a}_{n次}$$

- 蛮力法所赖的基本技术——扫描技术
- 关键——依次处理所有元素
- 基本的扫描技术——遍历

（1）集合的遍历

（2）线性表的遍历

（3）树的遍历

（4）图的遍历

虽然巧妙和高效的算法很少来自于蛮力法，实际问题的搜索空间可能很大，可能得花上几个世纪的时间才能检查完它的每一个解。基于以下原因，蛮力法也是一种重要的算法设计技术：

（1）理论上，蛮力法可以解决可计算领域的各种问题。

（2）蛮力法经常用来解决一些较小规模的问题。

（3）对于一些重要的问题蛮力法可以产生一些合理的算法，他们具备一些实用价值，而且不受问题规模的限制。

（4）蛮力法可以作为某类问题时间性能的底限，来衡量同样问题的更高效算法。

- 蛮力法简单，唯一要做的就是系统地产生问题的每一个可能解；

- 当运用蛮力法时，最基本的问题是：如何得到问题每一个可能解的序列？

北京郵電大學

组合问题中的蛮力法

1. 0/1背包问题

2. 任务分配问题

北京郵電大學

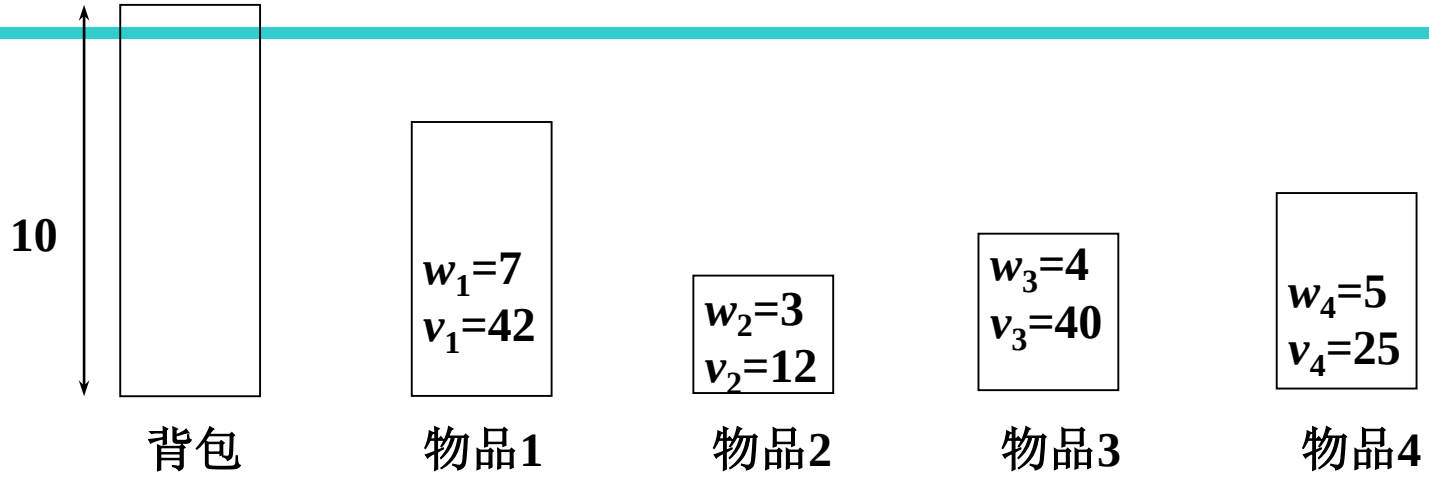
1. 0/1背包问题

- 物品 1：重量 5，价值 9
- 物品 2：重量 3，价值 6
- 物品 3：重量 4，价值 8
- 物品 4：重量 2，价值 3

- 重量单位为千克，价值单位为万元。
- 唯一可以用来装这些物品的背包最多只能装下10千克，那么该怎么选择呢？

0/1背包问题是给定n个重量为$\{w_1, w_2, \cdots, w_n\}$、价值为$\{v_1, v_2, \cdots, v_n\}$的物品和一个容量为$C$的背包，求这些物品中的一个最有价值的子集，并且要能够装到背包中。

用蛮力法解决0/1背包问题，需要考虑给定n个物品集合的所有子集，找出所有可能的子集（总重量不超过背包容量的子集），计算每个子集的总价值，然后在他们中找到价值最大的子集。



序号	子集	总重量	总价值	序号	子集	总重量	总价值
1	φ	0	0	9	{2,3}	7	52
2	{1}	7	42	10	{2,4}	8	37
3	{2}	3	12	11	{3,4}	9	65
4	{3}	4	40	12	{1,2,3}	14	不可行
5	{4}	5	25	13	{1,2,4}	15	不可行
6	{1,2}	10	54	14	{1,3,4}	16	不可行
7	{1,3}	11	不可行	15	{2,3,4}	12	不可行
8	{1,4}	12	不可行	16	{1,2,3,4}	19	不可行

对于一个具有n个元素的集合，其子集数量是2^n，所以，不论生成子集的算法效率有多高，蛮力法都会导致一个$\Omega(2^n)$的算法。

北京郵電大學

2. 任务分配问题

- 某软件开发项目主管需要将某个项目中的5个独立模块的开发任务分配给5个程序员，每个程序员只能分配到1个任务。通过已有的项目开发经验和程序员对任务的评估，得到5个程序员各自完成所有模块所需时间的估算表
- 那么各模块应如何分配给不同的程序员使总时间成本最短？

	A	B	C	D	E	F
1	时间估算	模块1	模块2	模块3	模块4	模块5
2	程序员1	11	13	15	9	17
3	程序员2	10	25	11	24	19
4	程序员3	14	18	12	23	22
5	程序员4	10	14	11	21	23
6	程序员5	22	23	18	20	25

假设有n个任务需要分配给n个人执行，每个任务只分配给一个人，每个人只分配一个任务，且第j个任务分配给第i个人的成本是$C[i, j]$（$1 \leqslant i, j \leqslant n$），任务分配问题要求找出总成本最小的分配方案。

下图是一个任务分配问题的成本矩阵，矩阵元素$C[i, j]$代表将任务j分配给人员i的成本。

	任务1	任务2	任务3	
C=	9	2	7	人员1
	6	4	3	人员2
	5	8	1	人员3

任务分配问题就是在分配成本矩阵中的每一行选取一个元素，这些元素分别属于不同的列，并且元素之和最小。

可以用一个n元组$(j_1, j_2, \ldots, j_n)$来描述任务分配问题的一个可能解，其中第i个分量j_i（$1 \leqslant i \leqslant n$）表示在第$i$行中选择的列号

序号	分配方案	总成本
1	1, 2, 3	9+4+1=14
2	1, 3, 2	9+3+8=20
3	2, 1, 3	2+6+1=9
4	2, 3, 1	2+3+5=10
5	3, 1, 2	7+6+8=21
6	3, 2, 1	7+4+5=16

任务1 任务2 任务3

$$C=\begin{pmatrix} 9 & 2 & 7 \\ 6 & 4 & 3 \\ 5 & 8 & 1 \end{pmatrix}$$

人员1
人员2
人员3

由于任务分配问题需要考虑的排列数量是$n!$，所以，除了该问题的一些规模非常小的实例，蛮力法几乎是不实用的。

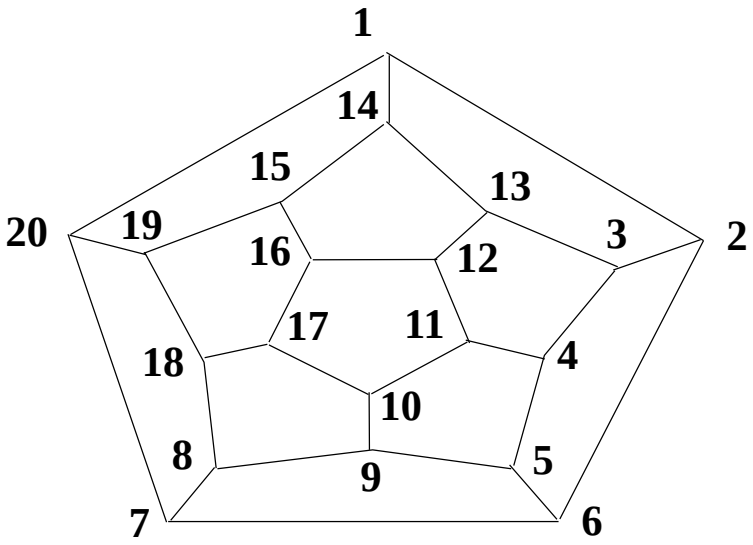
北京郵電大學

图问题中的蛮力法

1. 哈密顿回路问题
2. TSP问题

1. 哈密顿回路问题

著名的爱尔兰数学家哈密顿（William Hamilton，1805—1865）提出了著名的周游世界问题。他用正十二面体的20个顶点代表20个城市，要求从一个城市出发，经过每个城市恰好一次，然后回到出发城市。



正十二面体的展开图，按照图中的顶点编号所构成的回路，就是哈密顿回路的一个解。

使用蛮力法寻找哈密顿回路的基本思想是：对于给定的无向图$G=(V, E)$，首先生成图中所有顶点的排列对象$(v_{i1}, v_{i2}, \ldots, v_{in})$，然后依次考察每个排列对象是否满足以下两个条件：

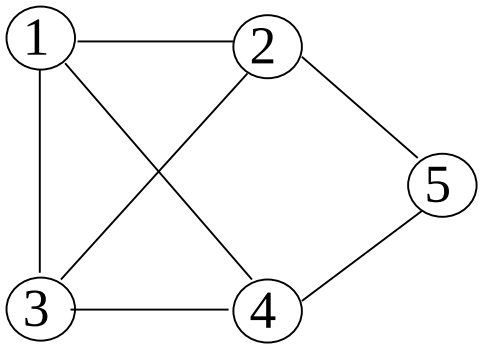
（1）相邻顶点之间存在边，即

$$(v_{ij}, v_{ij+1}) \in E \quad (1 \le j \le n-1)$$

（2）最后一个顶点和第一个顶点之间存在边，即

$$(v_{in}, v_{i1}) \in E$$

满足这两个条件的回路就是哈密顿回路。



(a) 一个无向图

路径	$(v_{ij}, v_{ij+1}) \in E$	$(v_{in}, v_{i1}) \in E$
12345	1→2→3→4→5（是）	否
12354	1→2→3 5→4 （否）	是
12435	1→2 4→3 5 （否）	否
12453	1→2 4→5 3 （否）	是
12534	1→2→5 3→4 （否）	是
12543	1→2→5→4→3（是）	是

(b) 哈密顿回路求解过程

显然，蛮力法求解哈密顿回路在最坏情况下需要考察所有顶点的排列对象，其时间复杂性为$O(n!)$。

2. TSP问题

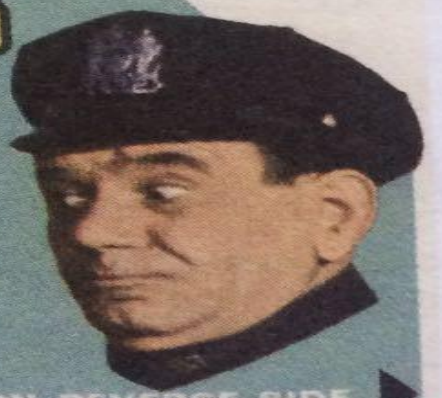
1962年春天，宝洁公司发起了一场广告宣传活动，在应用数学家中引起了不小的反响。活动的重头戏是一项竞赛，奖金高达1万美元，在当时足以买下一座房子。参赛规则如下：假设Toody和Muldoon打算开车环游美国，地图上用点标出的33个地点都要游览，并且要走满足条件的路线中最短的一条。请你为他们规划一条旅行路线，以伊利诺伊州的芝加哥市为旅途的起点和终点，依次用线连接各地点，并使得总里程最短。

救命啊！我们迷路了！

支援54号车，赢现金大奖

54名　各奖1000美元

1名　大奖10 000美元



Help Toody and Muldoon find the shortest round trip route to visit all 33 locations shown on the map.

All you do is draw connecting straight lines from location to location to show the shortest round trip route.

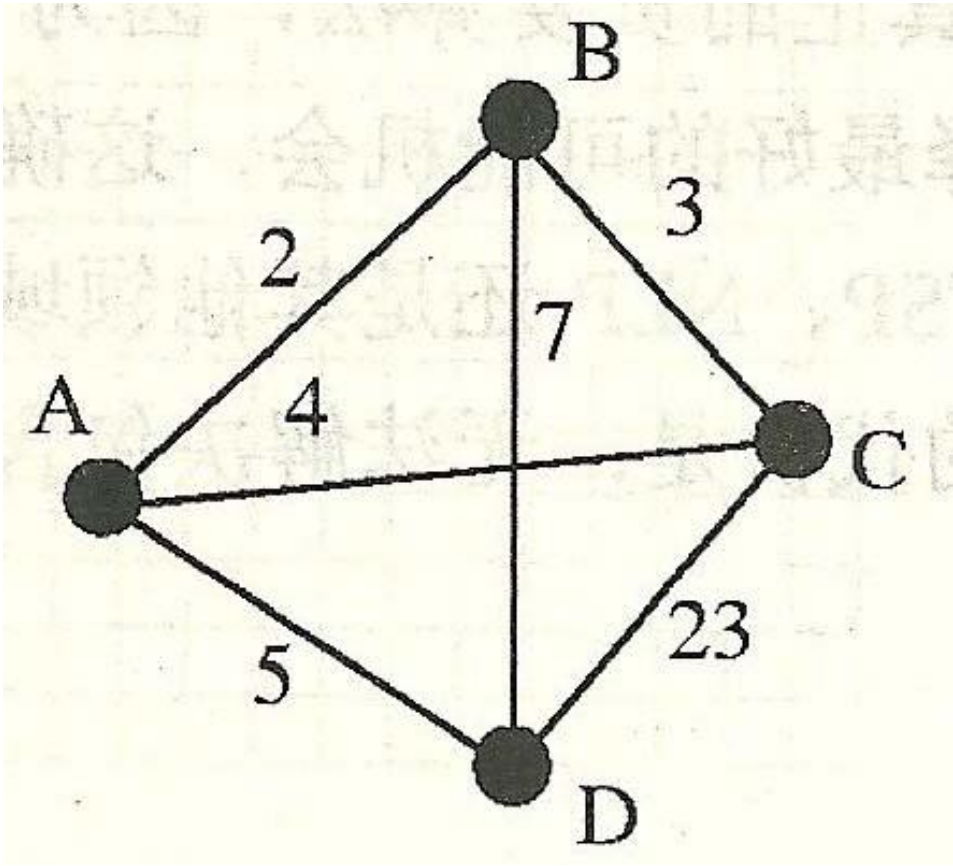
HERE'S THE CORRECT START...

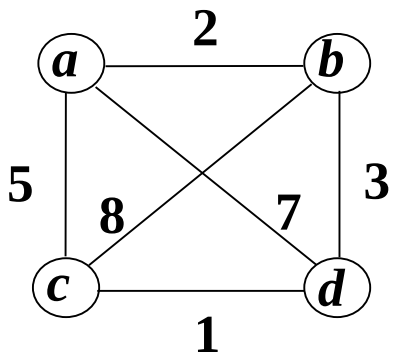
Begin at Chicago, Illinois. From there, lines show correct route as far as Erie, Pennsylvania. Next, do you go to Carlisle, Pennsylvania or Wana, West Virginia? Check the easy instructions on back of this entry blank for details.

OFFICIAL RULES ON REVERSE SIDE

TSP问题是指旅行家要旅行*n*个城市然后回到出发城市，要求各个城市经历且仅经历一次，并要求所走的路程最短。该问题又称为货担郎问题、邮递员问题、售货员问题，是图问题中最广为人知的问题。

用蛮力法解决TSP问题，可以找出所有可能的旅行路线，从中选取路径长度最短的简单回路。





序号	路径	路径长度	是否最短
1	$a→b→c→d→a$	18	否
2	$a→b→d→c→a$	11	是
3	$a→c→b→d→a$	23	否
4	$a→c→d→b→a$	11	是
5	$a→d→b→c→a$	23	否
6	$a→d→c→b→a$	18	否

某些不同的路径，不同的只是路径的方向，因此，可以将这个数量减半，则可能的解有$(n-1)!/2$个。这是一个非常大的数，随着n的增长，TSP问题的可能解也在迅速地增长，例如：

- 一个10城市的TSP问题有大约有180,000个可能解。
- 一个20城市的TSP问题有大约有

  60,000,000,000,000,000个可能解。

- 一个50城市的TSP问题有大约10^{62}个可能解，而一个行星上也只有10^{21}升水。

❖ 用蛮力法求解TSP问题，只能解决问题规模很小的实例。

TSP问题

兰德公司 49个→ IBM 64→1975年的80→1977年120
→1993年 3038

1998年，解决了美国13509个城市之间的TSP问题
2001年，解决了德国15112个城市之间的TSP问题

解决15112个城市之间的TSP问题，共使用了美国Rice大学和普林斯顿大学之间网络互连的，由速度为500MHz 的Compaq EV6 Alpha 处理器组成的110台计算机，所有计算机花费的时间之和为22.6年。

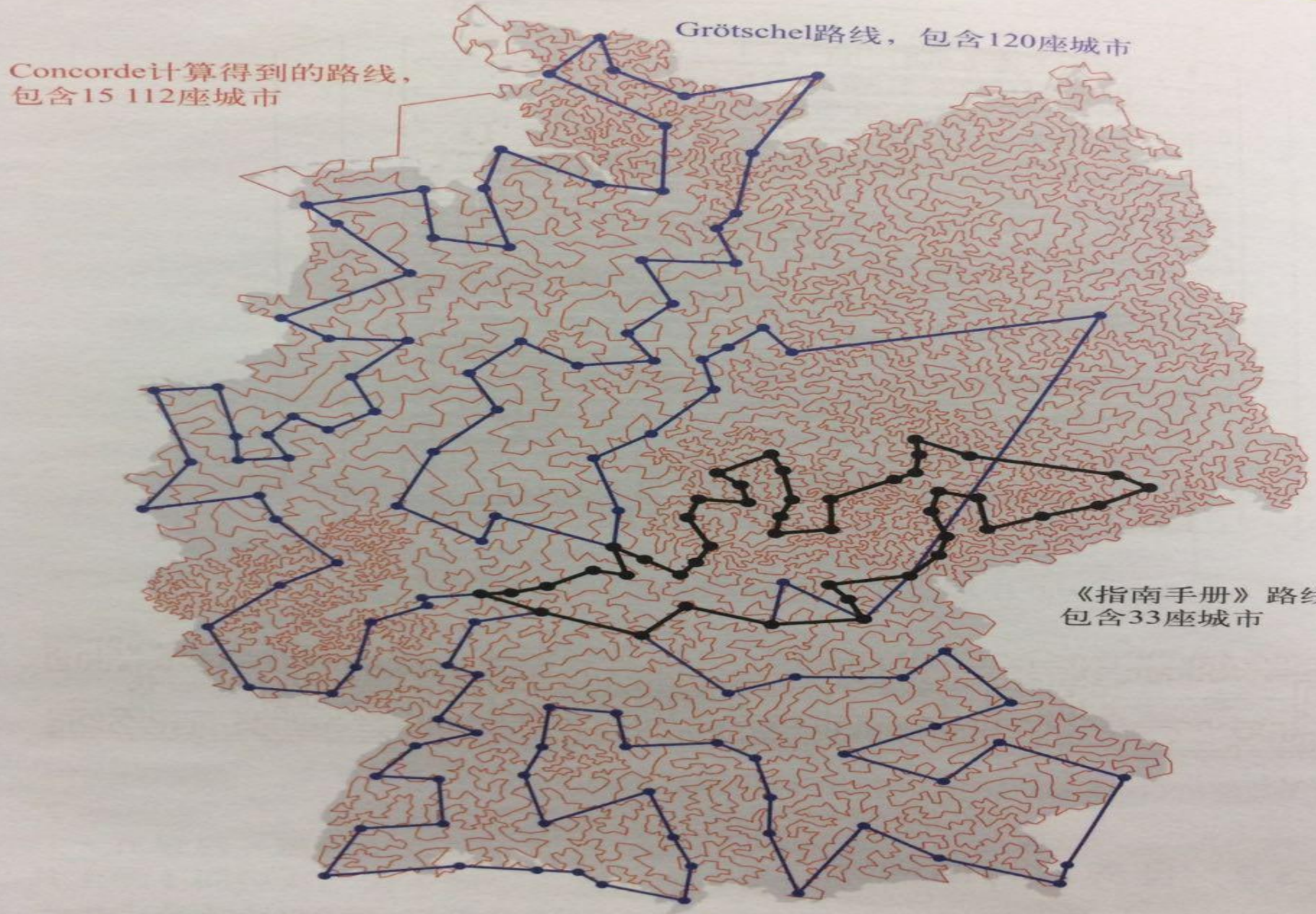


图 1-9　周游德国的三条路线

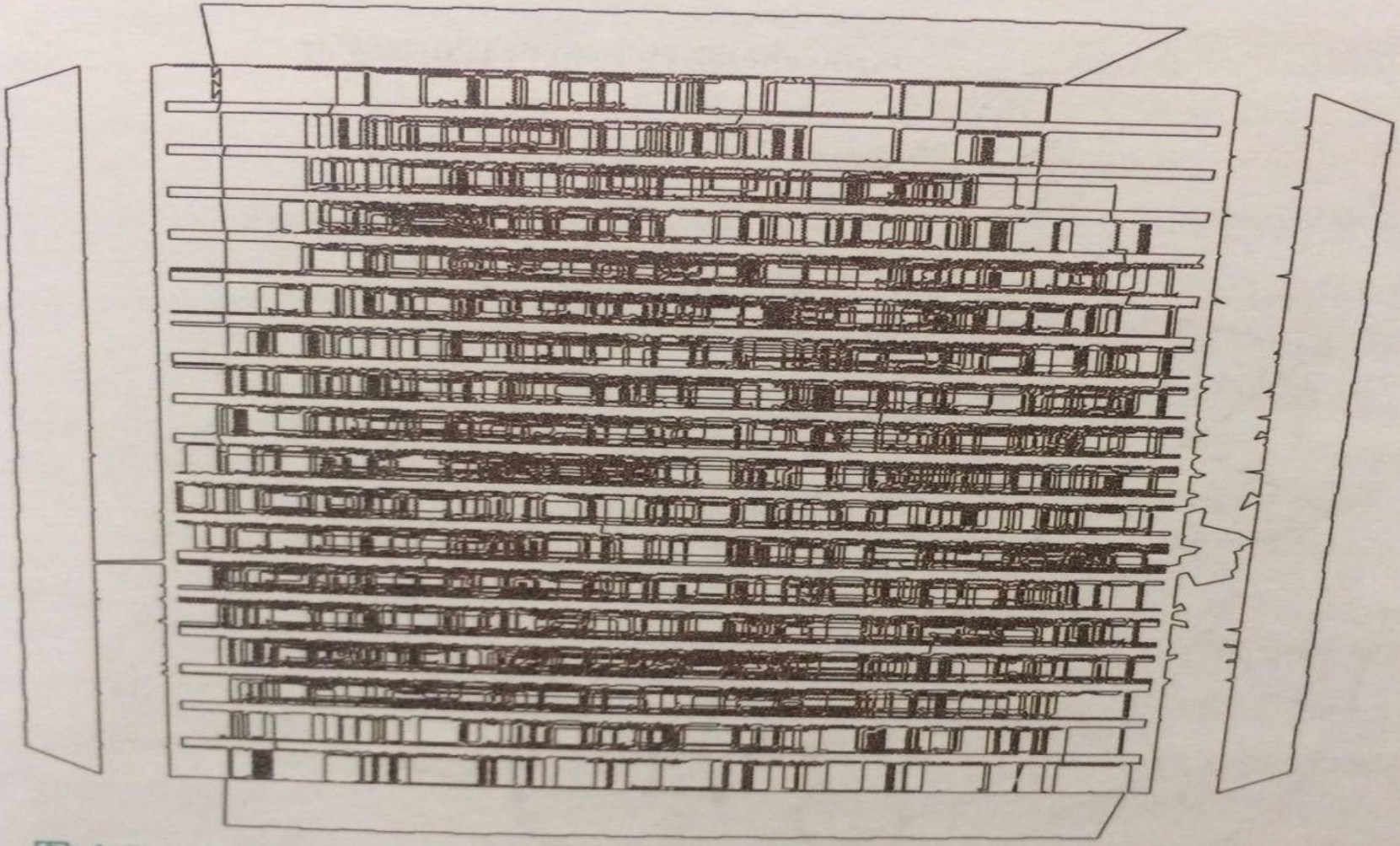


图 1-7　包含 85 900 座城市的 TSP 路线，题目出自计算机芯片应用

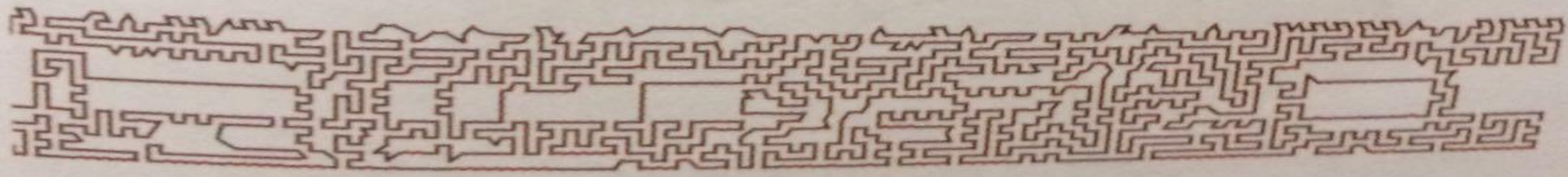
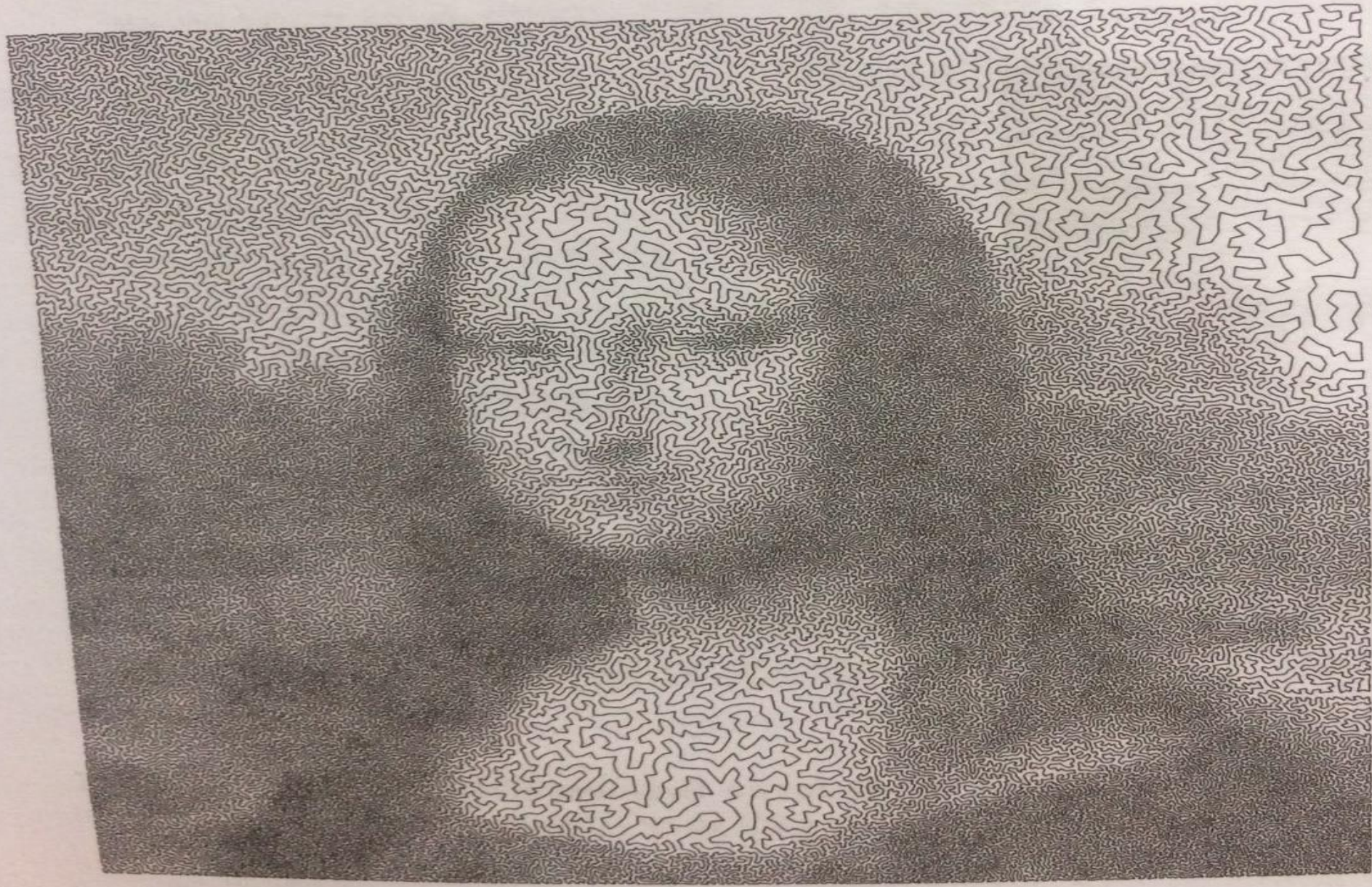


图 1-8　包含 85 900 座城市的 TSP 路线局部放大图

所示的《蒙娜丽莎》TSP，总共包含 100 000 座城市



思考题

- 某地刑侦大队对涉及六个嫌疑人的一桩疑案进行分析：（1）A、B至少有一人作案；（2）A、E、F三人中至少有两人参与作案；（3）A、D不可能是同案犯；（4）B、C或同时作案，或与本案无关；（5）C、D中有且仅有一人作案；（6）如果D没有参与作案，则E也不可能参与作案。试用蛮力法设计算法将作案人找出来。

主观题 10分

设置

某地刑侦大队对涉及六个嫌疑人的一桩疑案进行分析：（1）A、B至少有一人作案；（2）A、E、F三人中至少有两人参与作案；（3）A、D不可能是同案犯；（4）B、C或同时作案，或与本案无关；（5）C、D中有且仅有一人作案；（6）如果D没有参与作案，则E也不可能参与作案。试用蛮力法设计算法将作案人找出来。

正常使用主观题需2.0以上版本雨课堂

作答

练习

TSP问题是指旅行家要旅行*n*个城市，要求各个城市经历且仅经历一次然后回到出发城市，并要求所走的路程最短。