

Advanced Algorithms Weekplan 3

Lecture

This lecture is about string edit distance, alignment, and similarity problems.

Curriculum for week 3

“Algorithms on Strings, Trees, and Sequences” by Dan Gusfield.

- Chap. 11: All sections in the hand-out excluding 11.6.2, 11.8.2, and 11.8.3.
- Chap. 12: All sections in the hand-out excluding 12.1.5, and 12.2.

Exercises

Prepare solutions to the following exercises for the exercise session next week. Throughout the exercises assume that $S_1[1..n]$ and $S_2[1..m]$ are strings such that $n \geq m$.

1. Assign scores to the edit operations such that the maximum similarity between S_1 and S_2 is the length of the longest common subsequence of S_1 and S_2 .
2. How quickly can you compute the edit distance between S_1 and S_2 if you have n processors which can read and write in the table simultaneously? *Hint:* With n processors you can fill out n places in the table in a single step, but only if you already have computed the information needed.
3. **End-space Free Alignment** Consider a variant of similarity where spaces at the end of alignments are given a weight of 0. For example, in the alignment

```

- - c a c - d b d
l t c a b b d b -

```

the two spaces at the left end and the single space at the right end are free. How can we compute the variant of the problem efficiently?

4. Consider the following new edit operation:

Transpose Swap two adjacent characters in a string.

How can we compute the edit distance if we, in addition to the usual edit operations, include the transpose operation.

5. **Bounded differences** Consider the following problem: Given an integer $k \geq 0$, determine if there is an edit script from S_1 to S_2 using at most k insertions and deletions. Show how to solve this problem efficiently. The running time should depend on k .
6. The Four Russian algorithm in the text only computes the edit distance. Show how to modify it to produce the edit script.

Mandatory assignment

The mandatory hand-in assignment consist of the following two exercises. They should be handed in no later than Sep. 26 at the start of the lecture. You are encouraged to discuss the exercises with your classmates but you have to hand in your solutions individually.

1. Compute the edit distance between the strings `cocoa` and `acoa`. Write-up the edit distance matrix with pointers.
2. **Approximate String Matching** Consider the following problem: given strings $S_1[1..n]$ and $S_2[1..m]$ find a substring $S_1[k..j]$ minimizing the edit distance between $S_1[k..j]$ and $S_2[1..m]$. Give an efficient algorithm for this problem. *Hint:* Modify the dynamic programming recurrence for the string edit distance problem.