CS 171 Final Project Process Book

College Basketball Visualized

Juancarlos Aponte

Initial Project Proposal

Background and motivation

We like college basketball. We follow it on TV, on the Web and in newspapers. We read game summaries and analysis, pore over box scores, fret over the standings and chew on statistics.

The season is soon ending. Before long, we'll have an entire season to look back on and try to make sense of. As fans, we're interested in performance--of players, teams, conferences. And we're interested in comparing performance.

Project objectives

College basketball junkies love statistics. But data, as found on popular sites like ESPN.com, Yahoo! Sports and Sports-Reference.com, as well as in newspapers, have one thing in common: They're tabular. Apparently, that's by convention. We'd like to reinvent descriptive statistics for college basketball as visualizations.

For this project, we'd like visually contextualize the week-to-week performance of all 351 Division-1 teams, 32 conferences, and more than 4,000 players in the 2013-14 season.

For player entities, attributes include routine statistical categories like points per game, rebounds, assists. For team entities, the attributes can include those as well as, across the course of the season, winning percentage, position in polls, RPI ratings, winning streaks, conference standing. For conferences, we can look at points and other usual statistical categories, in addition to winning percentage, poll representation, RPI ratings and record against other conferences.

The question we want to answer: How does the performance of this team, this player, this conference measure up?

Data

For player, team, conference and game data, we can scrape ESPN.com and Sports-Reference.com. The latter looks especially promising because it makes many tables available as csv files. We anticipate getting a good portion of our player, team and conference data there. One of ESPN.com's strengths is its excellent play-by-play data, which we may tap for visualizing a game's lead changes.

We found and tested some Python scripts on Github for scraping Sports-Reference.com. We may fork the repo and adapt the scripts to our needs.

For every team, we are interested in tracking through the season its overall RPI and strength-of-schedule RPI. We would also like to track conference RPI and conference strength-of-schedule RPI. RPI ratings for all 351 teams and 32 conferences, for each day of the season, are available at http://www.teamrankings.com/ncb/rpi/.

The site uses an Ajax call to retrieve these values by calendar day. Using curl, we can retrieve the RPI rating on a given day as an HTML table, which we can then scrape. Here's a curl command to retrieve the table of the RPI ratings for all 351 teams, for instance, on Dec. 11, 2013:

```
curl --data
"type=detail&league=ncb&rating_id=514&season_id=311&cat_type=2&view=team_v2&view_type=team&table_view=team_v2&force_period_id=&is_rpi_ranking=1&date=12%2F11%2F2013"
http://www.teamrankings.com/ajax/league/v3/rankings_controller.php
```

Similar methods can be used on this site to get strength-of-schedule ratings, for teams and conferences.

Because the season is not yet over, we may use 2012-13 season data so we don't have to worry about rescraping as this season plays out.

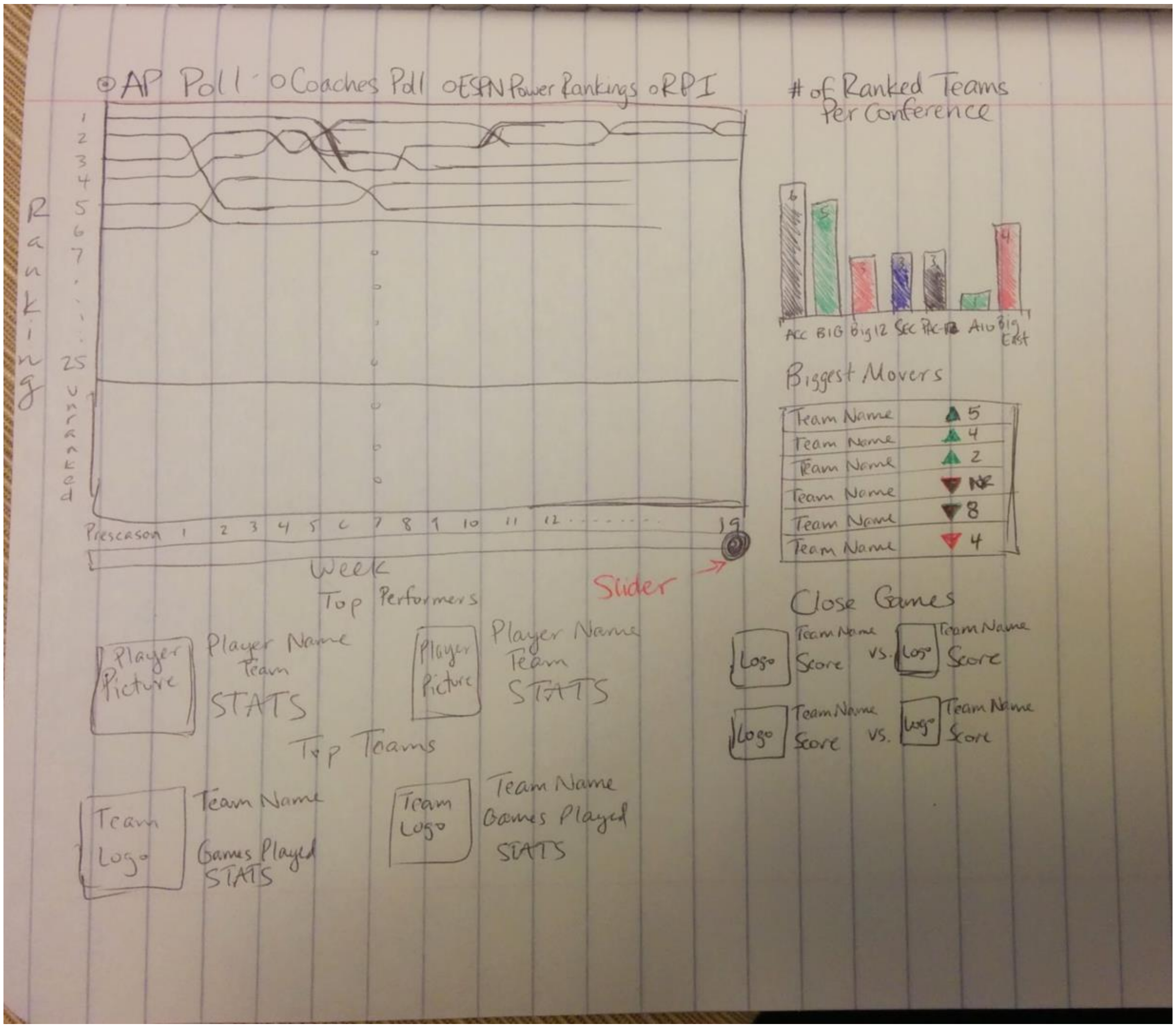
Data processing

We don't expect to need to clean the data substantially. From our initial tests, our data sources appear to be structured consistently. We expect to spend more time writing scripts to scrape the data than cleaning it. Although if the need arises, we will look to Excel, csvkit and regular-expression find-change operations in our text editors.

We believe all of the data--the player, team and conference statistics we're interested in--will be available on ESPN.com, Sports-Reference.com and TeamRankings.com, as described above.

Visualization

The main visualization will have a simple line graph that shows a weekly ranking of the top 25 teams with a selector for the poll (AP, Coaches/USA Today, ESPN Power Rankings, RPI, etc). A slider will be used to select the desired week. For each week, a separate column chart will display the number of ranked teams per conference. The biggest gainers and losers (# of places moved up/down in the poll versus previous week) will be listed. Top teams and players for that given week will be displayed below the polls graph. Any close or interesting games will be highlighted as well.



From the main visualization, a user should be able to click on a team or conference and access their relevant stats. The same goes for accessing a particular player.

Must-have features

We must be able to display information for every college basketball team, conference, and player. The ability to step through a season at least on a weekly basis is a must as well for the main visualization.

Optional features

An optional feature we might want to add is the ability to import and visualize previous seasons. We might want to go back to the introduction of the Coaches Poll in 1993-94 or perhaps back to the introduction of a 64-team NCAA Tournament format in 1985.

Project schedule

A tentative schedule (subject to change) is as follows:

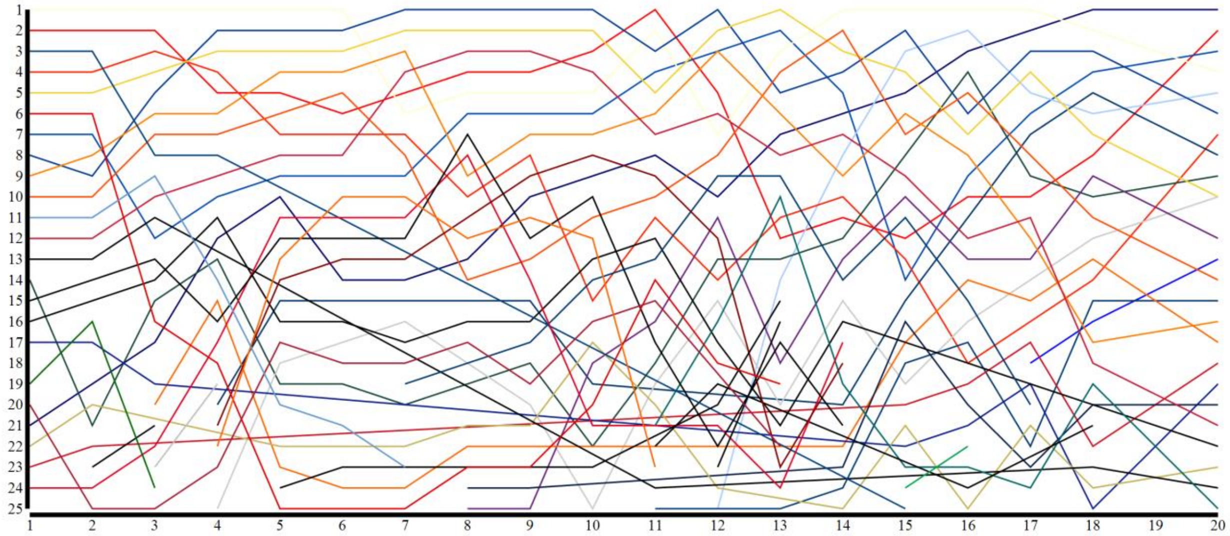
- Sunday March 23 - Have data downloaded and in repository, sketch visualizations for display of conference/team/player

- Sunday March 30 - Create main visualization
- Sunday April 6 - Conference/Team/Player visualizations created
- Thursday April 10 - Functional Prototype Due
- Thursday April 17
- Thursday April 24
- Thursday May 1

Detailed Process

March 23

Juancarlos – Created the initial poll graph. It currently shows the 2012-13 Associated Press poll data. I found a spreadsheet linked that contains schools and their team colors at (https://docs.google.com/spreadsheet/ccc?key=0Agb4DJOM-AsTdFNBMjl1ODRaQkNMMmp1Z3dEUnFmdWc&hl=en#gid=0). I noted that some of the team names don't match up with what is in the poll data, this should be cleaned up at a later date, but it is not imperative. For now, any missing data or colors that correspond to white (same color as current background) are interpolated as black, for ease of use. Currently, when a team goes in and out of the rankings the line starts/stops, which creates a lot of stray lines in the middle of the graph, although this does show how crazy the college basketball season can be. There should be some way of showing that a team falls out of the ranking, instead of just ending the line.

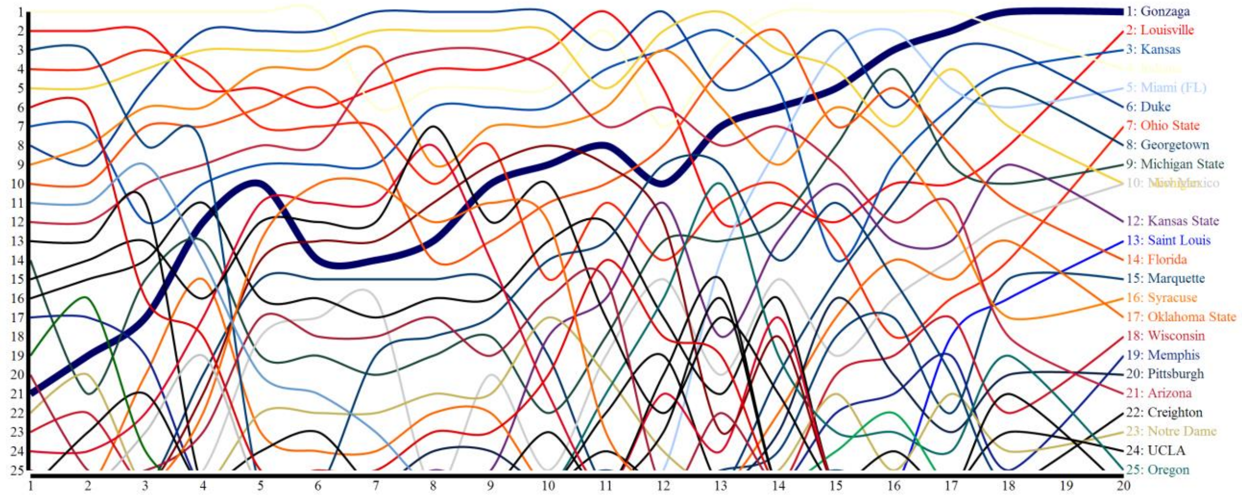


March 24

Patrick – There is an issue with differing id's on ESPN and Sports-Reference websites. Reconciled by creating a csv file with rows for each school along with the various id's, added team-rankings id's.

March 25

Juancarlos – I added some code to interpolate missing ranking data. When a team that is previously ranked falls out of the top 25, there is no entry. The code searches for any missing data and interpolates it as being a rank of 26 for the week that the team is unranked. Adding linear interpolation of “cardinal” to the graph changes the lines to curved lines and allows teams to flow smoothly out of the graph when unranked. I also added functionality to bold a team’s line when hovering over it.



March 26

Juancarlos – I changed the hover functionality so that it makes all other lines more transparent along with bolding the selected line. The line thicknesses were also changed to remove some of the cluttered areas where many teams are overlapping.

Patrick – Works on grabbing box scores.

March 27

Patrick – Downloads RPI data and parses it to only include data for each Monday of the season (this is the day that the Associated Press and Coaches Polls are released).

March 28

Juancarlos – Created a new master list of schools to correlate the different id’s on different websites as well as listing a friendly, human-readable school and team name. Added a javascript file containing functions for loading the data into the viz as well as getting the missing data based on existing data (e.g. only have espn-id, get all other id’s and friendly names).

I edited the existing code to provide functionality for displaying the different polls and radio buttons to select the poll. The code as is does not work properly as it does not remove the old poll, instead just appends new lines over the old ones. I asked Patrick for some help in remedying this situation.

Patrick – Offers suggestions and edits code to remove the old lines when updating the poll graph with a different poll.

March 29

Patrick – Begins to scrape gameplay data following a method suggested by Juancarlos wherein you first scrape the individual team's schedule page on espn for the gameid's and then scrape each game recap page for the play-by-play data, loading it into a json object and then outputting to a file.

March 30

Juancarlos – Found a d3 plugin called d3-slider that provides neat slider bars that allow you to run any function as you slide across.

March 31

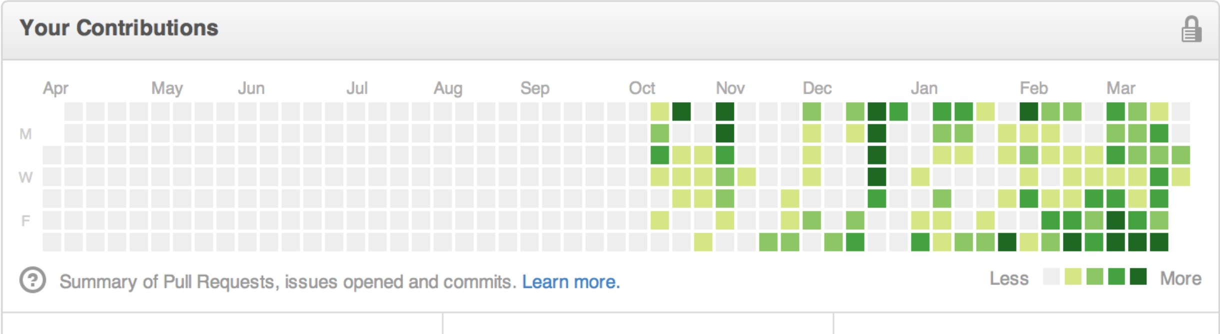
Patrick – Suggests that the slider is potentially overkill. Suggests instead that mousing over the individual week numbers on the graph's X axis could potentially perform the same function. Notes that the RPI data contains information for all 351 schools for every week of the season and that somehow we need to account for this excess of data, potentially by just removing it from our view.

April 1

Juancarlos – Suggesting implementing a d3 brush instead of the slider to act on the poll graph to highlight a specific portion of the season. This would help to de-clutter some portions of the graph which currently have many overlapping lines. We still have no idea what to do with all of our game data or where it fits in in our visualization. Our proposal offers to highlight top games for a week, upset games, etc, but perhaps a method of visualizing a box score would be preferable. Not sure how to fit it in with our current setup.

April 2

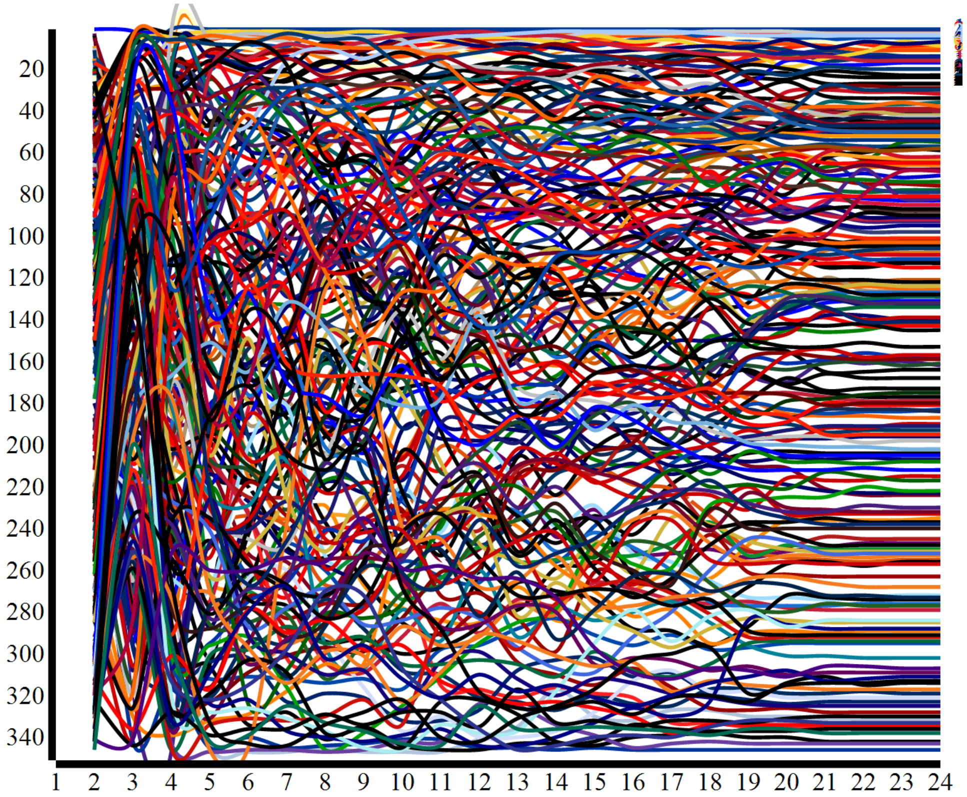
Patrick – Suggests showing every day of the season in a visualization. Suggests using something similar to the contribution activity heatmap on the Github profile page.



Summary of Pull Requests, issues opened and commits. Learn more.

Works on improving the play-by-play scraping. Suggests one way of handling the excess RPI data by changing the Y scale of the graph to go from 1 to 351 instead of 1 to 25.

Juancarlos – Adapts poll graph to change the Y scale as per Patrick's suggestion to 1-351. See below for new graph.



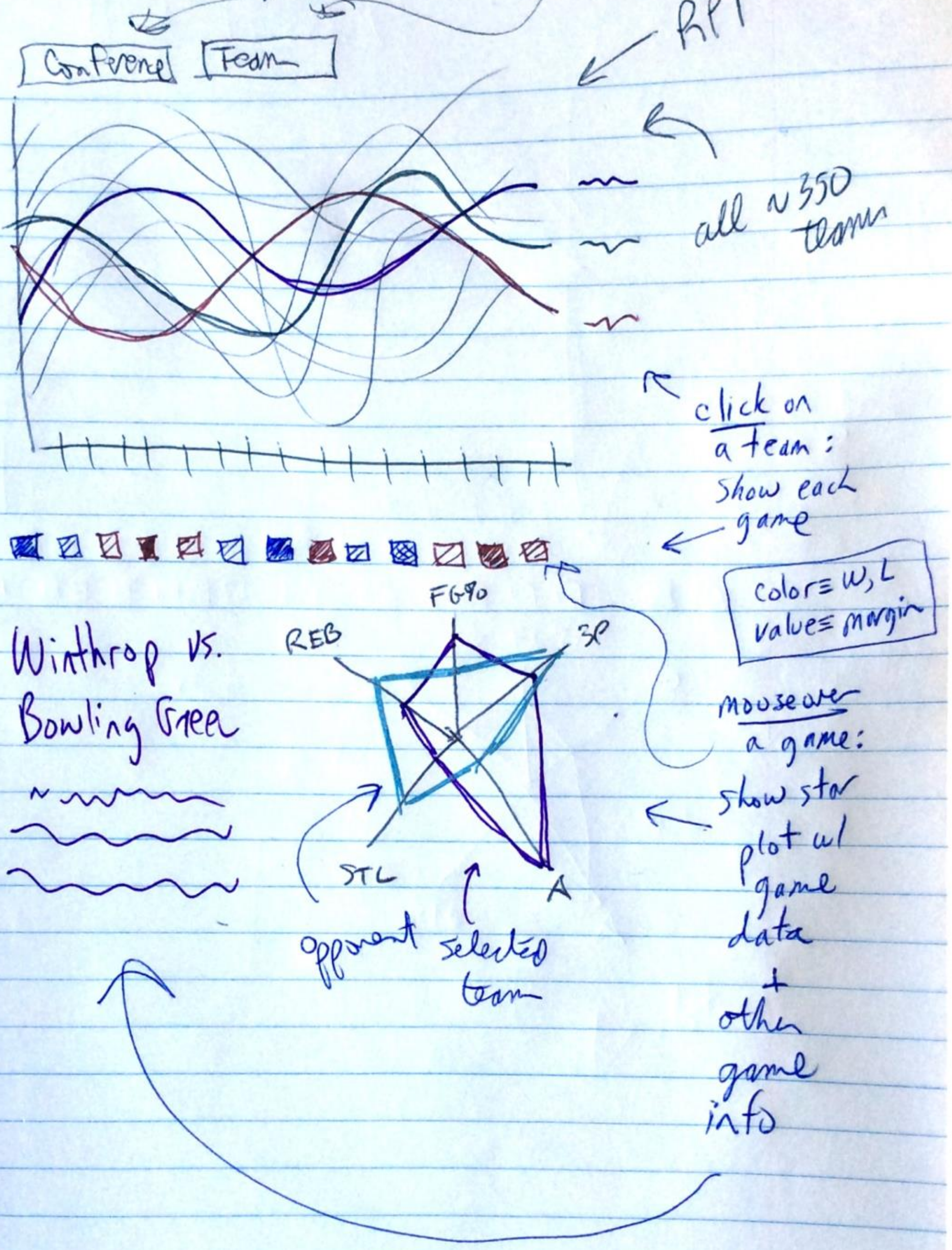
This is a nearly impossible view to absorb and interact with. It does do a neat job of showing how the RPI data is more objective – most teams swing a large amount in either direction week-to-week – whereas the polls, which are voted on, are more subjective and teams don't move around quite as much.

Also edited the RPI data and the new-master-id file to have all teams match the team names given on the ESPN website.

April 5

Patrick – Offers two new design ideas.

select teams by mouseover or dropdowns



RPI

all ~350 teams

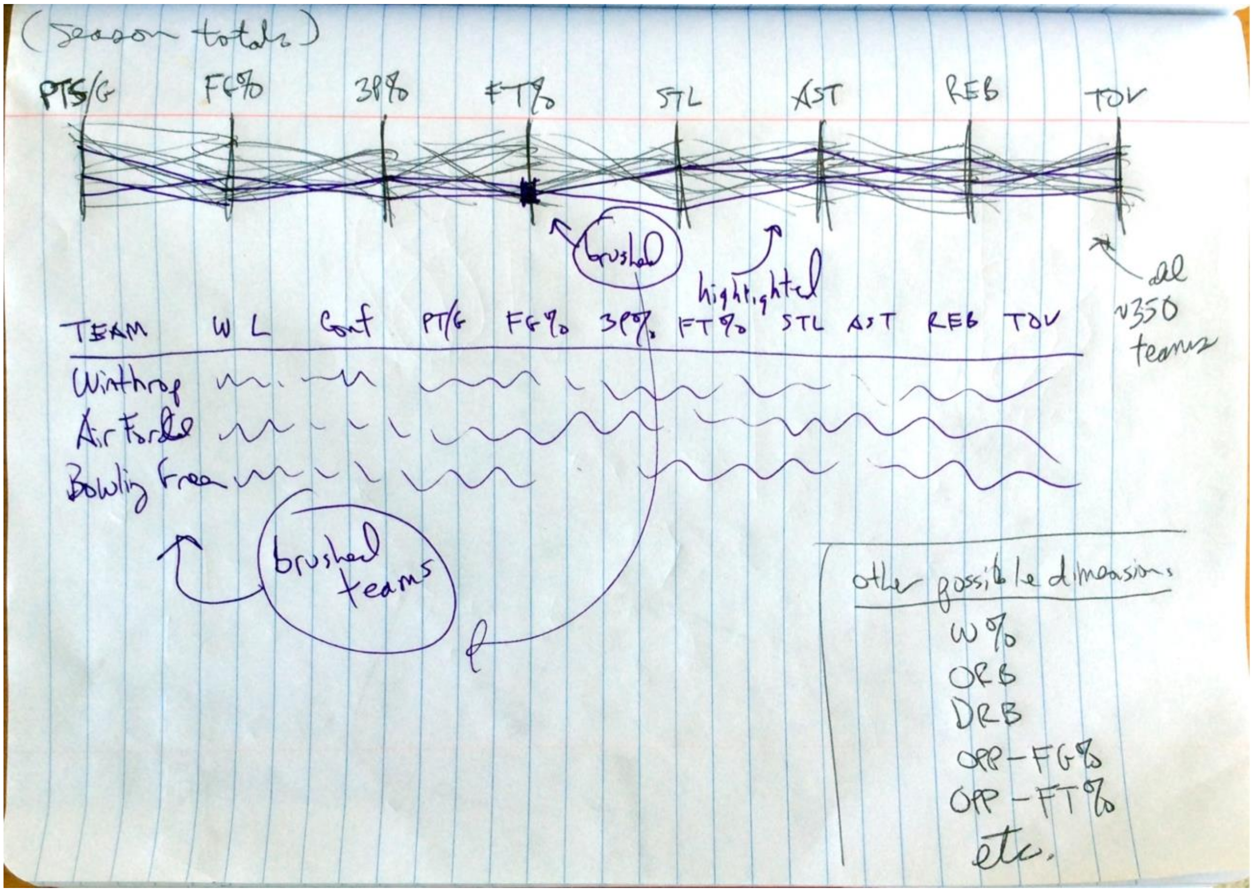
click on a team: show each game

color ≡ W, L
value ≡ margin

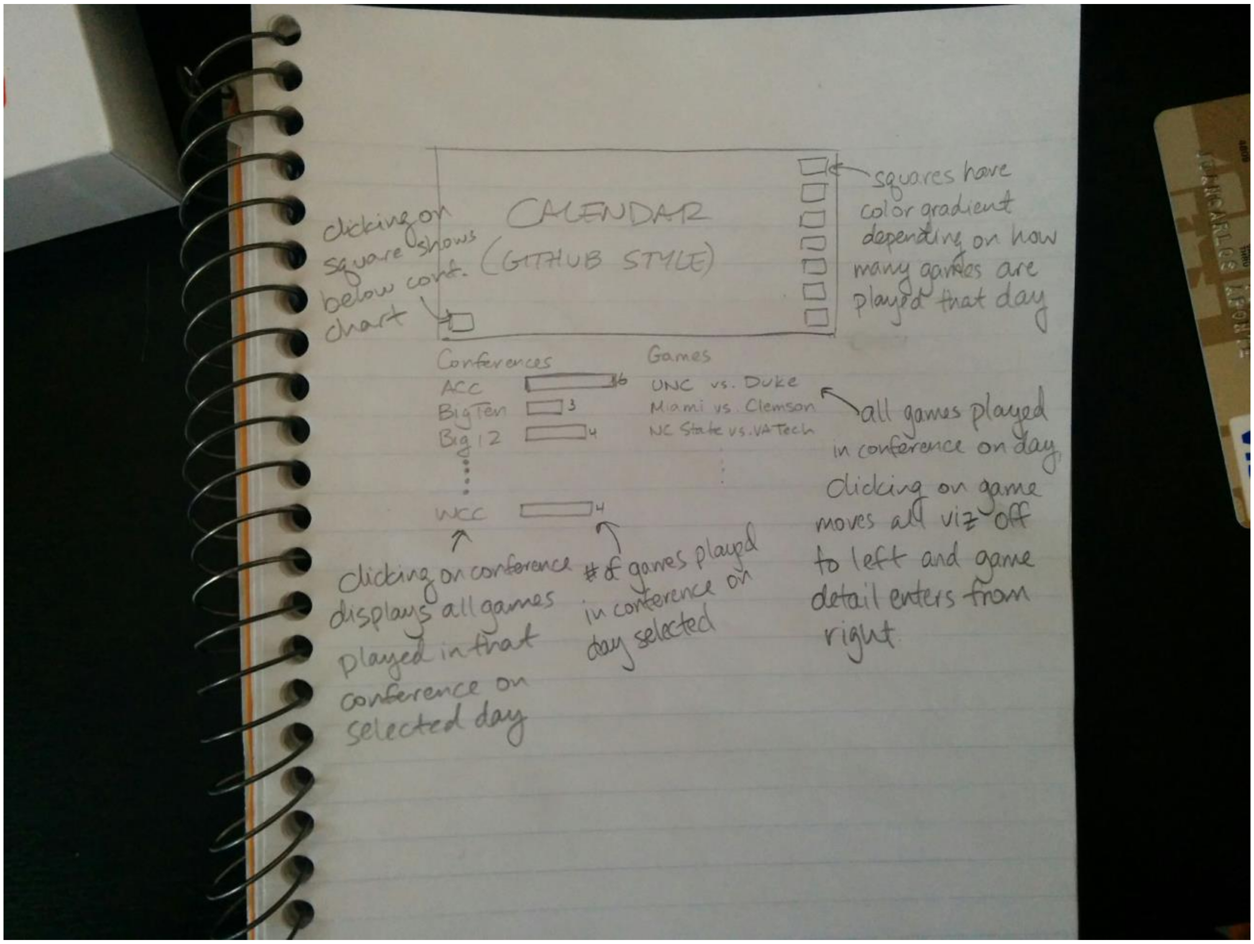
mouseover a game: show star plot w/ game data + other game info

opponent

selected team



Juancarlos – Created a sketch-up of what the previously mentioned heatmap calendar view might look like.


CALENDAR
(GITHUB STYLE)
clicking on square shows below conf. chart
squares have color gradient depending on how many games are played that day
Conferences
ACC
16
BigTen
3
Big 12
4
WCC
4
Games
UNC vs. Duke
Miami vs. Clemson
NC State vs. VATech
all games played in conference on day
clicking on game moves all viz off to left and game detail enters from right
clicking on conference displays all games played in that conference on selected day
# of games played in conference on day selected

Patrick – Offers to skip the poll graph originally in the proposal and worked on since the beginning of the project in lieu of either the heatmap or parallel coordinates view.

April 6

Juancarlos – I let Patrick know that his parallel coordinates view was good but alone was not enough for the project unless expanded on. I then suggested combining the separate visualizations we've created and melding them into a single coherent visualization. The heatmap calendar could serve as the focal point of the visualization allowing a user to drill down by day and then conference to see games played and then selecting a game to see a visualized box score and play-by-play data.

A "Season Overview" view would contain the poll graph and the parallel coordinates graph and would be linked to from the main view.

I also gathered gameid's for all 351 team's schedules and then scraped all ~5-6k games for relevant data and outputted that into a csv file.

April 7

Patrick suggests that the calendar view is not interesting. I suggest that we then ditch the calendar view and focus on fleshing out the poll graph and parallel coordinates view. Patrick works on re-creating the github style calendar.

Patrick edited the all-games csv file to include friendly names for home and away teams along with what conference the game is played in. It is incomplete and contains some values as NULL.

April 8

Disagreement over which views should be our top priority for the functional prototype.

Patrick – Works on the heatmap calendar view. Stuck on formatting and data wrangling.

Juancarlos – Fixes the issues in the heatmap and is able to display a color scaled calendar of the number of games played per day. There are latent design issues which prevent the calendar from displaying in the right order, November and December of 2012 display to the right of January-April of 2013.

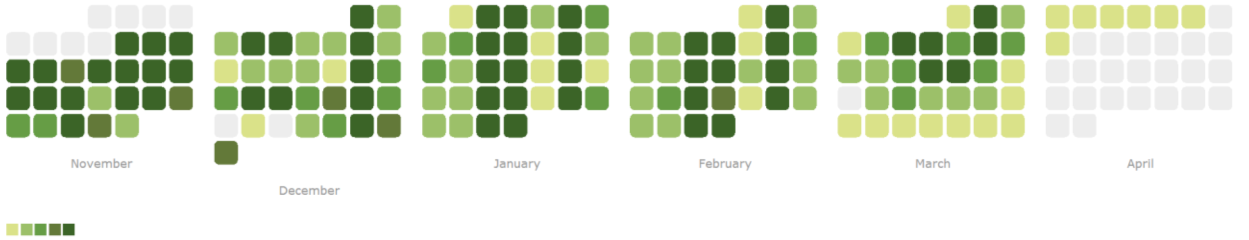
April 9

At this point, Patrick informs me that he wishes to continue the project solo.

I cleaned up the poll graph and added a table view on the right column below the conference bar chart that shows the biggest “movers” for the selected week.

April 10, 2014

I found a javascript plugin called cal-heatmap that creates heatmaps in a more concise code blob. There are some issues with using objects from variables as the data source, but it does work if I save the data and then input as a json file. The plugin does solve the issue from before where the calendar was laid out a little backwards. All months are now in order.


November
December
January
February
March
April