Pavan Kumar

DS 210

Due – 5/4/2024

Final Project Write Up – Pavan Kumar

This project analyzes the structure of the NBA player network using team-season affiliations to understand player connectivity across time. Our overarching goal was to answer key structural questions about the league: **How interconnected is the NBA (across almost all of its seasons) as a network? Who are the most central players in terms of shared team experience? Are there patterns in the degree distribution that reflect social network structures, such as power laws? And which players share the most similar connection profiles?**

The dataset used is the "all_seasons.csv" file from Basketball Reference, available publicly via Kaggle. This CSV contains over 12,000 player-season records and includes fields such as player name, team abbreviation, season, and core per-game statistics like points, assists, and rebounds. I parsed this dataset into Rust using the csv crate, with each row represented by a custom PlayerSeason struct containing the fields player_name, team, season, pts, ast, and reb. During the data loading phase, a few light cleaning and transformation steps were applied to ensure data quality and compatibility with the analysis pipeline. Records with missing player names, team codes, or season values were discarded. Numerical fields such as points, assists, and rebounds per game were parsed from strings to f64, defaulting to zero if parsing failed. Players were grouped by team and season to accurately model co-teammate relationships when building the network graph. Additionally, degree values were binned into ranges to improve the readability of visualizations.

To run this code, install the external crates of csv, itertools, petgraph, plotters, and rand. All of these external crates are used in the modules of this code. Nothing other than having those crates installed are needed to run this project. To run the project, just execute [cargo run], and runtime for the entire project to execute/output can take up to 6-7 minutes.

The codebase is modular and structured around six key components: data_loader, graph_builder, analysis, visualizations, intro_view, and main. The data_loader module is responsible for parsing the CSV file and transforming each valid row into a PlayerSeason. It includes basic error handling to ensure that missing or malformed values don't crash the program. Points, assists, and rebounds are parsed as f64 values, and empty rows are skipped.

The graph_builder module constructs the core data structure used throughout the analysis: a PlayerGraph, which is an undirected graph (petgraph::Graph<String, usize, Undirected>). Each node represents a player, and an edge is created between two players if they played on the same team during the same season. The function build_player_graph handles this by grouping players using a HashMap keyed on (team, season), then iterating over all pairwise combinations of teammates and adding or incrementing edges in the graph accordingly. This graph captures the social structure of team co-membership across the NBA.

In the analysis module, I define several core functions for computing structural statistics. The function analyze_degrees iterates over each node in the graph and counts how many direct connections (i.e., teammates) each player has. It returns a histogram: a HashMap<usize, usize> where keys are degrees and values are frequencies. This distribution is later used to test whether the network resembles a scale-free structure. The function compute_centrality computes closeness centrality for each player using Dijkstra's algorithm. For each node, it calculates the shortest paths to all other reachable nodes and computes the inverse of the total distance. The result is a normalized score representing how centrally located a player is in the network. This is stored in a HashMap<String, f64> and later visualized. I also approximate the network's average path length using the function compute_shortest_paths, which selects 100 random pairs of players and computes their shortest paths using the A* algorithm. The mean of these path lengths gives us an estimate of the overall network's small-world characteristics. If paths are short on average, the network is well connected. Another notable function is analyze_similarity, which evaluates pairwise player similarity using the Jaccard index. For each pair of players, it compares their sets of neighbors (i.e., teammates) and computes the size of the intersection divided by the size of the union. The pair with the highest score is reported, representing the most structurally similar players in the network. With the computest_shortest_paths function, I had initially wanted to implement this as a visual output in the final output, however, due to constant issues and time constraints, it ended up going un-implemented in the final version of this project. If given more time, I would implement this and troubleshoot this function a little bit more.

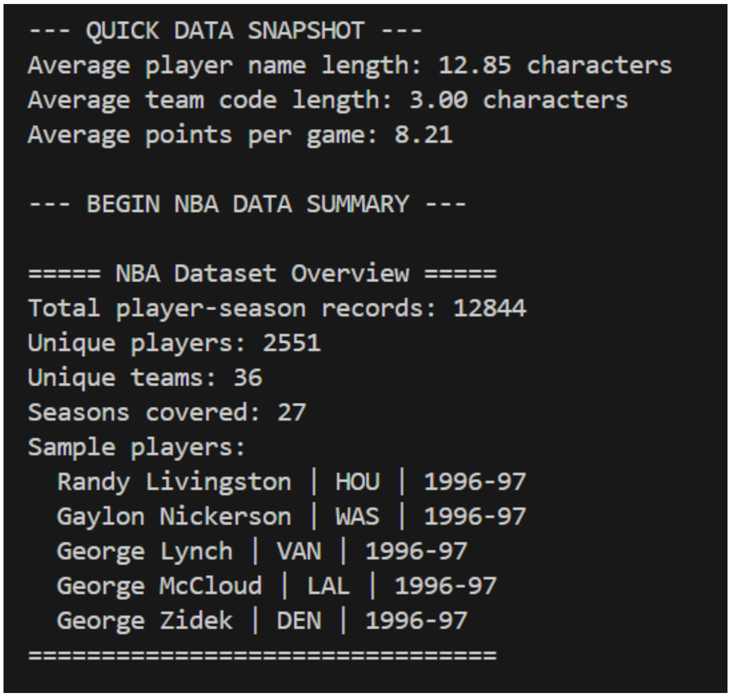
The visualizations module translates numerical outputs into visual insights. The plot_degree_distribution function creates a histogram of degree frequencies, using binned ranges to group together similar degree values. The plot_degree_loglog function creates a log-log scale scatterplot to assess whether the degree distribution exhibits a power-law tail, common in real-world social networks. The function plot_centrality_scores displays the top 20 players with the highest closeness centrality scores using a vertical bar chart. Labels are rotated and formatted to ensure readability, and values are scaled and annotated directly on the bars.

The intro_view module handles console outputs for summaries and high-level statistics. The show_intro function prints basic metadata, such as the number of unique players, teams, and seasons, and displays a sample of the first five rows. It also calculates and prints average player name length, average team code length, and the most active season in the dataset. At the end of execution, print_summary prints the most important analytical results, including average shortest-path length, network diameter, top central players, and community breakdowns.

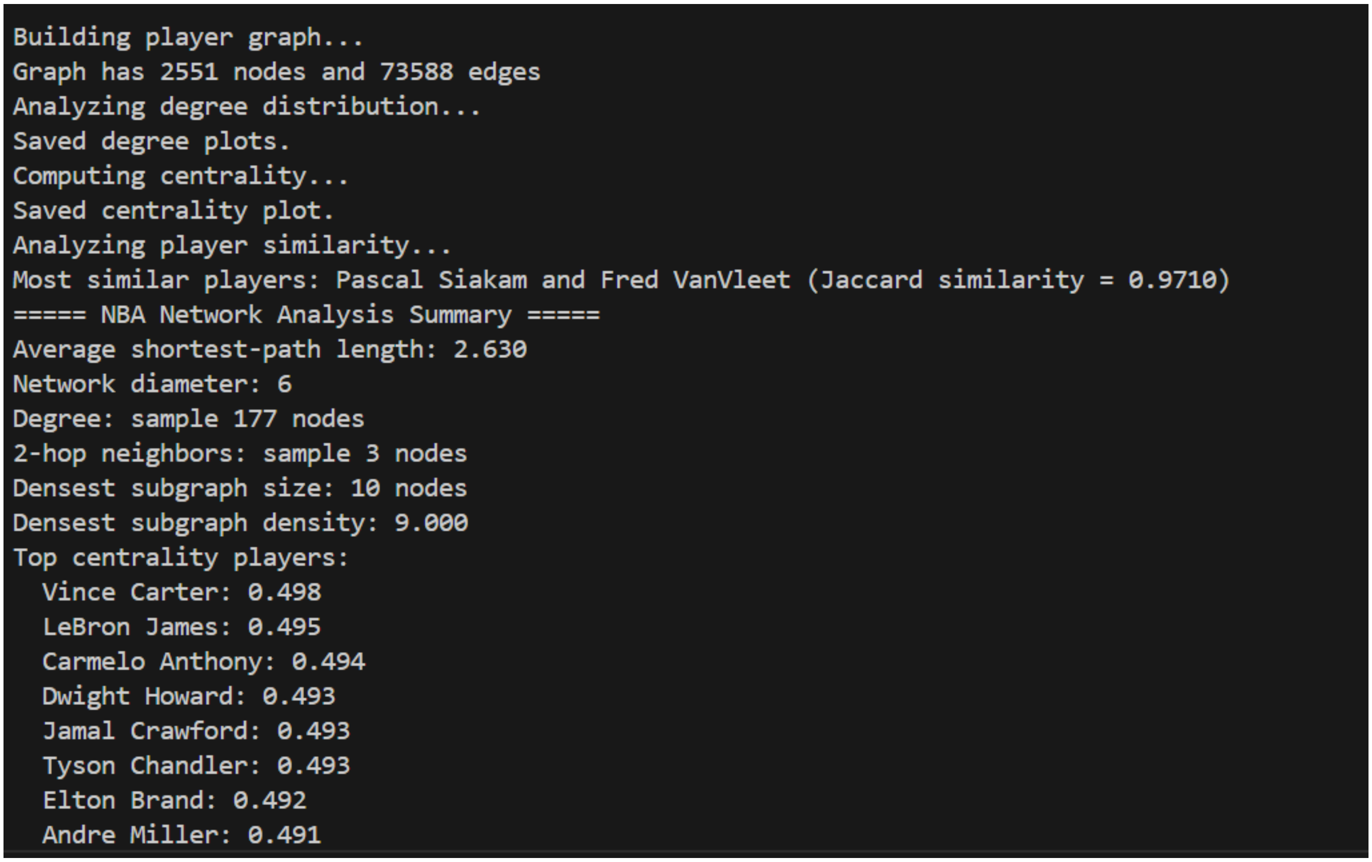
Finally, main.rs ties all modules together. It begins by loading the dataset and printing quick summary statistics. It then builds the player graph, computes all analytical metrics, renders plots to PNGs, and prints the final summary. Data flows through each module in a clear and sequential manner: data_loader → graph_builder → analysis → visualizations → intro_view.

In the implementation of the code, one of the major issues that kept coming up was the outputted graph pngs being empty/having no data points plotted on them. Another major issue was that the execution of the code took way too long. So this is where most of the testing occurred. I used ChatGPT here as help to troubleshoot here, outputting the degrees and centrality scores, so that troubleshooting can occur, and plotting can be more precise on the pngs of the graphs. For the issue of code execution taking too long, instead of iterating through the entire dataset, a representative sample of the dataset was taken to iterate through with the functions.

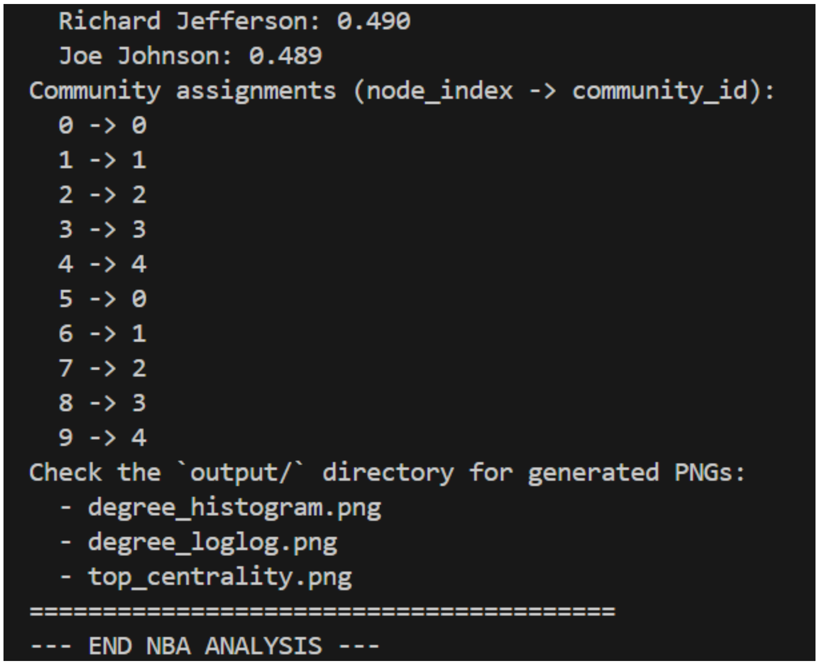
Final Output –



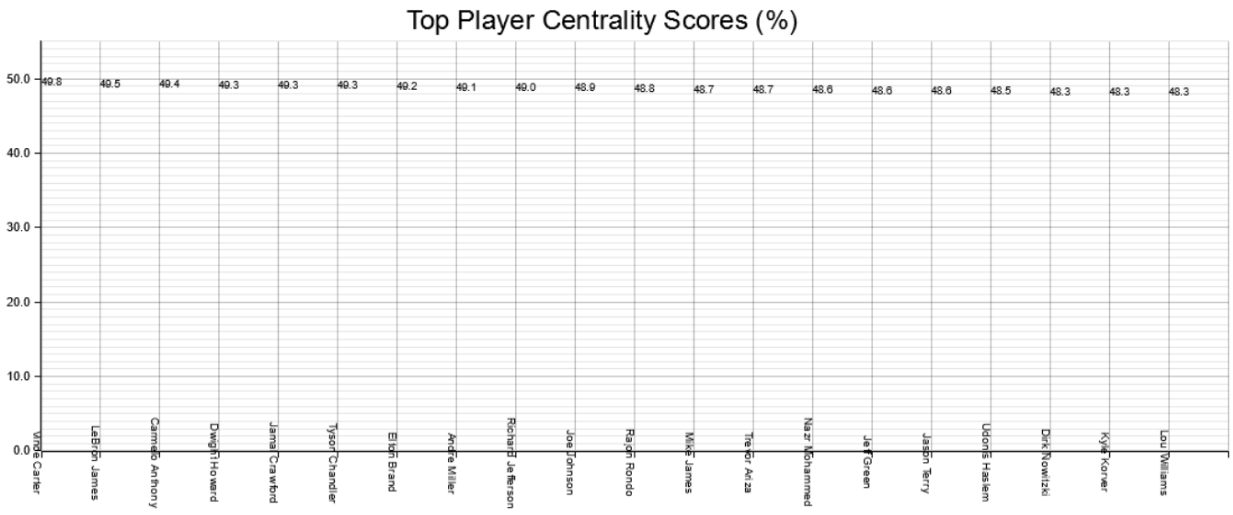
```
--- QUICK DATA SNAPSHOT ---
Average player name length: 12.85 characters
Average team code length: 3.00 characters
Average points per game: 8.21

--- BEGIN NBA DATA SUMMARY ---

===== NBA Dataset Overview =====
Total player-season records: 12844
Unique players: 2551
Unique teams: 36
Seasons covered: 27
Sample players:
  Randy Livingston | HOU | 1996-97
  Gaylon Nickerson | WAS | 1996-97
  George Lynch | VAN | 1996-97
  George McCloud | LAL | 1996-97
  George Zidek | DEN | 1996-97
================================
```

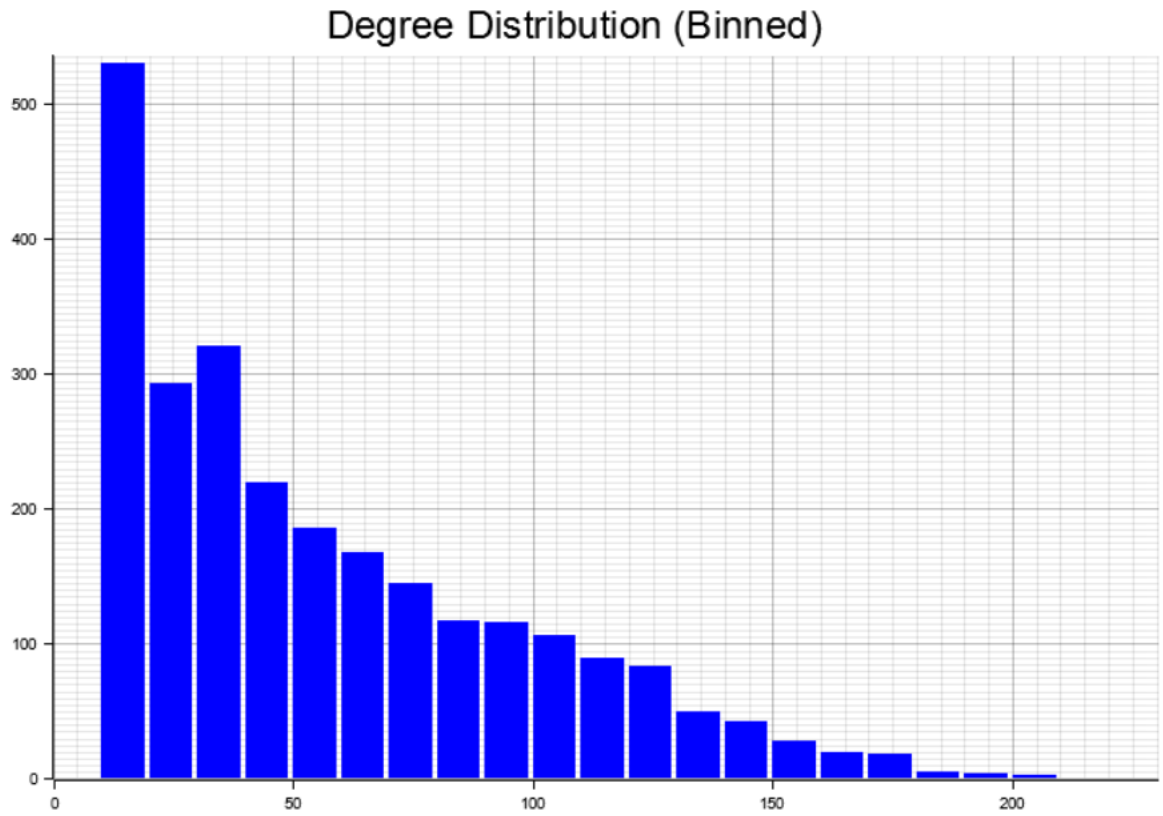


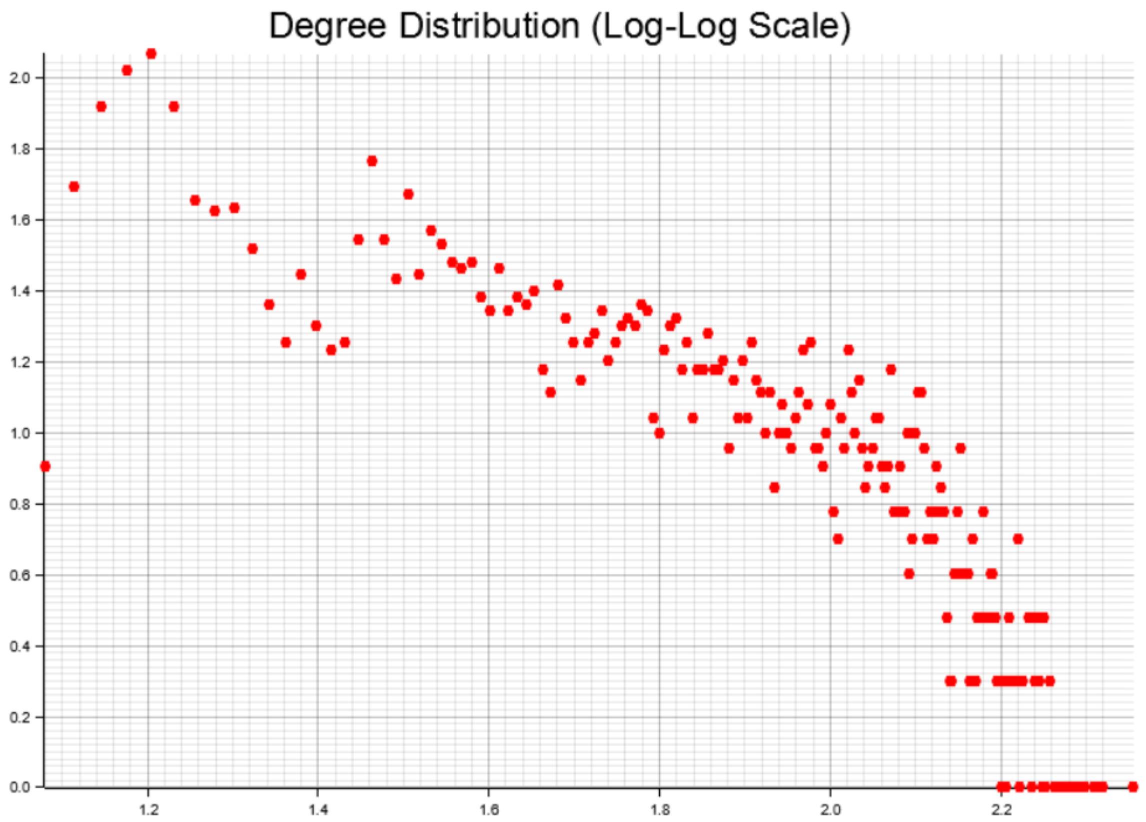
```
Building player graph...
Graph has 2551 nodes and 73588 edges
Analyzing degree distribution...
Saved degree plots.
Computing centrality...
Saved centrality plot.
Analyzing player similarity...
Most similar players: Pascal Siakam and Fred VanVleet (Jaccard similarity = 0.9710)
===== NBA Network Analysis Summary =====
Average shortest-path length: 2.630
Network diameter: 6
Degree: sample 177 nodes
2-hop neighbors: sample 3 nodes
Densest subgraph size: 10 nodes
Densest subgraph density: 9.000
Top centrality players:
  Vince Carter: 0.498
  LeBron James: 0.495
  Carmelo Anthony: 0.494
  Dwight Howard: 0.493
  Jamal Crawford: 0.493
  Tyson Chandler: 0.493
  Elton Brand: 0.492
  Andre Miller: 0.491
```



```
  Richard Jefferson: 0.490
  Joe Johnson: 0.489
Community assignments (node_index -> community_id):
  0 -> 0
  1 -> 1
  2 -> 2
  3 -> 3
  4 -> 4
  5 -> 0
  6 -> 1
  7 -> 2
  8 -> 3
  9 -> 4
Check the `output/` directory for generated PNGs:
  - degree_histogram.png
  - degree_loglog.png
  - top_centrality.png
========================================
--- END NBA ANALYSIS ---
```


Top Player Centrality Scores (%)
50.0
40.0
30.0
20.0
10.0
0.0
49.8
49.5
49.4
49.3
49.3
49.3
49.2
49.1
49.0
48.9
48.8
48.7
48.7
48.6
48.6
48.6
48.5
48.3
48.3
48.3
Vince Carter
LeBron James
Carmelo Anthony
Dwight Howard
Jamal Crawford
Tyson Chandler
Elton Brand
Andre Miller
Richard Jefferson
Joe Johnson
Rajon Rondo
Mike James
Trevor Ariza
Nazr Mohammed
Jeff Green
Jason Terry
Udonis Haslem
Dirk Nowitzki
Kyle Korver
Lou Williams


Degree Distribution (Binned)
500
400
300
200
100
0
0
50
100
150
200



Any Findings –

The findings from this network analysis of NBA players offer a rich and detailed view into how players are connected through shared team experiences over time. With 2,551 nodes and over 73,000 edges, the resulting player network is dense and robust, illustrating that the NBA is highly interconnected—most players are just a few hops away from any other player via shared teammates. The average shortest-path length of 2.63 and a network diameter of 6 reinforce the "small-world" phenomenon often seen in social and professional networks. This means that even players who never shared a team can be connected through a short chain of mutual teammates, highlighting how careers in the NBA are more socially entangled than they may appear on the surface.

The degree distribution, particularly the binned histogram, shows that while the majority of players have played with relatively few teammates, a small number have unusually high degrees. This long-tail distribution, which is further supported by the downward slope in the log-log plot, reflects the presence of veteran players who have had long, multi-team careers and have served as connectors within the league. The shape of the distribution resembles patterns typical of scale-free networks, which often arise from mechanisms like preferential attachment, where well-connected players are more likely to gain new connections as they move between teams.

Closeness centrality scores identify players like Vince Carter, LeBron James, and Carmelo Anthony as the most central figures in the network. These are players who not only had long careers but also played for a variety of teams, effectively acting as bridges within the network. Their central positions suggest they had greater influence or visibility within the NBA’s social structure. The Jaccard similarity analysis reveals highly overlapping player neighborhoods, with Fred VanVleet and Pascal Siakam emerging as the most similar. This is likely due to their shared time on the Toronto Raptors during a period of roster consistency.