CSE308

Requirements Background Part 2
Congressional District Generation System

Reading

- Reading - all class members
  - https://en.wikipedia.org/wiki/Gerrymandering_in_the_United_States
**Project Teams**

- 9 students currently not part of a team
- For now, the team names (and team members) are:

<table>
<thead>
<tr>
<th>Team</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>Astros</td>
<td>Jihan Sethi, Kevin Tran, Anshar Kattula, and MD Hossein</td>
</tr>
<tr>
<td>Blue Jays</td>
<td>Edward Jiang, Sean Yip, Nick Purello, and Zhouqi Gong</td>
</tr>
<tr>
<td>Cubs</td>
<td>Omar Zarate, Chaudhry Khan, Jannemay Bhavsar, and Daniel Elansinski</td>
</tr>
<tr>
<td>Dodgers</td>
<td>Aditya Sheth, Christopher Giaramita, David Le, and Randy Paquette</td>
</tr>
<tr>
<td>Expos</td>
<td>Alexander Chin, Frederica Chen, David Xie, and Jeffery Wong</td>
</tr>
<tr>
<td>Giants</td>
<td>William Walters, Joseph Pellegrino, Ian Talla, and WeiFeng Lin</td>
</tr>
<tr>
<td>Indians</td>
<td>Michael Alvin, Joseph Schmitz, Jonathan Winters, and William Braxton</td>
</tr>
<tr>
<td>Marlins</td>
<td>Kevin Giraldo, Fearan Yaseen, Mohanur Alam, and Juan Tarquino</td>
</tr>
<tr>
<td>Nationals</td>
<td>Zengxian Liu, Qian Jiang, Linyss Cai, and Shengzhi Zhou</td>
</tr>
<tr>
<td>Orioles</td>
<td>Zihao Wang, Shenggu Jin, Hafeng Liang, and Kaixuan Chen</td>
</tr>
</tbody>
</table>

**Project Data**

- Congressional election districts
- Congressional election results – most recent 2 elections
- Election district data – data by precinct
- Population demographic data – available from US Census

Think about preprocessing some of the data (e.g., distinct precinct boundaries, precinct neighbor relations, allocation of demographic data to precincts, and precinct-precinct edge weights)
Single Run - Operation of Algorithm

- User starts algorithm from GUI (specifying # districts and OF weights)
- Single run GUI/Server interaction options
  - Synchronous - algorithm processes n iterations, and returns
  - Asynchronous - algorithm processes to conclusion, but sends updates asynchronously

Batch GUI Options

- User initiates a set of runs
- Limited updates when running (just status, no map updates)
- User specifies a range of OF weights
- Server runs all simulations, returning only status updates
- User is informed that all simulations have completed and a summary of results is displayed
- User can request that a given simulation be displayed in client
Overall Algorithm

- System generates an approximate allocation of precincts to districts
  - Graph partition algorithm partitions all precincts (nodes) into districts
- System refines the solution with individual precinct moves
  - Simulated annealing - approximates the global optimum of a function

Remember that algorithm performance is important

Approximate District Generation

- User specifies the desired number of congressional districts
- Use graph partitioning to generate an approximate set of districts
  - Step 1 - join some neighboring precincts to form chunks
  - Step 2 - join some chunks to form a reduced set of chunks
  - Step 3 - repeat step 2 until the number of chunks equals the desired number of congressional districts

Suggest holding off on design of this part for 1-2 weeks
Simulated Annealing Steps

- Start with the assignment of precincts to CDs generated by the previous step
- Select a (random) precinct to move to a neighboring CD
- Measure the change in objective function \( f \)
- Keep the change or reject the change
- Test for termination condition
- Repeat

Which Measures Do You Include?

- Voting result measures
  - Past voting results in region
  - Past voting results in nation
  - Many gerrymander measures use voting results data only
- Geospatial measures
  - Compactness
  - Alignment with natural boundaries
  - Consistency with other election districts
  - Geospatial measures have not been favored by the courts as a basis for striking down districting, but they have been included in state districting laws
  - SW for many of the measures will be provided to you
Measures in Your Objective Function

- Determined by your analysis of the constitutions of your states and class discussions
- Must include
  - Compactness
  - Political fairness
  - Population equality
  - Community of interest

Remember your final set of measures will be the superset of all 3-state group measures. Your SW design will need to plan for this.

Background to Political Fairness Measures

- 14th Amendment took effect in 1868
- Part of post-Civil War reconstruction
- Includes an equal protection clause
  - No state shall deny “the equal protection of the laws”
  - Basis for many voting rights laws and court rulings
Vieth v. Jubelirer Case

- 2004 case heard before the Supreme Court
- Developed an explicit distinction between racial and political gerrymandering
- Basis for much of the current gerrymandering interest
- Plurality decision that political gerrymandering is not unconstitutional
- However, Justice Kennedy left open the possibility that political gerrymandering could be considered unconstitutional if there were better measures
- Justice Kennedy retired from Supreme Court in 2018

A plurality is a winning vote, but not an absolute majority

Gerrymander Measures Considered in Vieth

- Majority of votes, majority of seats
- When does gerrymandering go too far? - Gerrymandering is OK in individual districts, but might not be OK in statewide redistricting
- Comparison of proposed districting with neutral or algorithmic districting
- Partisan symmetry - no precise measures were available at the time

"No judicially discernable and manageable standards for adjudicating political gerrymandering claims have emerged. Lacking them, we must conclude that political gerrymandering claims are nonjustifiable"
Partisan Symmetry

- Used in many measures
- Informally, it measures the notion that the percentage of seats won by a party is somehow consistent with the percentage of total votes won by that party

Packing and Cracking Measures

- Most proposed measures analyze the extent to which districts are
  - Packed (very large vote majority to the underrepresented party) and
  - Cracked (modest vote majority to the overrepresented party)
- Typical results show 70% - 80% in packed districts and 55% in cracked districts
- Measure of vote percentage is usually the 2-party vote, which means you subtract all the third party votes and measure the percentage of votes for the remaining data

When you get to design phase, be sure to consider data for all parties
Efficiency Gap Measure

- Stephanopoulos & McGhee
- Mentioned often in regard to the fall 2017 Supreme Court case
- Defines wasted votes
- Measures difference between wasted votes for each party within a statewide election
- Aggregates packing and cracking measure into a single number

Ethnic Minority Data

- Ethnic minority data should be included in your precinct data
- 1965 Voting Rights Act mandates the establishment of districts in which an ethnic minority constitutes a majority within the district
- Majority-minority requirement has been used to pack districts
- Your system will have an option for n majority minority district and a setting for the max and min thresholds
Database

- Supported CS database (e.g., MySQL)
- What are the entities?
- What are the attributes?
- What are the data types of the attributes?
- What data are not obvious from the GUI?
- What are the search fields?
- What searches are not obvious from the GUI (e.g., login)

OO Structure

- How do you identify the classes you need?
- How do you identify the attributes of each class?
- How do you determine the best type for an attribute?
- What is the best relationship among your classes?

Remember to use encapsulation in your OO design

Start to think about the way you would incorporate multiple algorithms in your design
**Persistence Layer**

- Your design will be object-first
- Database design will allow you to persist much of your OO data
- You will implement a layer (or use a tool) to
  - Retrieve DB data
  - Persist OO data

  Requires OO-Relational mapping

JPA might not be applicable

![Diagram showing Java Application (no SQL) connecting to Persistence Layer, which connects to DB](image)

**Reports**

- You will determine the data that is too complex to be expressed in the GUI, and develop reports for those data
- Reports essential for the multi-simulation case
Scope

- Scope of the system is defined by your set of use cases
- A recommended set of use cases will be given to you following the requirements phase
- The list will include required use cases and optional use cases
- Use cases relating to standard system operation (e.g., change password) will be less important

Use cases are not a great fit for this project, but will be used as a way of normalizing units of work

There will likely be about 40 use cases

How do you Assign Responsibilities to Team?

- Some parts of the project are standard SW development
- Major risk/unknown areas
  - What is the user interface? How do you display maps?
  - How do you translate the project reading into project requirements? Good to assign a team member to each of the risk areas early in the project
  - What data is associated with your requirements?
  - What are the best sources of that data?
  - How will you extract that data if it is not available in a standard format?
  - How do you test your system? To what do you compare your results?
High Priority Project Tasks

- Understand terminology and concepts in problem domain (read background references)
- Select 3 states to use in your development DB
- Read the districting language in the constitutions of your 3 states
- Search for data and think about a starting DB structure that includes nodes and edges
- Understand advantages and limitations of algorithmic options
- Think about the components in your GUI and Ajax updates
- Think about data normalization (how do you use different metrics in a single objective function)
- Build a simple system prototype to help understand SW design issues
- Start to write your objective function (on paper)