



Algorithms: Design  
and Analysis, Part II

# Introduction

---

Motivating Application:  
Sequence Alignment

# Motivation

**Sequence alignment:** Fundamental problem in computational genomics.

**Input:** Two strings over the alphabet  $\{A,C,G,T\}$ . [Portions of one or more genomes]

**Problem:** Figure out how similar the two strings are.

Example:

A	G	G	G	C	T
A	G	G	C	A	

**Example applications:**

- Extrapolate function of genome substrings.
- Similar genomes can reflect proximity in evolutionary tree.

# Measuring Similarity

**Question:** What does similar mean?

**Intuition:** AGGGCT, AGGCA are similar because they can be “nicely aligned”.

**Idea:** Measure similarity via quality of “best” alignment.

**Assumption:** Have experimentally determined *penalties* for gaps and the possible matches.

**Example:**

A	G	G	G	C	T
A	G	G	C	A	

**Alignment:**

A	G	G	G	C	T
A	G	G	-	C	A

insert one “gap”

one mismatch

# Problem Statement

**Input:** 2 strings over  $\{A,C,G,T\}$ .

- Penalty  $pen_{gap} \geq 0$  for each gap.
- Penalty  $pen_{AT} \geq 0$  for mismatching A and T.
- etc.

**Output:** Alignment of the strings that minimizes the total penalty  
 $\Rightarrow$  Called the Needleman-Wunsch score [1970].

Small NW score  $\approx$  Similar Strings

# Algorithms are Fundamental

**Note:** This measure of genome similarity would be useless without an efficient algorithm to find the best alignment.

**Brute-force search:** Try all possible alignments, remember the best one.

**Question:** Suppose each string has length 500. Roughly how many possible alignments are there?

- A) # of students in this class  $\approx 10^4 - 10^5$
- B) # of people on earth  $\approx 10^9 - 10^{10}$
- C) # of atoms in known universe  $\approx 10^{80}$
- D) More than any of the above  $\geq 2^{500} \geq 10^{125}$

**Point:** Need a fast, clever algorithm.

**Solution:** Straightforward dynamic programming.