In [13]:
```
import numpy as np

def initialize(states, observations, start_prob, trans_prob, emit_prob):
    num_states = len(states)
    num_observations = len(observations)

    # Initialize the Viterbi table with zeros
    viterbi_table = np.zeros((num_states, num_observations))
    # Initialize the path pointer table with zeros
    path_pointer = np.zeros((num_states, num_observations), dtype=int)

    # Initialize the first column of the Viterbi table
    for s, state in enumerate(states):
        # Use state as a key to access the dictionaries
        viterbi_table[s, 0] = start_prob[state] * emit_prob[state][observations[0]]

    return viterbi_table, path_pointer

def forward(viterbi_table, path_pointer, states, observations, trans_prob, emit_prob):
    num_states = len(states)
    num_observations = len(observations)

    for t in range(1, num_observations):
        for s, state in enumerate(states):
            # Compute the max probability for the current state and observation
            (max_prob, max_state_index) = max(
                (viterbi_table[prev_state_index, t-1] * trans_prob[states[prev_state_i
                for prev_state_index in range(num_states))

            viterbi_table[s, t] = max_prob
            path_pointer[s, t] = max_state_index

def backtrack(viterbi_table, path_pointer, states, observations):
    num_states = len(states)
    num_observations = len(observations)

    # Initialize the path with the maximum probable last state
    path = np.zeros(num_observations, dtype=int)
    path[num_observations-1] = np.argmax(viterbi_table[:, num_observations-1])

    # Backtrack through the path pointer table
    for t in range(num_observations-2, -1, -1):
        path[t] = path_pointer[path[t+1], t+1]

    return [states[state] for state in path]

states = ['Noun', 'Verb']
observations = ['They', 'can', 'fish']
start_probability = {'Noun': 0.37, 'Verb': 0.14}
transition_probability = {
   'Noun' : {'Noun': 0.05, 'Verb': 0.37},
   'Verb' : {'Noun': 0.37, 'Verb': 0.05},
   }
emission_probability = {
   'Noun' : {'They': 0.14, 'can': 0.05, 'fish': 0.05},
   'Verb' : {'They': 0.00004, 'can': 0.37, 'fish': 0.05},
}

viterbi_table, path_pointer = initialize(states, observations, start_probability, trar
forward(viterbi_table, path_pointer, states, observations, transition_probability, emi
most_probable_path = backtrack(viterbi_table, path_pointer,states,observations)
```

In [14]:
```
most_probable_path
```

Out[14]:
```
['Noun', 'Verb', 'Noun']
```

In [18]:
```
np.log(viterbi_table)
```

Out[18]:
```
array([[ -2.96036513,  -8.95182968,  -8.93885422],
       [-12.09274396,  -4.94886968, -10.94033422]])
```

In [19]:
```
np.round(np.log(viterbi_table))
```

Out[19]:
```
array([[ -3.,  -9.,  -9.],
       [-12.,  -5., -11.]])
```