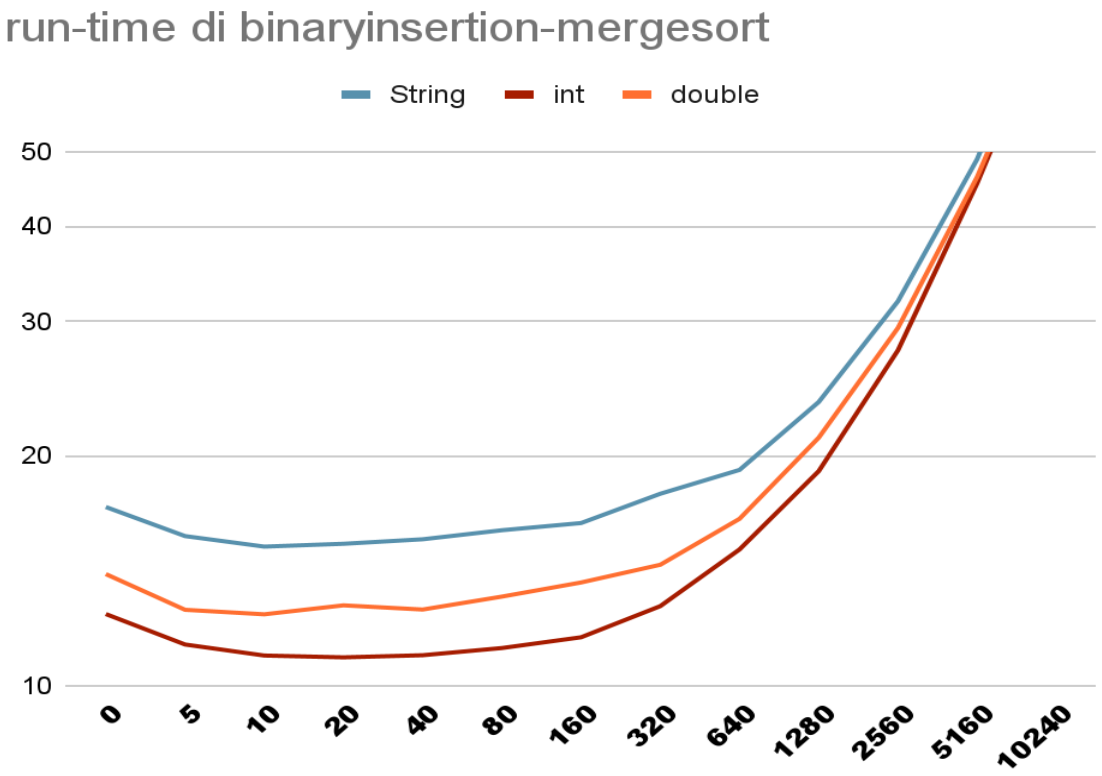
Analisi del costo temporale dell'algoritmo BINARYINSERTION-MERGESORT

L'algoritmo di ordinamento BINARYINSERTION-MERGESORT unisce due algoritmi di ordinamento (BinaryInsertionSort e MergeSort) che sceglie di usare a seconda della dimensione dell'input.
Se la dimensione eccede un certo valore k viene scelto il MergeSort(che richiama però ricorsivamente il BinaryInsertion-MergeSort), altrimenti viene scelto il BinaryInsertionSort.
L'algoritmo, scelto un k ottimo, è più efficiente del MergeSort poiché per insiemi piccoli l'algoritmo di BinaryInsertionSort è preferibile rispetto al MergeSort.
Scelto invece un k TROPPO grande, l'algoritmo diventa meno efficiente del MergeSort puro per insiemi grandi, e per k->array.size le prestazioni tendono verso quello del BinaryInsertionSort che ha complessità asintotica quadratica.



Nel grafico sono riportati i tempi di ordinamento dell'insieme di records, del file records.csv, al variare di k per i diversi campi.

Come previsto il tempo di esecuzione dell'algoritmo, al crescere di k, inizialmente diminuisce per poi aumentare notevolmente.

Per scegliere il k ottimale si può confrontare i tempi di esecuzione dei due algoritmi, MergeSort e BinaryInsertionSort, per insiemi di varie dimensioni, uno (il MergeSort) sarà più veloce per insiemi grandi e meno per insiemi piccoli e viceversa. Allora si può scegliere come k il punto d'intersezione delle due funzioni tempo.
Questo permette all'algoritmo di usare il BinaryInsertionSort quando è più efficiente del MergeSort e il MergeSort quando è più efficiente del BinaryInsertionSort.