ABIN Assignment 4

Group 16

Q3

```
def Q3(sequence, states, start_probability, transition_probability, emission_probability):
    # Initialize matrices
    T = len(sequence)
    N = len(states)
    V = [{} for _ in range(T)]
    path = [{} for _ in range(T)]

    # Initialization step
    for state in states:
        V[0][state] = start_probability[state] * emission_probability[state][sequence[0]]
        path[0][state] = [state]

    # Recursion step
    for t in range(1, T):
        new_path = {}

        for state in states:
            (prob, prev_state) = max(
                (V[t - 1][prev_state] * transition_probability[prev_state][state] *
                 emission_probability[state][sequence[t]], prev_state)
                for prev_state in states
            )

            V[t][state] = prob
            new_path[state] = path[t - 1][prev_state] + [state]

        path[t] = new_path

    # Termination step
    (prob, state) = max((V[T - 1][final_state], final_state) for final_state in states)

    # Return the most likely sequence of states
    return path[T - 1][state]
```

Code Explanation:

The provided code defines a function `Q3` that implements the Viterbi algorithm for determining the most likely sequence of hidden states in a Hidden Markov Model (HMM) given a sequence of amino acids. The function takes several parameters, including the amino acid sequence, possible states, initial probabilities, transition probabilities between states, and emission probabilities for each state.

The algorithm progresses through several steps:

1. **Initialization (Matrices):**
    - Sets up matrices to store probabilities (`V`) and paths (`path`) for each state at each observation.
2. **Initialization Step:**
    - Computes the initial probabilities for each state based on the emission probabilities of the first amino acid in the sequence.
3. **Recursion Step:**
    - Iterates through the observations, calculating the probability of the best path to each state at the current observation.
    - Keeps track of the best path to each state.
4. **Termination Step:**
    - Identifies the state with the highest probability for the last observation, representing the most likely final state.
5. **Backtracking to Find the Most Likely Sequence:**
    - Traces back through the stored paths to determine the most likely sequence of states.
6. **Return the Result:**
    - Returns the computed most likely sequence of states for the entire observation sequence.

In summary, the function utilizes the Viterbi algorithm to predict the protein's secondary structure elements based on the observed amino acid sequence, incorporating probabilities of state transitions and emissions.

Input

```
sequence = "ACDE"
states = ['Alpha-helix', 'Beta-sheet', 'Coil']
start_probability = {'Alpha-helix': 0.6, 'Beta-sheet': 0.3, 'Coil': 0.1}
transition_probability = {
    'Alpha-helix': {'Alpha-helix': 0.5, 'Beta-sheet': 0.2, 'Coil': 0.3},
    'Beta-sheet': {'Alpha-helix': 0.3, 'Beta-sheet': 0.5, 'Coil': 0.2},
    'Coil': {'Alpha-helix': 0.2, 'Beta-sheet': 0.3, 'Coil': 0.5}
}
emission_probability = {
    'Alpha-helix': {'A': 0.2, 'C': 0.3, 'D': 0.3, 'E': 0.2},
    'Beta-sheet': {'A': 0.1, 'C': 0.4, 'D': 0.3, 'E': 0.2},
    'Coil': {'A': 0.3, 'C': 0.2, 'D': 0.2, 'E': 0.3}
}
```

Output

```
output = Q3(sequence, states, start_probability, transition_probability, emission_probability)
print("Most likely sequence of states:", output)
```

```
Most likely sequence of states: ['Alpha-helix', 'Alpha-helix', 'Alpha-helix', 'Alpha-helix']
```