Dynamic Programming: Longest Common Subsequence

Take two species: one with DNA: AGCCCTAAGGGCTACCTAGCTT and another with DNA GACAGCCTACAAGCGTTAGCTTG. Their DNA has a long common subsequence of AGCCTAAGCTTAGCTT.

BDFH is a **subsequence** of ABCDEFGH.

If X and Y are sequences, their **common subsequence** must be a subsequence of both. For example: BDFH is the longest common subsequence of ABCDEFGH and ABDFGHI.

The **longest common subsequence (LCS)** is a common subsequence which is the longest. For example, in the previous example the LCS is ABDFGH.

When solving a problem with **Dynamic Programming (DP)**:

- The first step is identifying the optimal substructure and finding overlapping subproblems.
- The second step is using recursive formulation to solve the problem.

Dynamic Programming: Knapsack Problem

One well-known optimization problem is the knapsack problem, where n items are packed in a knapsack with a maximum capacity weight. Each object has w weight as well as V value. The items must be packed so that the knapsack holds the maximum total value of items.

The **unbounded knapsack problem** is when an item can be packed more than once. **The 0/1 knapsack problem** is when there is just one copy of each item.