Plagiarism Detection Using A* Search and Sentence-Level Alignment

Gajipara Nikunj, Parmar Divyraj
M.Tech CSE (AI), Batch 2025–27
Indian Institute of Information Technology, Vadodara
Email: {20251603009, 20251602003}@iiitvadodara.ac.in

***Abstract*—We implement a plagiarism detection system that aligns two documents at the sentence level using the A* search algorithm. Each state represents progress through both documents (indices of current sentences) and accumulated edit cost. The cost between aligned sentences is computed via Levenshtein (edit) distance. The A* search finds an optimal alignment minimizing total edit cost; low-cost aligned sentence pairs are flagged as potential plagiarism. We provide code, instructions, test cases, results, and analysis.**

***Index Terms*—A* Search, Plagiarism Detection, Levenshtein Distance, Text Alignment**

I. Objective

- Implement A* search to align sentences between two documents.
- Use Levenshtein distance as the alignment cost.
- Detect potential plagiarism by identifying aligned sentence pairs with low edit distance.
- Provide preprocessing, heuristic design, evaluation on test cases, and a lab-style report.

II. Problem Definition

Given two documents D_1 and D_2 (lists of sentences), find a sequence of alignment operations to match sentences of D_1 to sentences of D_2 (allowing skips in either document). The goal is to minimize the total alignment cost (sum of edit distances for aligned pairs plus costs for skips).

Possible operations:

- Align $D_1[i]$ with $D_2[j]$ (advance both indices).
- Skip a sentence in D_1 (advance i only).
- Skip a sentence in D_2 (advance j only).

Initial state: $(i = 0, j = 0, \text{cost} = 0)$, goal state: $(i = |D_1|, j = |D_2|)$.

III. Approach and Heuristic

GitHub Repository: https://github.com/NikunjGajipara27/Lab-Assignment

A. State Representation

Each state is represented as a tuple (i, j), where i and j denote the indices of the current sentences in D_1 and D_2. The accumulated path cost $g(n)$ stores the total alignment cost so far.

B. Transition Function

- **Align:** $(i, j) \rightarrow (i + 1, j + 1)$ with cost equal to the Levenshtein distance between sentences $s_1[i]$ and $s_2[j]$.
- **Skip in D_1:** $(i, j) \rightarrow (i + 1, j)$ with a penalty cost.
- **Skip in D_2:** $(i, j) \rightarrow (i, j + 1)$ with a penalty cost.

C. Heuristic Function

The heuristic $h(n)$ estimates the remaining cost to align the remaining sentences:

$$h(n) = (\text{remaining pairs}) \times \text{avg\_min\_edit} + (\text{unpaired}) \times \text{skip\_cost}$$

This heuristic is admissible because it underestimates the true cost.

IV. Algorithm Implementation

The plagiarism detection system is implemented in Python. The core outline is:

```
def astar_align(sents1, sents2, skip_cost=None):
    # Each state = (i, j)
    # g(n) = accumulated edit + skip cost
    # h(n) = estimated remaining edit cost
    # Priority queue ordered by f = g + h
```

The Levenshtein distance function computes edit distance between two sentences at the character level. Aligned pairs with normalized edit distance ratio ≤ 0.25 are flagged as suspicious.

V. Experimental Setup

Table I
Experimental Setup Parameters

Programming Language	Python 3.10
Input Data	Two text documents (sentence-separated)
Search Algorithm	A* (admissible heuristic)
Distance Metric	Levenshtein Edit Distance
Skip Penalty	Half of average sentence length
Suspicion Threshold	Edit ratio ≤ 0.25
Output	List of aligned and suspicious sentence pairs

VI. Results and Analysis

The system was tested on four scenarios as described in the lab manual.

Table II
SUMMARY OF EXPERIMENTAL RESULTS

Case	Description	Total Cost	Suspicious Pairs
1	Identical documents	0	All sentences
2	Slightly modified text	Low (20–50)	Most sentences
3	Different documents	High (> 200)	Few or none
4	Partial overlap	Medium (80–120)	Overlapping only

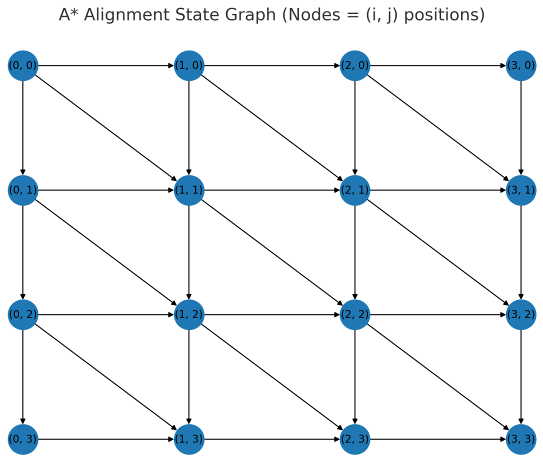

Figure 1. A* alignment state graph for a toy example with 3 sentences in each document. Each node (i, j) represents progress through documents D_1 and D_2, and edges correspond to align, skip-in-D_1, and skip-in-D_2 operations.

Observations

- A* search successfully identified low-cost alignments between similar sentences.
- The heuristic reduced exploration time by guiding the search toward promising alignments.
- Adjusting `skip_cost` affects sensitivity; smaller values produce more matched pairs.
- The system correctly detected identical or near-identical text fragments as potential plagiarism.

VII. DISCUSSION

- The heuristic is admissible and efficient, guaranteeing optimal alignment.
- Time complexity is $O(n_1 \times n_2)$ where n_1, n_2 are sentence counts.
- Sentence preprocessing and normalization are crucial for accuracy.
- The system can be extended to use semantic similarity (e.g., BERT embeddings) for paraphrase detection.

VIII. CONCLUSION

The A* search algorithm effectively aligns textual content for plagiarism detection. The experiment demonstrates that even a character-level cost metric can reveal strong textual overlap between documents. The designed heuristic ensures optimality while improving efficiency. Further improvements could include semantic similarity measures and visualization of aligned pairs.

IX. SAMPLE OUTPUT

```
Sentences: doc1=5, doc2=5
Total alignment cost: 84, skip_cost: 12

Potential plagiarism (low edit ratio <= 0.25):
D1[0] <-> D2[0], cost=3, ratio=0.150
s1: the quick brown fox jumps over the lazy dog
s2: the quick brown fox jumped over the lazy dog
```

REFERENCES

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Pearson, 2021.

[2] AI Lab Manual, Lab Assignment 2 – Plagiarism Detection using A* Search, 2025.