Algorithms Course Project

Premier League Standing

By:

Ahmed Nashaat

Kareim Alaa

Mohamed Sherif

Omar Mostafa

Omar Shaaban

2022-2032

Complexity Analysis

1.Time Complexity

1. __init__() method: This method initializes two attributes, (num_teams) and (adj_list), which takes constant time, which is O(1).
2. add_edge() method: This method inserts a new edge into the adjacency list for a given team. Therefore, the time complexity of this method is O(n), where n is the number of matches played by the team up to the current point in time.
3. graph_initialization() method: This method initializes the graph data structure by adding all the edges for each match played in the EPL. The time complexity of this method depends on the number of matches played in the EPL, which is at most equal to the product of the number of teams and the number of rounds played in the league. Therefore, the time complexity of this method is O(n^2), where n is the number of teams in the EPL.
4. calc_standings() method: This method updates the standings of two teams after a match has been played. The method involves updating eight elements of the standings list, which takes constant time. Therefore, the time complexity of this method is O(1).
5. BFS_round() method: This method performs a breadth-first search traversal of the graph data structure to calculate the standings of all the teams up to a given round number. Therefore, the time complexity of this method is O(n^2), where n is the number of teams in the EPL.

6. BFS_date() method: This method performs a breadth-first search traversal of the graph data structure to calculate the standings of all the teams up to a given date. The time complexity of this method depends on the total number of matches played up to the given date. Therefore, the time complexity of this method is O(n^2), where n is the number of teams in the EPL.
7. print_standing() function: This function takes a dictionary of standings as input and sorts the standings based on the total number of points. Therefore, the time complexity of this function is O(n log n), where n is the number of teams in the EPL.
8. solve() function: This function reads user input and calls either the BFS_round() or BFS_date() method based on the input. The time complexity of this function depends on the user input and the method called. Since both methods have a time complexity of O(n^2), the time complexity of this function is also O(n^2).

Overall, the time complexity of the program is dominated by the graph_initialization() method, which has a time complexity of O(n^2), where n is the number of teams in the EPL. The other methods and functions have a time complexity that is either constant or proportional to the number of matches played or the number of teams in the league, which is much smaller than n^2. **Therefore, the overall time complexity of the program is O(n^2).**

2.Space Complexity

1. Input data: The input data is stored in a Pandas DataFrame object. The space complexity of the DataFrame object depends on the number of rows and columns in the DataFrame, which is proportional to the number of matches played in the EPL. Therefore, the space complexity of the input data is O(n), where n is the number of matches played in the EPL.
2. Graph data structure: The graph data structure is represented using a dictionary of adjacency lists. The space complexity of the dictionary object depends on the number of keys in the dictionary, which is equal to the number of teams in the EPL. Each adjacency list contains information about the matches played by the corresponding team, which is proportional to the number of matches played by the team up to the current point in time. Therefore, the space complexity of each adjacency list is O(n), where n is the number of matches played by the team up to the current point in time. Therefore, the overall space complexity of the graph data structure is O(n^2), where n is the number of teams in the EPL.
3. The program also uses a deque data structure to implement the BFS traversal of the graph. The space complexity of the deque object depends on the number of elements in the deque, which is at most equal to the number of teams in the EPL. Therefore,the space complexity of the deque object is O(n), where n is the number of teams in the EPL.

4. Standings dictionary: The program uses a dictionary to store the standings of the teams. The space complexity of the dictionary object depends on the number of keys in the dictionary, which is equal to the number of teams in the EPL. Each value in the dictionary is a list of length 8, which contains information about the team's standings. Therefore, the space complexity of each value in the dictionary is O(1). Therefore, the overall space complexity of the standings dictionary is O(n), where n is the number of teams in the EPL.

Therefore, the overall space complexity of the program is O(n^2), where n is the number of teams in the EPL. This is dominated by the space complexity of the graph data structure, which is O(n^2). The space complexity of the input data and the standings dictionary is smaller and proportional to the number of matches played and the number of teams in the EPL, respectively, which is much smaller than n^2. The space complexity of the deque object and other variables is constant.