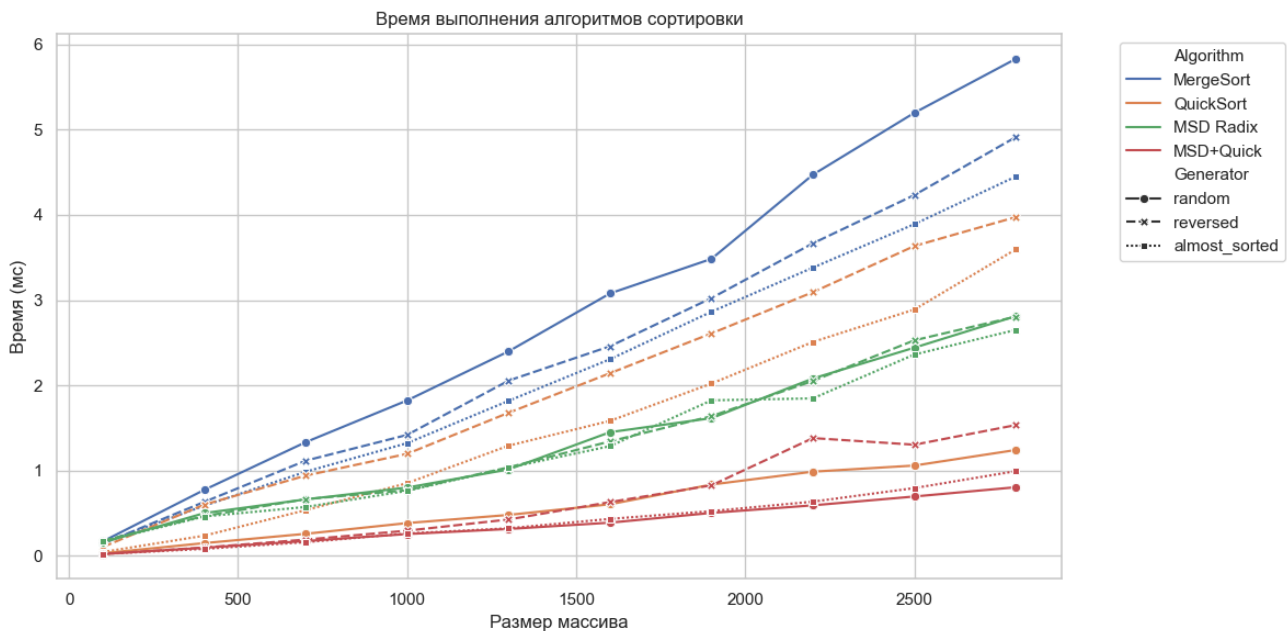


Block A

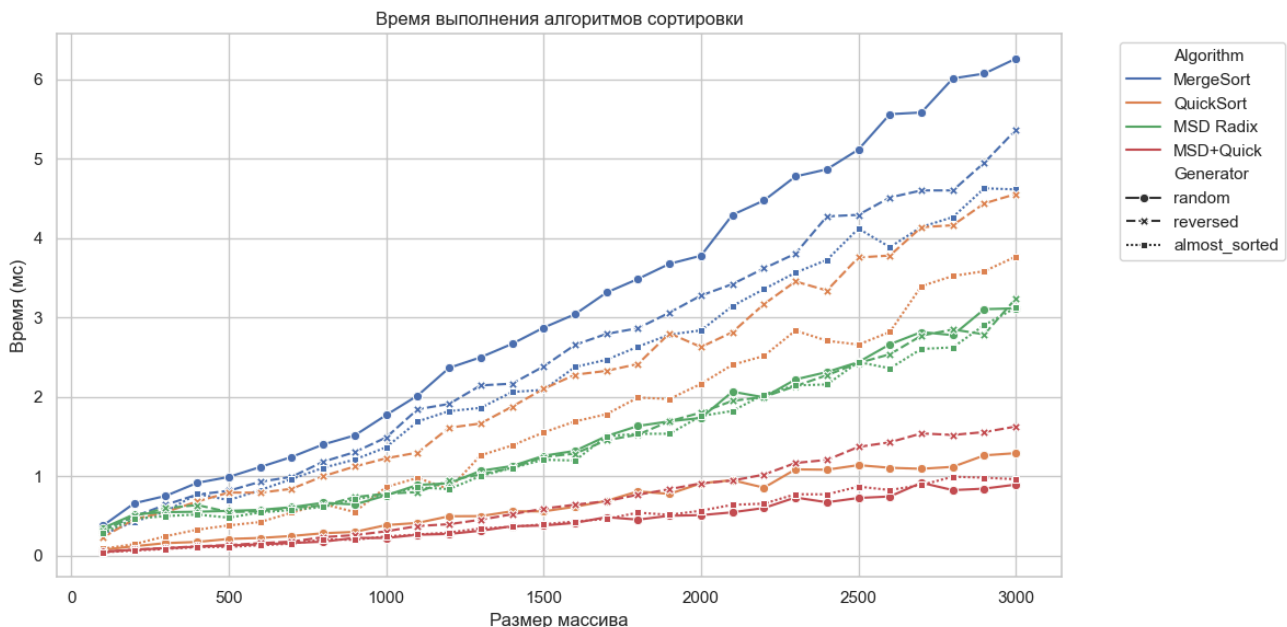
Время выполнения

На первом графике представлены зависимости времени выполнения различных алгоритмов сортировки строк от размера массива и характера входных данных.

Для шага 300



Для шага 100



Наблюдения:

- MSD+QuickSort уверенно показывает наименьшее время выполнения на всех типах входных данных.
- MSD Radix чуть медленнее, но также демонстрирует линейный рост времени, что соответствует его теоретической сложности $O(n * L)$, где L — средняя длина строки.

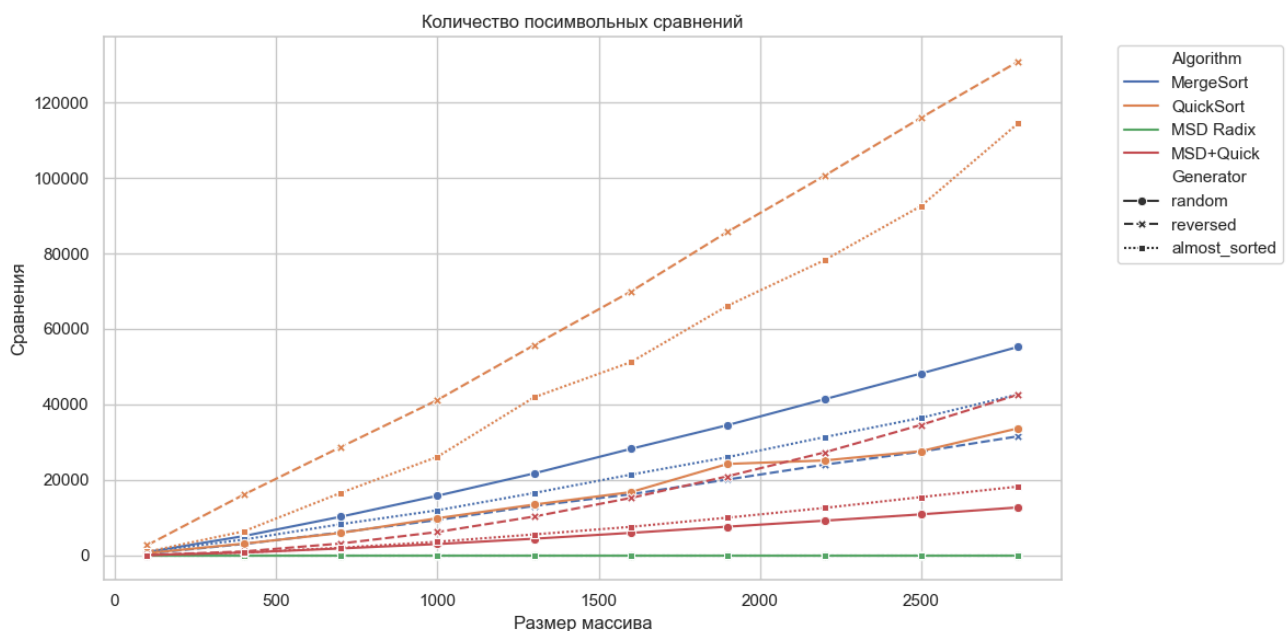
- MergeSort и QuickSort показывают худшие результаты по времени, особенно при reversed и almost_sorted входах. Это связано с необходимостью большого числа посимвольных сравнений.
- Разница между random, reversed и almost_sorted особенно заметна для QuickSort, что подчёркивает его чувствительность к входному порядку.

Вывод по времени:

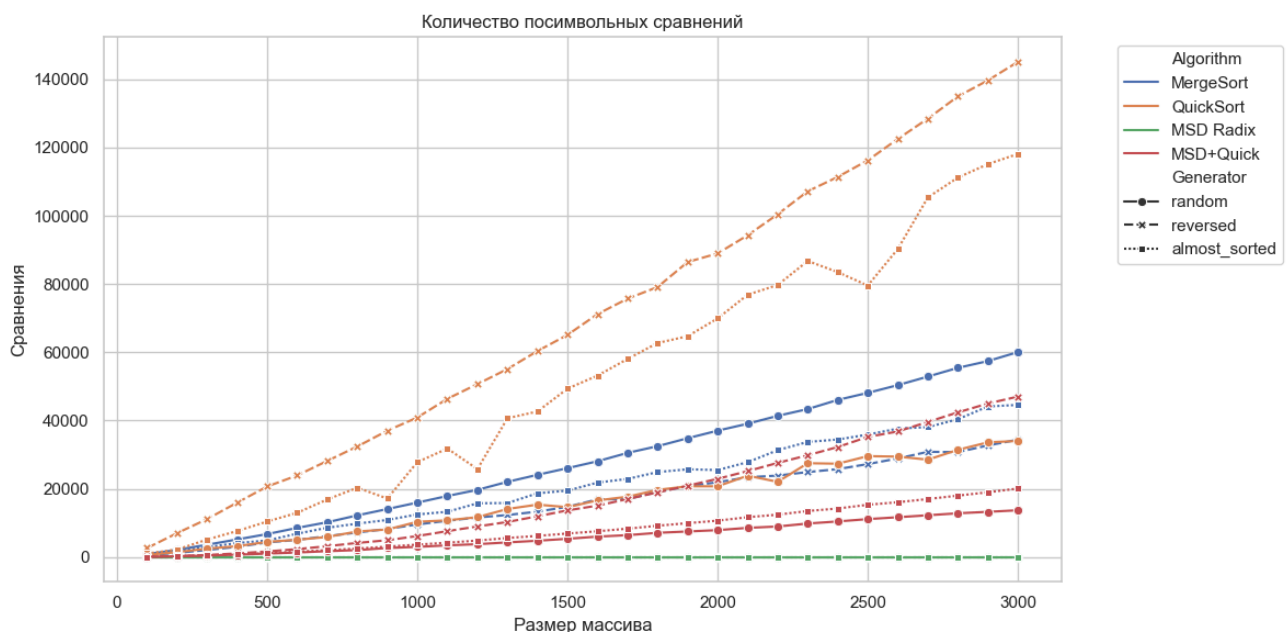
MSD+QuickSort — самый быстрый алгоритм на практике, особенно на почти отсортированных и случайных данных. Он сочетает в себе скорость MSD Radix и гибкость QuickSort для коротких фрагментов.

Количество посимвольных сравнений

На втором графике видно, как часто алгоритмы сравнивают символы строк.
Для шага 300



Для шага 100



Наблюдения:

- MSD Radix практически не выполняет посимвольных сравнений — за счёт разбиения по символам без непосредственного сравнения строк. Это подтверждает его эффективность при большом количестве однотипных префиксов.
- MSD+Quick сохраняет низкий уровень сравнений, немного уступая MSD Radix. Это связано с тем, что QuickSort применяется на малых подмассивах.
- QuickSort показывает максимальное количество сравнений, особенно на reversed данных — это подтверждает его худший случай $O(n^2)$, несмотря на тернарную оптимизацию.
- MergeSort более стабилен, но уступает MSD-подходам по эффективности сравнений.

Вывод по количеству посимвольных сравнений:

MSD Radix и MSD+Quick обеспечивают наилучшую экономию посимвольных сравнений, что особенно важно при работе с длинными строками и большим количеством совпадающих префиксов.

Выводы

Критерий	Лучший алгоритм
Время выполнения	MSD+QuickSort
Посимвольные сравнения	MSD Radix
Стабильность на входах	MSD+QuickSort
Практическое применение	MSD+QuickSort

Алгоритм MSD Radix с переключением на QuickSort (MSD+Quick) демонстрирует наилучший баланс между скоростью и экономией операций сравнения, что делает его оптимальным выбором для практической сортировки строк, особенно в условиях большого количества данных и разнообразия входных массивов.

Ссылки на послылки

id 321278551 [A1m](#)

id 321278635 [A1q](#)

id 321278855 [A1r](#)

id 321278946 [A1rq](#)

[github](#)