算法设计与分析第七章学习指南

视频

https://mooc.study.163.com/learn/1000005000?tid=1000005001#/learn/content

(算法设计与分析(进阶篇)第六讲)

阅读

算法导论(第三版) 第 17 章，第 19 章，第 21 章

练习题

1. 请比较平摊分析中聚集法和势能法的优劣

2. 请讨论平摊分析中会计法和势能法的区别

3. 请讨论平摊分析和概率分析的区别

4. 对某个数据结构执行大小为 n 的一个操作序列，若 i 为 2 的整数幂，则第 i 个操作的代价 为 i，否则为 1。请利用会计方法分析每次操作的平摊代价

5. Bill 提出了一种叫做翻转堆栈的数据结构，翻转堆栈只支持 Flipping-Push() 函数。在每 次 Flipping-Push() 中，首先压栈，并检查堆栈中的对象数目数是否是 2 的幂（2^i)。如果 是，则堆栈中的所有对象将翻转。例如我们使用 Flipping-Push() 将对象 1,2,3,4 压入堆栈 中，堆栈中的内容（从底向上看）在每次压栈后为:

(1) ⇒ (2, 1) ⇒ (2, 1, 3) ⇒ (4, 3, 1, 2) Bill

请求你分别使用聚集分析、会计方法、势能方法分析 Flipping-Push() 函数的平摊代价 (amortized cost), 堆栈反转的代价等于堆栈现有对象数目

6. 有两个堆栈 A 和 B，都可以使用以下 5 种操作（A 大小为 n，B 大小为 m）:

PushA(x):x 压入堆栈 A，实际代价 =1 PushB(x):x 压入堆栈 B，实际代价 =1

MultiPopA(k): 从 A 中弹出 min{k, n} 个元素，实际代价 =min{k, n}

MultiPopB(k): 从 B 中弹出 min{k, m} 个元素，实际代价 =min{k, m}

Transfer(k): 从 A 中弹出元素并压入 B 中，直到转移 k 个元素或 A 为空，实际代价 = 转移元素数目 (a)MultiPopA(k),MultiPopB(k),Transfer(k) 三种操作最坏情况下的时间复杂度为多少？

(b) 定义一个势能函数 $\Phi(n, m)$, 用它证明以上操作平摊代价为 O(1)

7. Suppose our multistack data structure includes an operation multipush(m, x) that pushes m copies of x. Analyze the amortized running time using each of the aggregate, accounting, and potential methods. Is it possible to implement such a multistack so that it has O(1) amortized cost per operation?

8. Suppose our binary counter includes an operation decrement( ) that subtracts one from its value. Analyze the amortized running time using each of the aggregate, accounting, and potential methods. Is it possible to implement such a binary counter so that it has O(1) amortized cost per operation?

9. The dynamic table we discuss in class maintains a load factor between 1/4 and 1. Redefine the insert and remove operations to maintain a load factor between any arbitrary given load factors f1 and f2, where 0 < f1 < f2 <= 1. For example, consider f1 = 1/3 and f2 = 3/4. For which values of f1 and f2 is it possible to implement such a dynamic table so that it has O(1) amortized cost per operation?

10. A min-priority queue provides operations insert(x) and removeMin( ). The off-line-minimum problem has input a sequence of n insert(x) and p removeMin( ) operations, where each value in {1,2,...,n} is inserted exactly once. The goal is to determine the value returned by each removeMin( ), or equivalently, to construct an array R[1...p], where each R[k] is the value returned by the kth removeMin( ). This problem is "off-line" because it can read the entire input sequence before determining any of the returned keys. Show how to efficiently solve the off-line-minimum problem by using a disjoint-sets data structure.

11. We implement a disjoint-sets data structure on values {1,2,...,n} that performs a sequence of O(m) find(x) and union(S,T) operations in O(m lg* n) time. We use two heuristics, union-by-rank and path-compression, to achieve this running time. Show that the same running time can be obtained if union-by-rank is replaced by union-by-size, and/or if path-compression is replaced by either path-splitting or path-halving. What are some practical advantages/disadvantages of each heuristic?

union-by-rank	make the root of the shorter tree point to the root of the taller tree
union-by-size	make the root of the smaller tree point to the root of the larger tree
path-compression	make every encountered node point to the root of the tree
path-splitting	make every encountered node point to its grandparent
path-halving	make each alternate node point to its grandparent

12. Suppose we have a data structure where the cost T(i) of the ith operation is:

$$T(i) = \begin{cases} 2i & \text{if } i \text{ is a power of 2,} \\ 1 & \text{otherwise.} \end{cases}$$

What is the amortized running time of the operation? Use both the aggregate method and the accounting method to analyze the amortized running time.

13. Potential functions: Doubling arrays In this exercise we consider dynamic tables with insertions only (no deletions) using the doubling technique. Can we show that the amortized cost of an insertion is O(1) using the following potential function: Φ(Di) = k, where k is the number of elements in the array?

14. In the Set Union problem we have n elements, that each are initially in n singleton sets, and we want to support the following operations:

• Union(A,B): Merge the two sets A and B into one new set C = A ∪ B destroying the old sets.
• SameSet(x, y): Return true, if x and y are in the same set, and false otherwise.

We can implement it the following way. Initially, give each set a color. When merging two sets, recolor the smallest one with the color of the larger one (break ties arbitrarily). To answer SameSet queries, check if the two elements have the same color.
(1) Analyze the worst case cost of the two operations.
(2) Show that the amortized cost is O(log n) for Union and O(1) for SameSet. That is, show that a any sequence of m unions and l SameSet operations takes time O(m log n + l).
Hint: Give a bound on the number of times an element can be recolored.

15. Professor Bille suggests a simpler version of Splay Trees that he calls Play trees. Play Trees are similar to Splay Trees but uses only single rotations when splaying.

(1) What is the amortized cost of splay(x) if we only use single rotations? Analyze it with the same potential function as we used for Splay Trees.
(2) What is the total (actual) cost of first n inserting elements with keys 1, 2, 3, . . . , n in that order in a Play Tree and then searching for 1, 2, 3, . . . , n (in that order)?
(3) Professor Bille claims that the amortized cost of the operations insert, search and delete in Play trees is
O(log n). "You just need to use a more sophisticated potential function" he says. Could he be correct?

16. Explain how to make a dynamic hashtable with insertions using the doubling technique. Your solution should use Θ(n) space, where n is the number of elements in the hashtable. What is the insertion time of your solution?

17. It is sometimes possible to deamortize data structures, i.e., getting the same bounds worst case as amortized by doing the work in the "background". Show that you can get worst case constant insertion time in dynamic tables, while still having space usage O(n), where n is the number of elements currently in the table.

Hint. Use the doubling technique, but spread out the work on all the insertions.

18. You are a young Prince from the country Algo. The King in the neighboring country Logic has 3 daughters. The oldest one always tells the truth, the youngest one always lies, and the middle one sometimes lies, sometimes tells the truth. You want to marry either the oldest one or the youngest one (since you know she always lies that is as good as the one always telling the truth). The only one you don't want to marry is the middle one. The king is a sneaky man, and he tells you, that you can ask one of the daughters one question. The question should be one with a yes/no answer. After that you have to choose which one to marry. They all look alike, so it is not possible for you to determine which one is which by looking at them. What question should you ask, and which one should you then pick?