ECE374 SP23 HW5

Contributors

Zhirong Chen (zhirong4)

Ziyuan Chen (ziyuanc3)

Problem 4

(a) Given an array A of n integers, find an efficient algorithm to compute the largest sum of a continuous run.

(b) Now assume that there are n numbers (some possibly negative) on a circle, and we wish to find the maximum contiguous sum along an arc of the circle. Give an efficient algorithm for solving this problem.

Solution

(a)

Recurrence relation. For the given array $A[0 \ldots n-1]$, we define $S[i]$ as the largest sum of a continuous run that ends at $A[i]$:

$$S[i] = \begin{cases} A[0] & i = 0 \\ A[i] & i > 0 \text{ and } S[i-1] \leq 0 \\ S[i-1] + A[i] & i > 0 \text{ and } S[i-1] > 0 \end{cases}$$

Memoization. We fill out the memoized array S from left to right. Starting from $S[0] = A[0]$, we start a new continuous run at $A[i]$ if $S[i-1] \leq 0$ (which means there's no point in including the previous run). Otherwise, we extend the previous run by adding $A[i]$ to $S[i-1]$. The final answer is $\max_i S[i]$.

Time complexity. We fill out the linear S in $O(n)$ time. Locating the maximum value (a single scan) takes another $O(n)$. The total time complexity is $O(n)$.

(b)

We extend the solution to **(a)** to the circular case by simply concatenating a copy of the array with itself, or

$$A[i] = \begin{cases} A[i] & i < n \\ A[i-n] & i \geq n \end{cases}, \quad 0 \leq i < 2n$$

Any continuous run that covers the sequence $\{\ldots, A[n-1], A[n], \ldots\}$ indicates that the optimal continuous run is "wrapped around" at the end of the original linear array.

Consequently, the length of $S[0 \ldots 2n-1]$ is also doubled. The algorithm for **(a)** still works, except that the time complexity is now $O(2n) = O(n)$.