假设将n个圆盘从A移到C
假设圆盘编号为1~n且编号越大圆盘越大

算法步骤

1. **问题分析**:
   如果只有一个圆盘，直接将其从 A 移到 C。
   如果有多个圆盘:
   - 先将前 $n-1$ 个圆盘从 A 移到 B，借助 C。
   - 将第 n 个圆盘从 A 移到 C。
   - 再将 $n-1$ 个圆盘从 B 移到 C，借助 A。
2. **递归公式**:
   $T(n)=2T(n-1)+1$，其中 $T(n)$ 是移动 n 个圆盘所需的最少步数。
3. **根据生成函数求解**$T(n)$
   $G(x)=\sum_{n=0}^{\infty}T(n)x^n$
   $2xG(x)=\sum_{n=0}^{\infty}T(n)x^{n+1}=\sum_{n=1}^{\infty}2T(n-1)x^n$
   所以:
   $(1-2x)G(x)=T(0)+\sum_{n=1}^{\infty}(T(n)-2T(n-1))x^n$
   代入递归公式 $T(n)=2T(n-1)+1$:
   $(1-2x)G(x)=T(0)+\sum_{n=1}^{\infty}x^n$
   解得:
   $G(x)=\frac{T(0)+\frac{x}{1-x}}{1-2x}$
   而:
   $T(0)=0$
   所以:
   $G(x)=\frac{\frac{x}{1-x}}{1-2x}=\frac{x}{(1-x)(1-2x)}$
   $G(x)=\frac{x}{(1-x)(1-2x)}=\frac{1}{1-2x}-\frac{1}{1-x}$
   所以:
   $G(x)=\sum_{n=0}^{\infty}(2^n-1)x^n$
   所以:
   $T(n)=2^n-1$

Python 实现

```python
def hanoi(n, source, target, auxiliary):
    """
    解决汉诺塔问题的递归函数。
    参数:
    n         : 圆盘数量
    source    : 起始柱子
    target    : 目标柱子
    auxiliary : 辅助柱子
    """
    if n == 1:
        print(f"将圆盘 1 从 {source} 移动到 {target}")
        return
    # 将前 n-1 个圆盘从 source 移到 auxiliary，借助 target
    hanoi(n-1, source, auxiliary, target)
    # 将第 n 个圆盘从 source 移到 target
    print(f"将圆盘 {n} 从 {source} 移动到 {target}")
    # 将前 n-1 个圆盘从 auxiliary 移到 target，借助 source
    hanoi(n-1, auxiliary, target, source)

# 测试
n = 3  # 圆盘数量
hanoi(n, "A", "C", "B")
```

示例输出（$n = 3$）：

```
将圆盘 1 从 A 移动到 C
将圆盘 2 从 A 移动到 B
将圆盘 1 从 C 移动到 B
将圆盘 3 从 A 移动到 C
将圆盘 1 从 B 移动到 A
将圆盘 2 从 B 移动到 C
将圆盘 1 从 A 移动到 C
```

这个算法的时间复杂度是 $O(2^n)$，也是汉诺塔问题的最优解。