A digression on amortised analysis

Giulia Bernardini
giulia.bernardini@units.it

Reference: Part of chapter “Amortized Analysis ” of: Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. *Introduction to algorithms*. (Chapter 17.1 of the third edition)

Amortised analysis

In an amortised analysis, we average the time required to perform a sequence of operations over all the operations performed.

With amortised analysis, we can show that the average cost of an operation over a sequence of operations is small, even though a single operation within the sequence might be expensive.

Probability is not involved; an amortised analysis guarantees the average performance of each operation in the worst case.

Amortised analysis via aggregate analysis

In aggregate analysis, we show that for all n, a sequence of n operations takes worst-case time T(n) in total.

In the worst case, the average cost (or amortised cost) per operation is therefore T(n)/n.

We describe this technique with an example.

Consider a stack S with the usual operations PUSH(S,x), POP(S), and an additional operation MPOP(S,k) which pops k objects from S, or the whole stack if it contains less then k objects.

The cost of MPOP is linear in the number of objects popped, that is, min{k,|S|}, as it can be implemented by making subsequent calls to POP(S).

Amortised analysis via aggregate analysis

What is the total cost of a sequence of n calls to PUSH, POP and MPOP on an initially empty stack S?

The worst-case cost of a MPOP operation in the sequence is O(n), since the size of S is at most n.

Thus the cost of any operation in the sequence is O(n), and since there are n of them, the worst-case cost is $O(n^2)$.

This analysis considers each operation individually and is not tight.

Amortised analysis via aggregate analysis

Although a single MPOP operation can be expensive, any sequence of n PUSH, POP, and MPOP operations on an initially empty stack can cost at most $O(n)$.

The reason is that we can pop an object from the stack only if we have pushed it before. Since we can push at most n objects, we cannot pop more than n of them, thus the total cost of POP and MPOP operations is $O(n)$.

The average cost of an operation in the worst case is $O(n)/n=O(1)$. Thus all three stack operations have an amortized cost of $O(1)$.