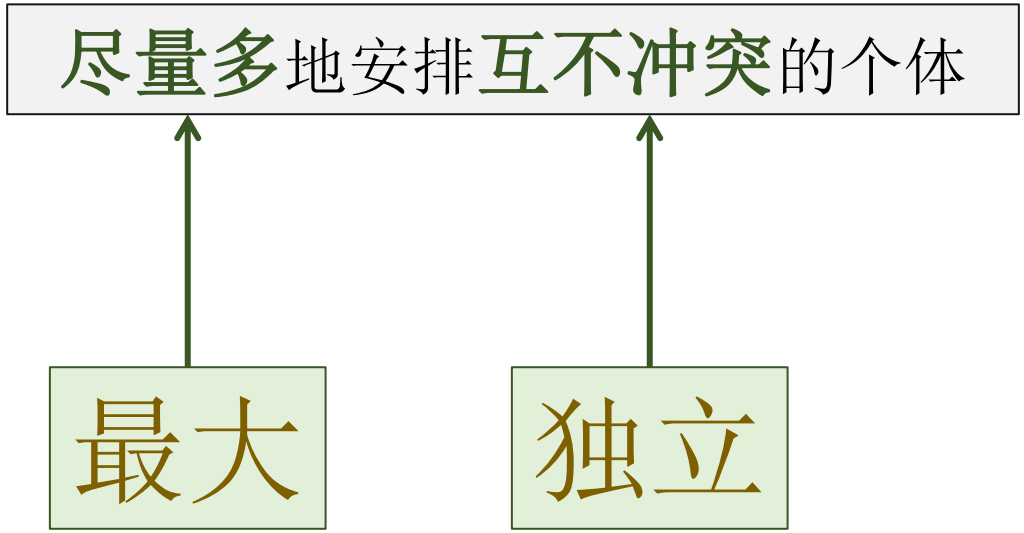

C++
算法

深度优先搜索
Depth-first Search

棋盘格
最大独立集问题

最大独立集



例题：四个斜眼枪手

在3*3格棋盘上要放四个斜眼枪手。每个斜眼枪手可以朝四个斜方向开枪：左上/右上/左下/右下。要求枪手互相不在对方开枪方向上，请输出所有摆放方案，1代表枪手0代表空位。

111 000 010	110 000 110	101 101 000	100 101 100

请找一个另外方案

如何枚举所有方案

```
12 bool valid(int x,int y){
13     for(int i=1;i<=min(x,y);i++)
14         if(f[x-i][y-i])return 0;
15     for(int i=1;i<=min(x,N-1-y);i++)
16         if(f[x-i][y+i])return 0;
17     return 1;
18 }
19 void dfs(int x,int y,int c){
20     if(c==GUNS){ans++;print();return;}
21     if(x==N)return;
22     int nx=(y==N-1?x+1:x);
23     int ny=(y==N-1?0:y+1);
24     if(valid(x,y)){
25         f[x][y]=1;
26         dfs(nx,ny,c+1);
27         f[x][y]=0;
28     }
29     dfs(nx,ny,c);
30 }
```

例题：六个斜眼枪手

在4*4格棋盘上要放六个斜眼枪手。每个斜眼枪手可以朝四个斜方向开枪：左上/右上/左下/右下。要求枪手互相不在对方开枪方向上，请输出所有摆放方案，1代表枪手0代表空位。

```
1111
0000
0000
0110
```

```
1110
0000
0000
1110
```

```
1101
0001
1000
0010
```

```
1100
0001
1000
1010
```

请找一个另外方案

如何枚举所有方案

```
12 bool valid(int x,int y){
13     for(int i=1;i<=min(x,y);i++)
14         if(f[x-i][y-i])return 0;
15     for(int i=1;i<=min(x,N-1-y);i++)
16         if(f[x-i][y+i])return 0;
17     return 1;
18 }
19 void dfs(int x,int y,int c){
20     if(c==GUNS){ans++;print();return;}
21     if(x==N)return;
22     int nx=(y==N-1?x+1:x);
23     int ny=(y==N-1?0:y+1);
24     if(valid(x,y)){
25         f[x][y]=1;
26         dfs(nx,ny,c+1);
27         f[x][y]=0;
28     }
29     dfs(nx,ny,c);
30 }
```

例题：八个斜眼枪手

在5*5格棋盘上要放八个斜眼枪手。每个斜眼枪手可以朝四个斜方向开枪：左上/右上/左下/右下。要求枪手互相不在对方开枪方向上，请输出所有摆放方案，1代表枪手0代表空位。

10010 10000 10001 00001 11000	10001 10001 10001 10001 00000	10000 10001 10001 10001 10000	01111 00000 00000 00000 01111

请找一个另外方案

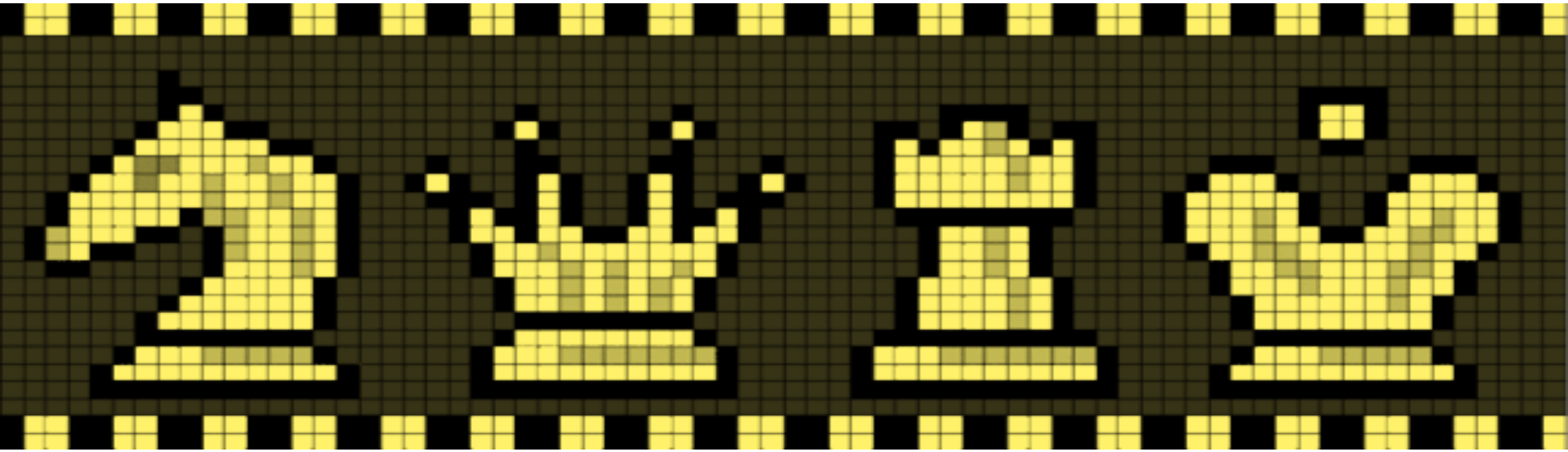
如何枚举所有方案

```
12 bool valid(int x,int y){
13     for(int i=1;i<=min(x,y);i++)
14         if(f[x-i][y-i])return 0;
15     for(int i=1;i<=min(x,N-1-y);i++)
16         if(f[x-i][y+i])return 0;
17     return 1;
18 }
19 void dfs(int x,int y,int c){
20     if(c==GUNS){ans++;print();return;}
21     if(x==N)return;
22     int nx=(y==N-1?x+1:x);
23     int ny=(y==N-1?0:y+1);
24     if(valid(x,y)){
25         f[x][y]=1;
26         dfs(nx,ny,c+1);
27         f[x][y]=0;
28     }
29     dfs(nx,ny,c);
30 }
```

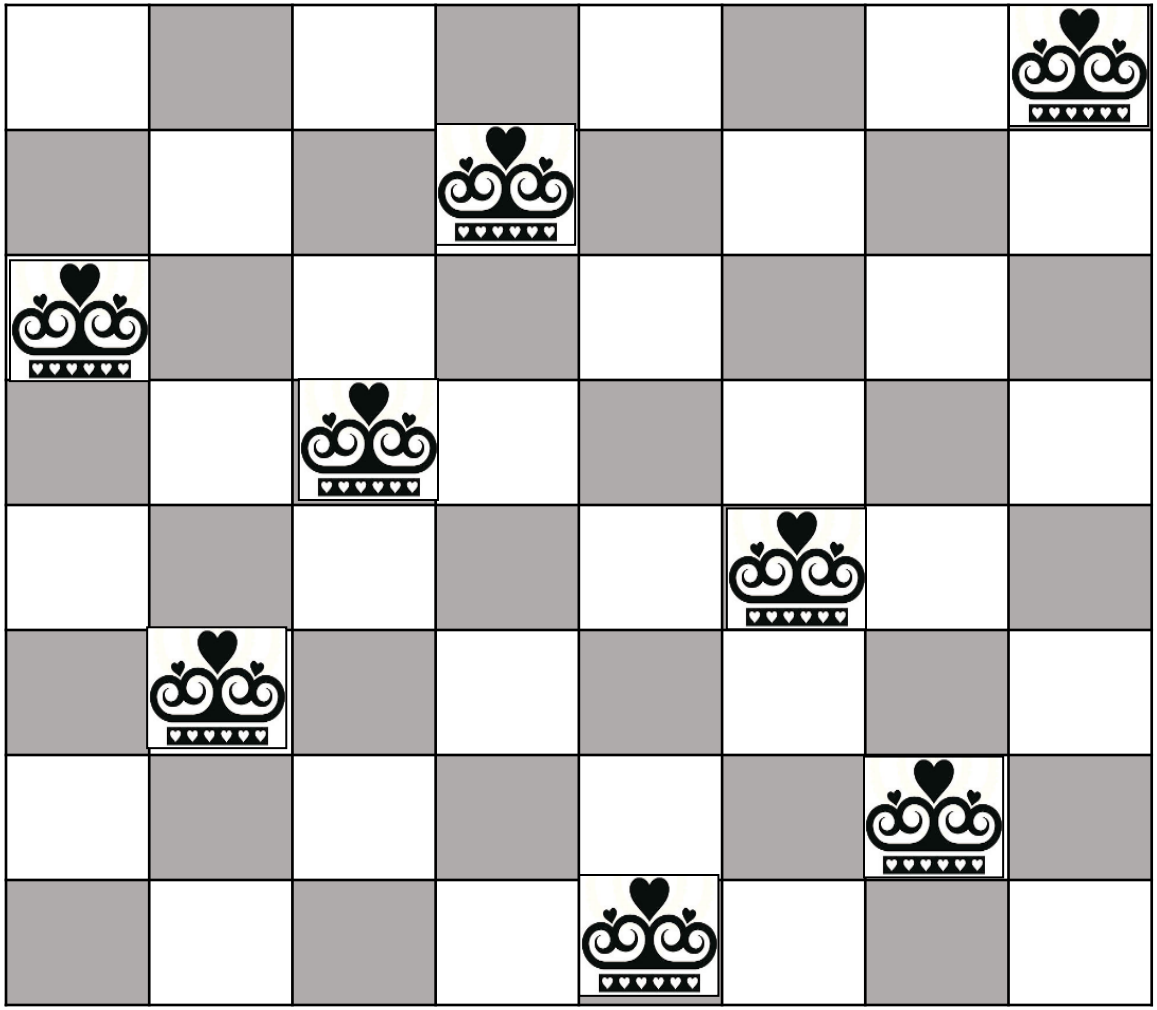
八皇后

著名的八皇后问题是国际象棋棋手贝瑟尔于1848年提出

在8×8格的国际象棋上摆放八个皇后，使其不能互相攻击，即任意两个皇后都不能处于同一行、同一列或同一斜线上，问有多少种摆法



八皇后：方案举例



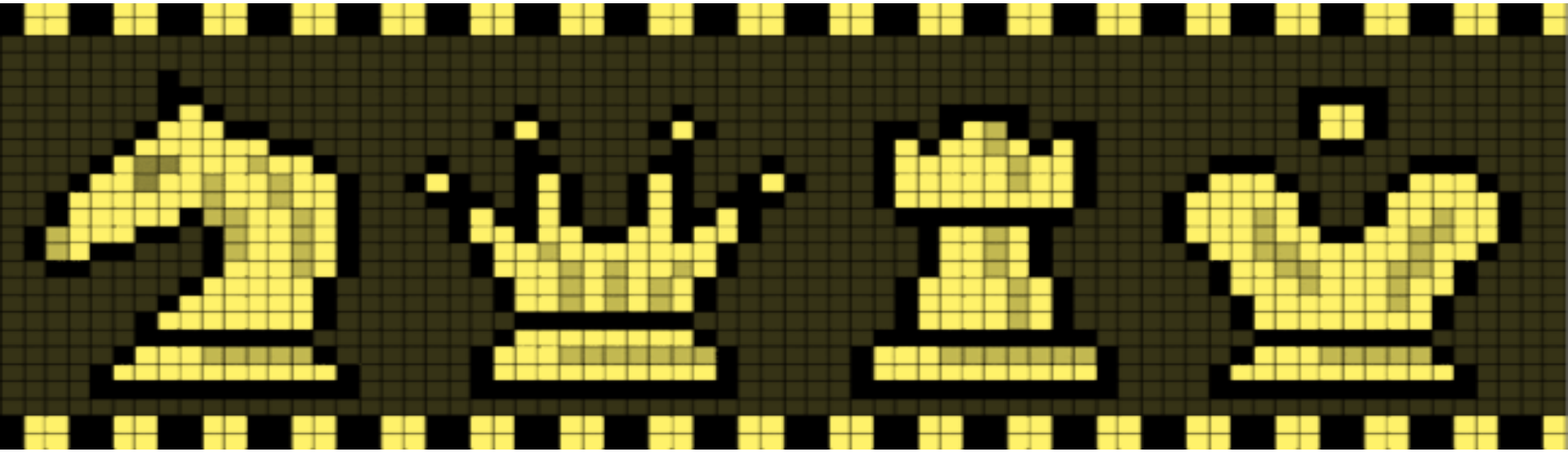
共有多少种摆法

八皇后

高斯认为有76种方案

1854年在柏林的象棋杂志上不同的作者发表了40种不同的解

后人共解出92种方案



八皇后算法

1

逐格放置：
对于每一格考虑是否放皇后

2

逐行放置：
对于每一行考虑放皇后在哪一列

八皇后：逐格放置

```
11  bool valid(int x,int y){
12      for(int i=1;i<=min(x,y);i++)
13          if(f[x-i][y-i])return 0;
14      for(int i=1;i<=min(x,N-1-y);i++)
15          if(f[x-i][y+i])return 0;
16      for(int i=0;i<x;i++)
17          if(f[i][y])return 0;
18      for(int i=0;i<y;i++)
19          if(f[x][i])return 0;
20      return 1;
21  }
22  void dfs(int x,int y,int c){
23      if(c==N){ans++;print();return;}
```

八皇后：逐行放置

```
12  bool valid(int x,int y){
13      return!clmn[y]&&!d1[x+y]&&!d2[x-y+N-1];
14  }
15  void dfs(int x){
16      if(x==N){ans++;print();return;}
17      for(int y=0;y<N;y++)
18          if(valid(x,y)){
19              f[x][y]=clmn[y]=d1[x+y]=d2[x-y+N-1]=1;
20              dfs(x+1);
21              f[x][y]=clmn[y]=d1[x+y]=d2[x-y+N-1]=0;
22          }
23  }
24  int main(){
25      dfs(0);
26      return 0;
27  }
```