HoopMatchups

Lawrence Rogers
Stanford University
Lawrence.X.Rogers@Gmail.com

Tony Liu
Stanford University
tdliu@stanford.edu

ABSTRACT

HoopMatchups is an online tool that visualizes real-time data about National Basketball Association (NBA) games. In this paper we introduce the growing interest of analytics in the NBA and the challenge of real-time lineup analysis. We survey existing work which have various drawbacks mainly that they fail to associate aggregated lineup statistics with actual portions of real games, whether live or offline. We describe the methods used to collect data from various sources and surface it in a real-time D3-based webapp. HoopMatchups has a lot of potential, much of which has been reached, but more still has not been tapped. Improvements other than bug-fixes and minor UI tweaks include more visually interesting on-court displays and richer lineup-segment information. Nonetheless, HoopMatchups is a unique, powerful, and useful tool for analyzing NBA games from fan, media, and team perspectives.

Author Keywords

Basketball, Analytics, D3, Real Time, Matchups, Lineups, Data Visualization

INTRODUCTION

At the very basic level, basketball analytics means using numbers and statistics to better understand the spot. Points scored, rebounds, steals, are terms that are very common and represent the most basic form, however, as the field has grown, more advanced statistics have become more and more popular. It has become increasingly popular in three main realms: Fans, media, and teams. Fans use basketball analytics to better understand the sport from an amateur perspective. Media, similarly, use analytics to explain to fans about the game, give theories on the success of some players, and make predictions about future games. Teams, to varying degrees, use analytics to help prepare their teams, make improvements to players, and even select players in the draft.

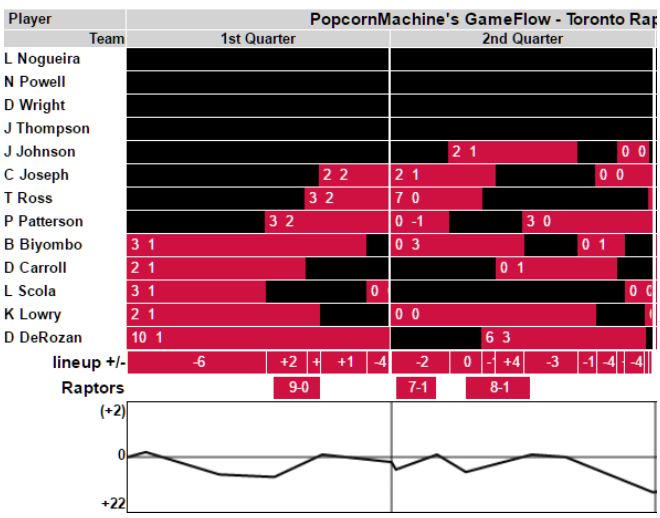
Visualization insofar as making complex analysis intuitive and digestible is crucial for all three realms. Existing work will be explored in more detail in the following section, however, in general there are two missing pieces to basketball analytics and visualization. The first is real-time visualization, and the second is a focus on lineups. What are lineups? A basketball team has around 10-15 players, but only 5 can play at a time. Throughout the game, the coach selects which players will be in the game via a starting lineup and substitutions. Although changing lineups are an obvious and large factor in basketball analysis, the implementation of real-time lineup based visualization is very difficult, as explored by HoopMatchups.

The main challenges include association of real-time data with existing season data, efficiently surfacing that data through a web interface, and visualizing the information in both a useful and elegant way.

RELATED WORK

The related work for this project consists mainly of two other line-up related data visualizations: PopcornMachine's Gameflow, and NBA Wowy.

PopcornMachine's Gameflow

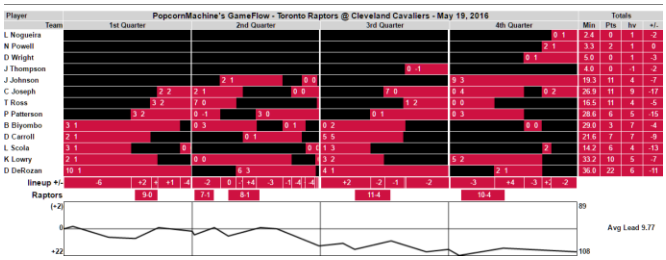


PopcornMachine's Gameflow represents the players on the court using red blocks. Time is on the x-axis and the score differential between the two teams is shown as a line along the bottom. The most effective parts of Gameflow are being able to instantly see the contribution of a player to the game, visually, as well as numerically with important statistics being shown on the red bars. Their contribution to the game via their lineup is also readily available as the lineup plus-minus score at the bottom.

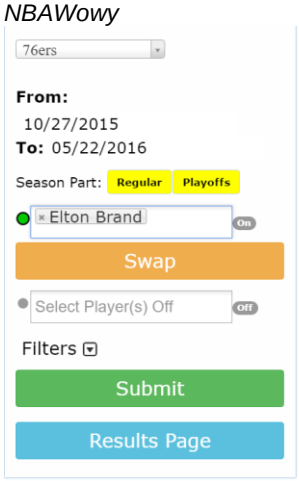
Aside from live analysis, items that we aimed to improve upon in our visualization were the following:

- Ability to explore more player statistics

- Expansion of score differential to show scoring of both teams so that offensive contribution vs. defensive contribution can be intuited
- Integration of historical lineup statistics

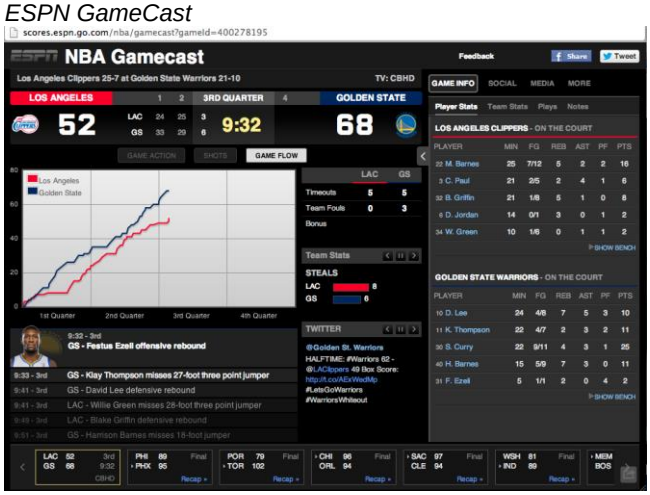


NBAWowy



NBAWowy is a popular engine for looking up lineup statistics for players and teams. In many ways, it inspired this project, because a common use case that we found for it was to look up lineup statistics while watching a game live. Wondering about this lead us to think of the benefits of having an analytical tool combined with a live visualization. NBAWowy has many UI issues, but its contribution to our work was mainly in showing the utility of season lineup statistics on informing solid analysis of NBA games; it is used often by prominent media figures to explain the results of games.

ESPN GameCast



Visually, our project was very similar to ESPN GameCast, which is an online webapp that aims to give as much real time information about a game as possible. It is somewhat of a proxy for watching a game live. As far as analysis goes, it also provides useful visualizations on the flow of the game so far, as well as traditional box score statistics. As far as the visual representation of the live progression of the game, we used GameCast as a starting point. We focused on modifying this existing visualization to integrate our lineup analysis functionality.

Importantly, we felt that GameCast did provide an engaging and informative way to view the game real time, but serves as a substitute to viewing games, whereas our goal for HoopMatchups is to make a tool to be used alongside watching the game live. This led us to subtract some of its features, and augment other features, as will be described later.

SUMMARY OF LEARNINGS FROM RELATED WORK

From the related work we learned that lineup statistics inform basketball analysis well, that existing visualizations can be improved upon, and that a tool integrated with a live game portal could be very useful.

METHODS

This section will detail how the various parts of our visualization were developed. To understand the parts that are being referenced, refer to the video overview of the visualization.

This section will be split into the following parts of HoopMatchups:

- Live Web App Infrastructure
- Data Engineering
- Live Court Visualization
- Backtracking Visualization (line chart)

Live Web App Infrastructure

The app was developed using NodeJS, SocketIO, Angular, D3, PostgreSQL, and Skeleton CSS. SocketIO was used to manage live connections to deliver game updates to clients. Angular was used to display the information from those updates in all of the non-visual cases, whereas D3 was used to form the court visualization and the backtracking visualization. Skeleton CSS was used as an HTML5 responsive boilerplate. Static data was stored in PostgreSQL and was interfaced with via pg-promise, a NodeJS package.

Data Engineering

There were two types of data collected for this visualization: static, and live. Static data included physical statistics of players, such as weight, height, wingspan, etc., season box score statistics for individual players, and season box score statistics for lineups.

Static Data

The static data was collected with a Python script that combined both scraping and API calls to various endpoints: NBA Stats, DraftExpress, and more. This data was aggregated into a single PostgreSQL database, and can be updated via a single script which could be run daily during the regular season to update all player and lineup information.

This data is leveraged by the web app via a database connector interface, which we called DBConnector, which supports commonly used requests such as information for all players in a game, information for all possible lineups in a game, etc.

An important step in aggregating the static data was creating global keys. Different data sources had different player IDs, for example. We found a useful global key with almost no overlap to "Firstname – Lastname", although this did result in some players not matching up, especially with players who have first names that are two initials, "J.R. Smith" for example. We manually fixed inconsistencies. These global keys also had to be referenced by our live data.

Live Data

The live data consisted of real-time game information, and was collected from the Sports Illustrated API. This data consisted of daily game schedules, live play-by-play data, and some basic stat aggregation during live games.

Our server handled the polling of the SI endpoint for live game data, preprocessed some of the data, and then packaged it for ingestion by the clients. A significant data wrapper was added to the SI data, which abstracted commonly used methods used against the data, such as creating timestamps, determining which players were on the court, and formatting play-by-play data.

However, much of the processing was also left to the client-side data abstraction. This was to simplify the work of the server, so that it did not have to keep track of the time that each client joined the server, but instead could send the same information to each client, and leave the visualization up to them.

The bulk of this processing occurs in GameUpdate.js, which is data wrapper for the live data served by the server, originating from the SI API endpoint.

Live Court Visualization

As mentioned above, the ability to determine which players were on the court was provided by the SI API. We combined the data of which players were on the court with information about which position they played from NBA Stats, which has detailed position information, such as Combo-Guard, Guard-Center, etc. However, for the visualization, we simplified it to three positions: Center, Forward, and Guard, which were placed in that order on the court, with the centers being on the edges of the court. When there are more than one player on the court per team in one position, the visualization divides them evenly on the y-axis.

The court was visualized using an NBA Stats court svg image file and D3.

The season lineup statistics were displayed by calculating the database key for that lineup given the live players on the court, requesting that data from the cache of the server, and then displaying it using Angular.

Controlling which stats are displayed for each player is also handled by Angular. Overall, very little static data is requested by the client once it has connected to the server. All of it is sent upon connection, and the only information sent thereafter is live data which is associated with the static data via the global keys.

Backtracking Visualization

The bulk of difficulty of the project lay in constructing the backtracking visualization. As an overview, the visualization shows the history of the game by showing a line representing points scored over time for each team. It places a circle when there is a change to the lineup for that team, the size of which is proportional to the number of players substituted. Thus, the line segments formed by the circles are portions of time played by a particular lineup in the game.

The circles can be hovered to show the details of the substitution, and the segments can be hovered to show the lineup during that period of play.

Each time an update to the game is sent to the client, it must recalculate the entire chart, since interpolation was used on the line segments in the plot; the subsequent line segments depend on the previous line segments.

Much of this could be done by processing the play-by-play data, since substitutions were noted as plays. Two data structures were introduced to abstract the play-by-play data into a form visualizable in this format: LineupChanges (circles), and LineupSegments (line segments). While traversing the list of plays given by the play-by-play data up to the current period in time, when substitutions were encountered, they were clustered by team to form a single LineupChange. At this point, which players were substituted in and which were substituted out could be calculated. LineupSegments consisted of a series of plays other than substitutions, across which stats could be calculated like points scored per team, assists, etc.

These data were fed to D3 to be visualized in real time supporting most of the functionality described previously. However, one important part of the visualization that was extremely difficult to achieve was the hovering of the line segments showing the current players on the court. It turns out that this is actually a difficult problem for lineup analysis in general. Play by play data gives the starting lineup of a team, as well as all substitutions during the game, however, they crucially lack substitutions *which occur during period changes*. If a player is substituted in during halftime, for example, there is no record of it in the Play-by-play. Our first attempt to fix this was to infer these substitutions by other actions of players on the court, such as a rebound. However, this did not give information on which player was substituted out until every player on the court records a play in the play-by-play.

The solution to this problem lay in integrating live on-court data, provided by SI, with the play-by-play data, and creating our own substitutions that happened during period changes. This integration was particularly hard because the on-court information provided by SI was available only on a polling basis, not on a game-time basis. That is, the resolution of the information was as small as our polling interval. Everytime we polled the endpoint, we could determine if a player was on the court or not, but, given a prior point in the game, we couldn't. So, we reduced our polling interval to a manageable amount, and recorded the first on-court data delivered by SI for each period start, and marked that down as a substitution that occurred between periods. Then, when constructing the LineupChanges, we inserted these substitutions into the dataset whenever we traversed plays that indicated the start of a new period.

RESULTS

A visual tour of the web app is available in video format with this paper. Therefore, in this section, we will describe the choices behind the different parts of the visualization, and discuss their effectiveness, propose improvements to each part, and conclude with a summary of improvements to be made.

On Court Visualization

We chose to display the player's first initial and last name at their position on the court to save space but preserve identifiability.

The second aspect to the live court visualization is the display of statistics underneath each player. Using a dropdown above the court, the user can select from an extensive list of stats to display for each player, which will appear directly underneath the player's name. We associated the dropdown to these stats by coloring both blue. The purpose of this format was to help users answer questions such as "why was that player just substituted in?" and "how does this player contribute to the lineups overall statistics?"

The third aspect to the live court visualization is the integration of lineup season statistics for each team. These are displayed to either side of the court to associate them with each team. One stat per side is also in blue, because it corresponds to the selected stat above. For example, selected fgm, or field-goal-makes, will show each players season average beneath his name. It will who the field-goal-makes average for that lineup for the season to the left and right of the court.

The visualization is successful in showing the players on the court in an engaging way by showing their formation. This also provides some intuition about the nature of the lineup: is the team playing tall post players, or shorter guards? This is conveyed by the formation displayed. However, the statistics for the players is not displayed in a visually intuitive way.

A visually engaging way to show these statistics could be to show a simple bar for each player that represents their value of the statistic normalized to either the players on the court or to NBA averages.

The lineup season statistics are easy to digest, and seeing them update in real time is an effective way to see how the lineups are changing. However, comparing the two lineups is not easy because the numbers are located opposite each other, and there is no visual representation.

An improvement to this part could be to show a bar chart for the two lineups, with a bar for each statistic and for each team. Further, a stacked bar chart which breaks down each player's contribution to the bar would make it easy to compare the two lineups and the players that compose them.

Finally, we said earlier that answering the question of why a player was substituted in was a main goal for this visualization. To that end, an algorithm that could determine automatically the salient differences between lineups would be a very useful addition.

Something that would also contribute to this would be animations that track the changes. It's easy to miss some changes to the lineups, but we could draw them out longer

by showing the players entering the court and leaving. Further, it would be particularly useful to view the court visualization for the lineup directly prior, to go back in time one lineup so to speak, to see the changes more clearly.

Backtracking Visualization

The backtracking visualization, without any of its interactivity or live-updating is unique and powerful. If our app only generated plots of points scored over time segmented by substitutions, it would be an extremely valuable contribution to basketball analytics visualization. It instantly conveys the back-and-forth nature of the game. It shows how much the coach influenced the game, and how the game changed over time. Adding on top of it the ability to see which substitutions caused the lineup changes and subsequent segments of the chart is even better, and seeing which players are responsible for different parts of the runs is as well.

Improvements to the backtracking visualization would be to include more aggregated stats over the line segments. What exactly happened during the portion? What was the lineups defensive efficiency not during the season but during these exact 5 minutes? Answering those questions is the next step for HoopMatchups and would make the tool at least twice as useful.

Continuing improvements to associating points on the graph with information about the game at that state, hovering over lineup segments could temporarily update the court to show what it looked like during that period of time. On top of that, it could update all of the lineup-descriptive visualizations previously proposed to what they were at that moment.

Case Study

The case study presented in the video overview is the Golden State Warriors vs. Oklahoma City Thunder Game 6 of the Western Conference Finals, one of the most contentious of the 2016 playoffs. Our visualization, when used in real time, captures much of this. The lines of our graph intersect at interesting moments, and lots of substitutions occur at crucial points in the game.

Some drawbacks that this game showed us about HoopMatchups is that our visualization fails to deliver specific insights about the lineups and their success or lack thereof. For example, when the Warriors come back at the end, the user is able to look at the lineup and guess, but there isn't a story to this data other than which players were on the court. Of course, they can watch the game live, but showing stats like the points scored by each player at that time, advanced stats like which player played defense against whom would provide an in-depth analysis of the team's performance due to the players that it put on the court.

Another specific point that using HoopMatchups during a game revealed was the lack of normalization, or a sense of what was average. What is the average height? How does this lineup compare to all the other lineups for this team, and for all other lineups in the NBA?

LEARNINGS

In this section we will discuss general learnings about visualization from HoopMatchups.

The first learning is to leverage interactivity to explain events over time. The backtracking visualization, before interactivity, makes the user wonder many questions. It gives him or her some information, but it gives him or her many more questions. This is where the interactivity comes into play: hovering over circles in lines answers some of these questions but also allows the user to create his or her own story of the data.

The second learning is that spatial orientation, in the context of basketball analytics, is powerful. Without reading any of the names on the court, we were able to get a sense of the two lineups put on the court just by seeing how many centers, forwards, and guards were being played. Especially in the current NBA, where we see many lineups without true centers, and some four-guard sets going against 2-guard sets!

Third, comparing data in table form is difficult, and not intuitive especially when the tables are spatially distant. This may not be so much of a learning rather than a tenant of data visualization, but our progress on HoopMatchups emphasized this point to us.

Our fourth learning had to do with live visualization and the addition of the play text ticker that is directly underneath the court visualization. Before adding this, the visualization did not feel very dynamic. The score and time updated every 10 seconds, but barring adding elaborate animations, there weren't many solutions to make it feel like a live companion. However, by adding the three most recent plays in descending shades of grey, we were able to add a definitively dynamic feel to the app. Furthermore, almost every piece of information other than these texts is aggregate or abstract dada in the form of box score statistics. Klay Thompson scored two points, but how? The play texts contain detailed information, such as "Klay Thompson makes 20-ft jumper" reinforces that the web app is associating real data from a live source that corresponds to the game that they are watching.

FUTURE WORK

Improvements to HoopMatchups were discussed in detail in the results section. This section will discuss the future overall direction of HoopMatchups. We intend to develop HoopMatchups into a more robust companion app for the NBA finals to find out more about its use cases. We have tested it with various NBA fans over the course of the 2016 playoffs, and have also shown it to staff of NBA teams. Both have shown interest in the concept, which leads us to believe that each could be a successful path.

As a companion app, the visualization would be simplified a lot, and would be more interactive. That is, we would want to show less information at once, and allow the user to pick anything specific they like to see. If possible, we would try to guess salient information.

However, as a tool for NBA teams, this becomes less of a concern compared to increased functionality. The main improvement to be made here would be to allow aggregation of advanced statistics over individual lineups over the course of the game, and possibly to leverage SportVU data along with the current statistics.

DIVISION OF WORK

Lawrence Rogers was enrolled in the class while Tony Liu was not. Work on the project started at the beginning of the quarter. Though we had discussed the idea quite a bit, no coding had been started at all. Over the course of the quarter, Tony worked on all parts of the data scraping and database setup. Everything else was written by Lawrence. In the code directory, this corresponds to everything outside of the models directory. As far as the visualizations and ideas went, we discussed these together and developed them collaboratively. Tony also knows a lot more about basketball than Lawrence.

REFERENCES

1. http://popcornmachine.net/
2. http://espn.go.com/
3. http://nbawowy.com/