不会影响其他键值对的插入。
2. **事件发生的概率相同**：假设哈希函数是均匀和随机的，那么每个键值对被插入到任意一个桶中的概率是相同的。
3. **稀疏事件**：对于一个较大的哈希表（即桶的数量很多），每个桶中的元素数量相对较少。

基于这些条件，可以将哈希表中每个桶中的元素数量看作是服从泊松分布的。对于负载因子为 α\alphaα 的哈希表，某个桶插入的元素数量为 k，k 服从泊松分布：

P(X=k)=αke−αk!P(X = k) = \frac{\alpha^k e^{-\alpha}}{k!}P(X=k)=k!αke−α

2.2 几何分布

几何分布用于描述在独立重复试验中，第一次成功所需的试验次数。其概率质量函数为：

P(Y=n)=(1−p)n−1pP(Y = n) = (1 - p)^{n-1} pP(Y=n)=(1−p)n−1p

其中：

* YYY 是第一次成功所需的试验次数。
* ppp 是每次试验成功的概率。

几何分布的期望值为：

E(Y)=1pE(Y) = \frac{1}{p}E(Y)=p1

在 `HashMap` 中，找到一个非空桶可以看作是一次成功的试验，而每个桶是否为空是独立的事件。

在 `HashMap` 中，找到一个非空桶的概率为：

P(X>0)=1−P(X=0)=1−e−αP(X > 0) = 1 - P(X = 0) = 1 - e^{-\alpha}P(X>0)=1−P(X=0)=1−e−α

查找非空桶的期望时间服从几何分布，成功概率 ppp 为 1−e−α1 - e^{-\alpha}1−e−α，期望查找时间为：

E(T)=11−e−αE(T) = \frac{1}{1 - e^{-\alpha}}E(T)=1−e−α1

2.3 平均查找时间

在 `HashMap` 中，查找一个元素的平均时间包括找到桶的位置和在桶内查找元素的时间。对于负载因子为 α\alphaα 的 `HashMap`。

* 查找到一个非空桶的概率为 1−e−α1 / ((1 - e_minus_a) ** 3)
        except OverflowError:
            return float('inf')

    for i in range(max_iterations):
        f_value = function(alpha)
        if abs(f_value) < epsilon:
            break
        derivative_value = derivative(alpha)
        if derivative_value == 0 or math.isinf(derivative_value) or math.isnan(derivative_value):
            break
        step = f_value / derivative_value
        if abs(step) > max_step:
            step = max_step if step > 0 else -max_step
        alpha = alpha - step
        # Debug print statement to track iterations
        print(f"Iteration {i}: alpha = {alpha}, f_value = {f_value}, derivative_value = {derivative_value}, step = {step}")

    return alpha

# Run the function to find the optimal load factor
optimal_alpha = find_optimal_load_factor()
print("Optimal Load Factor:", optimal_alpha)

```
```