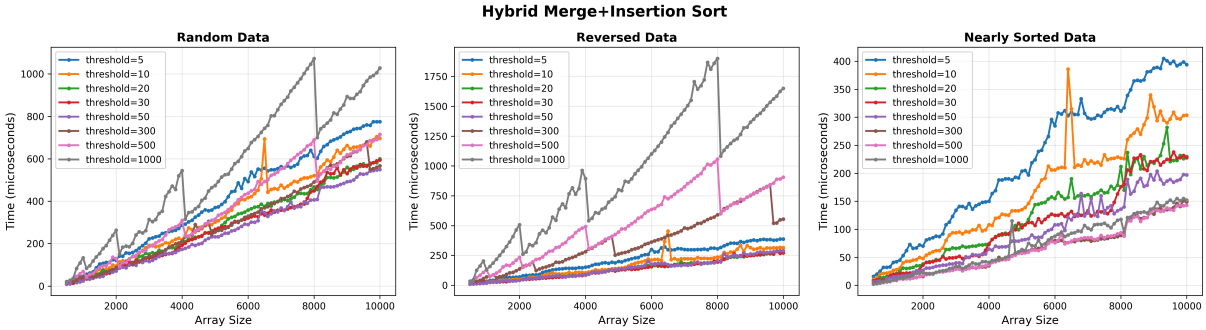
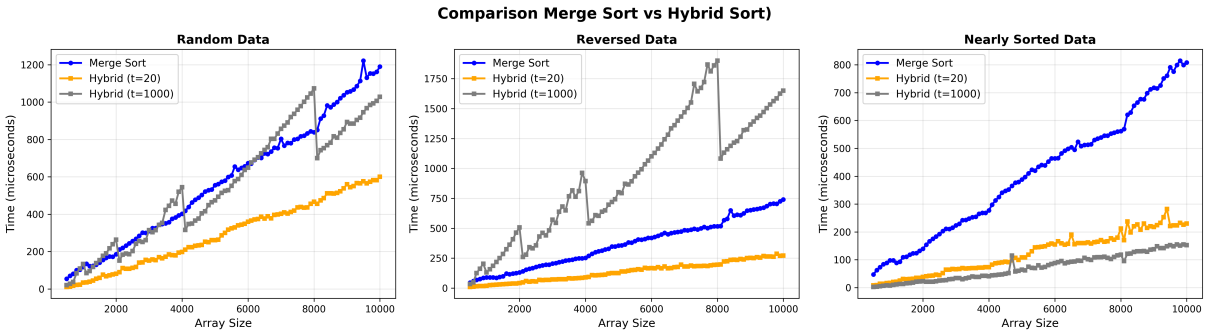
Solution A2

ID: 349242759 Репозиторий гитхаб



Видно, что по результатам запуска алгоритма **Merge+Insertion sort** лучшее время выполнения приходится в среднем на занчение **thresholds** $\in [20, 50]$, зависит это от константы, которая будет **у MergeSort**, для количества элементов равных **thresholds**. То есть для каких-то небольших n, $\Theta(c_1 \cdot n \cdot \log(n)) > \Theta(c_2 \cdot n^2)$.

Возьмем значение равное 20, для сравнения с обычным **Merge sort**.



С рандомными массивами гибрид работает примерно в два раза быстрее, это видно на левом графике. Чем больше массивы, тем больше разница, так как с увеличением количества элементов, увеличивается и попадаение в условие, что **количество элементов < thresholds**.

Данные отсортированные в обратном порядке будут сортироваться также быстрее, хоть и **insertion sort** для такого набора должно работать еще хуже.

Данные, почти отсортированные, **Hybrid sort** сортирует еще быстрее. Что и логично, ведь мы сделаем не так много сравнений в **insertion sort**, так как мы останавилваемся, как только **a[i] < a[i + 1]**.

Также видно, что при значениях n близких к 1000, сортировка начинает работать медленнее, чем обычный **Merge sort**. Особенно это заметно на данных, отсортированных в обратном порядке. Хотя при данных, где данные почти отсортированны, алгоритм показывает еще лучший результат. Так как опять же сравнений будет меньше, чем в **Merge sort**.

По итогу, можно сказать, что гибридный алгоритм будет более эффективным, чем обычный **Merge sort**, во всех случаях.