《算法复杂性理论》
第3讲 穷举法

山东师范大学信息科学与工程学院
段会川
2014年9月

目录

- □ 穷举法的定义
- □ 穷举法的通用算法
- □ 百元买白鸡问题的穷举法
- □ 素数测试问题的穷举法
- □ 0-1背包问题的穷举法
- □ TSP问题的穷举法

穷举法定义

- □ **In computer science, brute-force search or exhaustive search, also known as generate and test, is a very general problem-solving technique that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement.**

Wikipedia

穷举法定义

- □ **In order to apply brute-force search, one must implement four procedures, first, next, valid, and output. They should take the problem instance P a parameter, and do the following:**
  1. **first (P): generate a first candidate solution for P.**
  2. **next (P, c): generate the next candidate for P after the current one c.**
  3. **valid (P, c): check whether candidate c is a solution for P.**
  4. **output (P, c): use the solution c of P as appropriate to the application.**
- □ **The next procedure must also tell when there are no more candidates for the instance P, usually by returning a "null candidate", Λ. Likewise the first procedure should return Λ if there are no candidates at all for the instance P.**

Wikipedia

穷举法的通用算法

- □ 算法名称：通用穷举法(**ExhaustiveSearch**)
- □ 输入：问题实例**P**
- □ 输出：问题的解

```
1: c ← first(P)
2: while c ≠ Λ
3:     if valid(P,c) then output(P, c)
4:     c ← next(P)
5: end while
```

百元买百鸡问题

- □ 鸡翁一值钱5，鸡母一值钱3，鸡雏三值钱1。百钱买百鸡，问鸡翁、母、雏各几何？”
- □ 算法问题：**n**元买**n**鸡问题
- □ 数学模型
  - ■ $x+y+z=n$
  - ■ $5x+3y+\frac{z}{3}=n$

百元买百鸡问题

```
for x=1 to n
    for y=1 to n
        for z=1 to n
            if x+y+z=n then
                if z mod 3 = 0 then
                    if 5x+3y+z/3 = n then
                        print x,y,z
```

百元买百鸡问题

```
for x=1 to n/5
    for y=1 to n/3
        z=n-x-y
        if z mod 3 = 0 then
            if 5x+3y+z/3 = n then
                print x,y,z
```

素数测试—试除法(trial division)

- 试除法是测试一个数*N*是否为素数的蛮力方法
  - 由于如果*N*有大于$\sqrt{N}$的因子*p*，则一定有一个小于$\sqrt{N}$因子*q*，因此只要用小于$\sqrt{N}$每个素数去试除*N*，如果找到一个数能够除尽*N*，则*N*就不是素数，如果所有的素数都除不尽*N*，则*N*必是素数
  - 上述方法未考虑获得所有小于$\sqrt{N}$的素数的代价
  - 也未考虑计算$\sqrt{N}$的代价

素数测试—朴素(na ïve)试除法伪代码

```
ret = true
for i=2 to sqrt(N)
    if N MOD i =0
        ret = false
        break
    end if
return ret
```

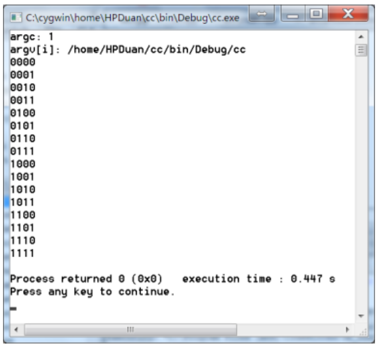
素数测试—朴素(na ïve)试除法伪代码

```
ret = true
i = 2
do
    if N MOD i =0
        ret = false
        break
    end if
    i = i+1
while i*i<=N
```

0-1背包问题的穷举法

```
vector<int> KSv;
void Knapsack(int n) {
  if (n==0) {
    for (auto x:KSv)
      cout << x;
    cout << endl;
    return; }
  KSv.push_back(0);
  Knapsack(n-1);
  KSv.pop_back();
  KSv.push_back(1);
  Knapsack(n-1);
  KSv.pop_back();
}
```

0-1背包问题的穷举法



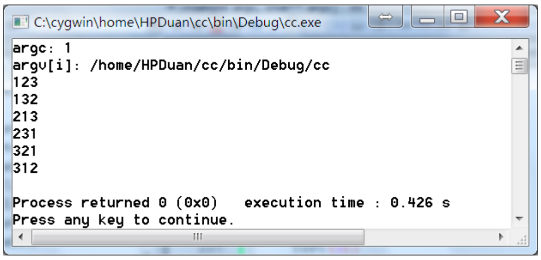
```
argc: 1
argv[i]: /home/HPDuan/cc/bin/Debug/cc
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

Process returned 0 (0x0)   execution time : 0.447 s
Press any key to continue.
```

TSP问题的穷举法

```
vector<int> TSPv;
void TSP(int i, int n) {
   if (i==n-1) {
      for (auto x:TSPv)
         cout << x+1;
      cout << endl;
      return;
   }
   for (int j=i; j<n; ++j) {
      swap(TSPv[i],TSPv[j]);
      TSP(i+1, n);
      swap(TSPv[i],TSPv[j]);
   }
}
```

```
void TSPt(int n)
{
   for (int i=0; i<n; ++i)
      TSPv.push_back(i);
   TSP(0,n);
}
```

TSP问题的穷举法



```
argc: 1
argv[i]: /home/HPDuan/cc/bin/Debug/cc
123
132
213
231
321
312

Process returned 0 (0x0)   execution time : 0.426 s
Press any key to continue.
```

目录

- □ 穷举法的定义
- □ 穷举法的通用算法
- □ 百元买白鸡问题的穷举法
- □ 素数测试问题的穷举法
- □ 0-1背包问题的穷举法
- □ TSP问题的穷举法

The End

Thanks！