Analysis Report — Kadane's Algorithm (Student B)

Zhaniya Abdraiym

SE-2421

Pair 3 (with Baldauren Zaman)

1. Algorithm Overview

Kadane’s Algorithm is a linear-time algorithm for finding the contiguous subarray within a one-dimensional array that has the largest sum. It maintains a running sum of the current subarray and updates the global maximum when the running sum exceeds it. This allows detection of the maximum subarray sum in a single pass, using only constant additional space.

1. Complexity Analysis

* Best Case: Θ(n) — even in best conditions, all elements are scanned once.
* Average Case: Θ(n) — each element contributes to one comparison and one addition.
* Worst Case: Θ(n) — every element is processed; no recursion or nested loops.
* Space Complexity: O(1) — only a few variables for running sums and indices.
* Recurrence Relation: T(n) = T(n−1) + O(1) ⇒ T(n) = O(n).

1. Implementation Review

The implementation follows a clear modular structure with KadaneAlgorithm.java as the core logic, supported by PerformanceTracker for empirical data collection. It correctly handles edge cases such as empty and all-negative arrays, uses in-place computation, and includes JUnit tests for validation.

Strengths:

* Clean, readable code with good variable naming.
* Handles edge cases (empty, single-element, all-negative) gracefully.
* Integrated metrics collection enables accurate empirical analysis.
* Possible improvements:
* Slightly reduce redundant tracker calls to minimize overhead.
* Aggregate metrics results across trials for more stable averages.

1. Empirical Results

The runtime data was collected for input sizes 100, 1000, 10,000, and 100,000 under various distributions (random, sorted, reverse, nearly-sorted). The plotted results confirm near-linear scaling, consistent with the theoretical O(n) time complexity.

Both axes are in logarithmic scale, clearly illustrating the linear trend (slope ≈ 1). Minor differences between distributions stem from varying early termination points and array traversal patterns, but remain within expected variance.

1. Conclusion

Kadane’s Algorithm demonstrates ideal linear time performance with minimal memory footprint. Empirical tests confirm the Θ(n) behavior across all datasets. Its simplicity, determinism, and in-place nature make it one of the most efficient subarray sum algorithms. Further optimizations are marginal, as the algorithm is already asymptotically optimal.