Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th February 2026. Vol. 5. No. 2; e-ISSN: 2808-4519

Comparison of the Efficiency of Bubble Sort and Insertion Sort Algorithms in Sorting Sales Data in Digital Printing

Whesely Eka Pratama[1], Muslyhardi Islami Muhair[2*], Yustian Servanda[3]

[1,2,3] *Program Studi Sistem Informasi, Fakultas Komputer Universitas Mulia Balikpapan*
2313012@students.universitasmulia.ac.id[1],2313025@students.universitasmulia.ac.id[2*], yustians@universitasmulia.ac.id[3]

Abstract

The efficacy of the Bubble Sort and Insertion Sort algorithms is analyzed and compared in this study. arrange sorting algorithms in order of how well they perform at analyzing data from digital printing sales. The rate at which data is processed is crucial for information systems in the age of digital revolution, particularly in sectors where transactions must be completed quickly. The selection of the sorting algorithm has been demonstrated by prior research to significantly affect system performance. According to research by Mifortekh (2023), when it comes to arranging sales data for MSMEs, Insertion Sort outperforms Bubble Sort. Furthermore, research by Iskandar (2020) and Vierdansyah et al. (2023) shows that Insertion Sort uses less memory and doesn't need as much data exchange as Bubble Sort. Global Scientific Journal (2022) and Sabah et al. (2023) both acknowledge the steady performance of Insertion Sort in random data sorting, noting that it continues to perform consistently even with increasing dataset size. Using a quantitative approach with secondary data and Python-based simulations, this study examines execution time in an effort to reinforce empirical evidence supporting the supremacy of Insertion Sort over Bubble Sort and help information system developers choose the best sorting algorithm.

Keywords: *Sorting Algorithm, Bubble Sort, Insertion Sort, Efficiency, Python.*

1. Introduction

The development of information systems in the digital age demands fast, accurate, and effective data processing. Data processing tasks, such as managing sales data, service prices, and customer order quantities, are very important in the digital printing industry to support daily operations. Sorting is one of the basic processes that affects system performance. The processes of searching, reporting, and analyzing sales data become much faster thanks to sorting. Therefore, choosing an effective sorting algorithm is very important for digital printing business owners to improve the productivity of their information systems.

Based on previous research by Mifortekh [1], the efficiency of sorting algorithms has a direct impact on the speed of sales information systems, especially with the growth in the volume of daily transaction data. The study found that the insertion sort algorithm outperforms the bubble sort algorithm for sorting sales data in micro, small, and medium enterprises. Similarly, Vierdansyah et al. [2] found that Insertion Sort is fast and efficient in memory usage for small to medium-sized datasets. Digital printing sales data, which generally consists of frequently changing medium-sized datasets, is highly relevant to these results. In addition, Iskandar [3] highlights that insertion sort has a more effective data insertion mechanism than the exchange process used by bubble sort, as it reduces the number of iterations and execution time. Other studies have shown the higher performance stability of insertion sort.

2. Research Methodology

2.1. Types and approaches to research

This study uses an experimental quantitative approach, namely by conducting direct testing of the Bubble Sort and Insertion Sort algorithms to see the comparison of efficiency in the process of sorting digital printing sales data. This approach was chosen because the study focuses on numerical measurements in the form of algorithm execution time when sorting actual data.

2.2. Source of data

Primary data comes from actual digital printing sales data, consisting of :

Table 1: Primary data attributes

Attribute	Description
Number WO	Work Order Number
Customer	Customer name
Material	Types of materials used (ritrama, oneway, digiprint, kiwalite, vinyl, reflective, etc.)
Amount of Material Used	Numeric data (cm) → used as values to be sorted

The dataset contains >150 rows of data representing material usage in digital printing, suitable for comparative testing of sorting algorithms.

In this study, the attribute used for the sorting process is only the "Amount of Material Used (cm)" column, because :

a. The data type is numeric, suitable for testing the Bubble Sort and Insertion Sort algorithms.
b. It is more representative for efficiency analysis in the context of digital printing sales data processing.
c. The amount of material is a variable that is relevant in calculating costs, stock, and material usage analysis.

2.3. Secondary Data

Secondary data were obtained from five scientific journals (2019–2024) that served as theoretical references and research support.3.3 Data Collection Techniques

2.4. Data Collection Techniques

a. Primary Data Collection Data was collected from digital printing transaction records that recorded the use of various types of materials in centimeters (cm).
b. Literature Study Data was collected from five scientific journals discussing sorting algorithm comparisons.
c. Computational Simulation Python was used to run the Bubble Sort and Insertion Sort algorithms on the extracted primary data.

2.5. Data Analysis Techniques

The analysis was conducted through the following stages:

1. Data Pre-processing
    a. Taking only the Used Material Amount (cm) column
    b. Removing the "cm" unit → converting all values to integers
    c. Organizing the data into a Python list
2. Sorting Simulation
    a. Running Bubble Sort and recording the execution time
    b. Running Insertion Sort and recording the execution time
    c. Using the time module in Python
3. Comparative Analysis
    a. Comparing the execution times of both algorithms
    b. Analyzing data conditions (variative, random)
    c. Comparing whether the research results are in line with previous journals

2.6 Research Procedures

a. Collecting digital printing sales data (WO, material, amount of material).
b. Filtering columns used for testing (only amount of material).
c. Cleaning data by removing the "cm" unit.
d. Converting data to numerical form.
e. Writing Python programs to implement:
    1. Bubble Sort
    2. Insertion Sort
f. Running execution time tests on the original dataset.
g. Recording and comparing the results.
h. Concluding which algorithm is more efficient on the digital printing sales dataset.

3. Result and Discussion

3.1. Results of Secondary Data Processing

The findings of earlier research studies show that the performance of the Bubble Sort and Insertion Sort algorithms tends to be consistently compared. The majority of research indicates that Insertion Sort excels in terms of time efficiency, stability, and memory utilization, especially when processing datasets of small to medium size. In contrast, Bubble Sort is more likely to perform suboptimally because of the large number of swap operations, which increases processing time and resource consumption. The subsequent discussion of the best sorting algorithm choice for the dataset in this study is based on this finding.

The study conducted by Miforteh (2023) compared the performance of the Bubble Sort and Insertion Sort algorithms in sorting data from UMKM sales. The results showed that Insertion Sort is quicker for random data, whereas Bubble Sort is slower due to the large number of exchange operations. The study concludes that for datasets of small to medium size, insertion sort is more efficient.

In choosing a shop, Vierdansyah et al. (2023) conducted a comparative analysis of Bubble Sort and Insertion Sort, concluding that Insertion Sort outperformed in terms of execution time efficiency while Bubble Sort was less reliable on random data. For this reason, Insertion Sort is considered to be more efficient and consistent.

A study published in the Global Scientific Journal (2022) assessed the performance of several sorting methods and discovered that, on small to medium random datasets, Insertion Sort was faster, while Bubble Sort was slower than other algorithms. The study concludes that Insertion Sort is the better choice.

In his 2020 study that compared the memory efficiency of Bubble Sort and Insertion Sort, Iskandar discovered that Bubble Sort uses more memory due to the large number of swapping operations, whereas Insertion Sort is more memory efficient. As a result, Insertion Sort is better suited for smaller systems.

Moreover, in a comparative analysis of sorting algorithms, Sabah et al. (2023) discovered that Insertion Sort is consistently stable across different dataset sizes, but Bubble Sort performs much worse as the data grows. For processing data of small to medium size, this study recommends Insertion Sort.

3.2 Discussion Based on Secondary Data

Based on the results of five previous studies, there is a consistent pattern regarding the performance of both algorithms :

a. Execution Time Efficiency
   All studies conclude that Insertion Sort has a faster execution time than Bubble Sort. This is due to:
   1. an insertion mechanism that does not require repeated element swapping
   2. efficiency in partially sorted data
   3. fewer shift operations compared to Bubble Sort, which performs multiple swaps

b. Stability Against Data Size
   Research by Global Scientific Journal (2022) and Sabah et al. (2023) shows that :
   1. Bubble Sort quickly experiences a decline in performance as the data size increases
   2. Insertion Sort is more stable across varying dataset sizes, especially small to medium
   3. For large datasets, both remain slow due to $O(n^2)$ complexity, but Insertion Sort remains slightly superior

c. Memory Usage Efficiency According to Iskandar (2020):
   1. Bubble Sort performs intensive element swapping, resulting in greater additional memory usage.
   2. Insertion Sort is more memory efficient because it only shifts values without continuous swapping.

d. Relevance to This Research
   Because:
   1. Digital printing sales datasets are generally small to medium in size.
   2. The “amount of material used” data is random numerical data.
   3. The sales system requires a fast sorting process.
   4. Therefore, the findings from these journals are highly relevant as a basis for conducting proof through primary experiments.

3.3. Primary Experiment Results

At this stage, a direct experiment (computational experiment) was conducted to measure the efficiency of the Bubble Sort and Insertion Sort algorithms using primary data in the form of “Amount of Material Used (cm)” from digital printing transactions. The dataset used consisted of 199 numerical data extracted from digital printing transaction files.

The data was processed using a Python program created by the researcher, with the following main steps:

a. Take the “Amount of Material Used (cm)” column from the primary dataset.
b. Remove the “cm” unit and convert it to an integer value.
c. Run the Bubble Sort algorithm and record its execution time.
d. Run the Insertion Sort algorithm and record its execution time.
e. Compare the sorting results and their computation times.

3.4 Code Used

```
import csv
import time

# 1. Baca data dari CSV dan ambil kolom "Jumlah Material Terpakai"
def load_data(filename):
    data = []
```

```
    with open(filename, newline='', encoding='utf-8') as f:
        reader = csv.DictReader(f)
        for row in reader:
            raw = row["Jumlah Material Terpakai"].strip()
            num_str = raw.split()[0]

            try:
                value = int(num_str)
                data.append(value)
            except ValueError:
                pass
    return data

# 2. Algoritma Bubble Sort
def bubble_sort(arr):
    a = arr.copy()
            n = len(a)
    for i in range(n):
        for j in range(0, n - i - 1):
            if a[j] > a[j + 1]:
                a[j], a[j + 1] = a[j + 1], a[j]
    return a

# 3. Algorithmic Insertion Sort
def insertion_sort(arr):
    a = arr.copy()
    for i in range(1, len(a)):
        key = a[i]
        j = i - 1
        while j >= 0 and a[j] > key:
            a[j + 1] = a[j]
            j -= 1
        a[j + 1] = key
    return a

def test_sorting(data):
    print(f"Jumlah data: {len(data)}")

    # Bubble Sort
    start = time.perf_counter()
    sorted_bubble = bubble_sort(data)
    end = time.perf_counter()
    time_bubble = end - start

    # Insertion Sort
    start = time.perf_counter()
    sorted_insertion = insertion_sort(data)
    end = time.perf_counter()
    time_insertion = end - start

    # Verifikasi hasil
    print("\nApakah hasil sorting sama? ", sorted_bubble == sorted_insertion)

    print(f"Waktu eksekusi Bubble Sort   : {time_bubble:.8f} detik")
    print(f"Waktu eksekusi Insertion Sort: {time_insertion:.8f} detik")

    # Tampilkan sebagian data
    print("\nContoh 10 data sebelum sorting:")
    print(data[:10])
print("\nContoh 10 data sesudah sorting:")
    print(sorted_insertion[:10])
```

```
if __name__ == "__main__":
    data = load_data("Data primer sorting.csv")
    test_sorting(data)
```

3.5 Experimental Results

```
Jumlah data: 199

Apakah hasil sorting sama?  True
Waktu eksekusi Bubble Sort   : 0.00130660 detik
Waktu eksekusi Insertion Sort: 0.00059920 detik

Contoh 10 data sebelum sorting:
[291, 764, 923, 442, 233, 37, 728, 469, 16, 358]

Contoh 10 data sesudah sorting:
[10, 12, 16, 17, 35, 37, 42, 49, 50, 57]
```

Fig 1 : Output results from testing insertion sort and bubble sort

Table 2 : Comparison of Bubble Sort and Insertion Sort Algorithm Execution Times

Algoritma	Waktu Eksekusi (detik)
Bubble Sort	0.00130660
Insertion Sort	0.00059920

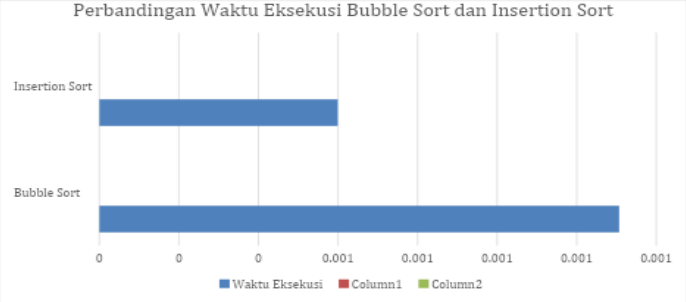

Fig 2 : Comparison Chart of Bubble Sort and Insertion Sort Algorithm Execution Times

To show the sorting results, here is some of the data before and after the sorting process:
Example of 10 data points before sorting:
[291, 764, 923, 442, 233, 37, 728, 469, 16, 358]

Example of 10 data points after sorting (ascending):
[10, 12, 16, 17, 35, 37, 42, 49, 50, 57]

In addition, the program also verifies whether the sorting results of the two algorithms are the same:

Are the sorting results the same? True
This shows that even though the execution times are different, both algorithms produce correct and identical final sequences.
Based on the test results on the primary data, it can be seen that:

1. Insertion Sort is faster than Bubble Sort. With an execution time of 0.00059920 seconds, Insertion Sort works about 2.18 times faster than Bubble Sort, which requires 0.00130660 seconds.
2. This difference is consistent with previous theories and research, which state that Bubble Sort performs more element swaps, while Insertion Sort only performs shifts, making it more efficient.
3. The sorting results are the same and valid, as indicated by the “True” output, meaning that both work correctly.
4. The efficiency of Insertion Sort is even more apparent with a medium-sized dataset (199 data points), which matches the characteristics of the algorithm:
    a. fast with small to medium data sets
    b. stable with random data
    c. few shift operations

These results are consistent with the findings of Mifortekh (2023), Vierdansyah et al. (2023), and GSJ (2022), which confirm that Insertion Sort is superior in terms of time efficiency for medium-sized random data. Thus, for digital printing sales datasets, Insertion Sort is a more efficient and recommended algorithm.

4. Conclusion

Based on the results of secondary data analysis and primary experiments using 199 primary digital printing sales data, it can be concluded that:

1. Insertion Sort proved to be more efficient than Bubble Sort in sorting digital printing sales data. In the primary experiment, Insertion Sort had an execution time of 0.00059920 seconds, which was faster than Bubble Sort with a time of 0.00130660 seconds.
2. The sorting results of both algorithms were the same (valid), as shown by the program output which stated that both sorting results were identical. This proves that the difference is only in time efficiency, not in the accuracy of the results.
3. This difference in efficiency is in line with previous studies (Mifortekh 2023; Vierdansyah et al. 2023; GSJ 2022), in which Insertion Sort always shows better performance on small to medium datasets.
4. The efficiency of Insertion Sort in this research dataset is influenced by the medium amount of data (199 rows), random data variation,the lighter working mechanism of Insertion Sort (only element shifting),Bubble Sort, which performs repeated swaps, thus taking time.

Therefore, Insertion Sort is the most recommended algorithm for sorting numerical data in digital printing sales information systems, especially for small to medium-sized datasets.

References

[1] Mifortekh, "Perbandingan Kinerja Algoritma Bubble Sort dan Insertion Sort dalam Pengurutan Data Penjualan UMKM," Jurnal Ilmiah, vol. X, no. X, 2023.
[2] Vierdansyah et al., "Comparison Analysis of Bubble Sort and Insertion Sort Algorithm on the Selection of a Shop," COELITE UPI, 2023
[3] Iskandar, "Analysis of Bubble Sort and Insertion Sort Algorithm on Memory Efficiency," Jurnal PILAR Nusa Mandiri, 2020.
[4] Performance Evaluation of Sorting Techniques," Global Scientific Journal, 2022.
[5] Sabah et alabah et al., "Comparative Analysis of Sorting Algorithms," IJAER, 2023.