

COMPENSATION ANALYSIS TOOL

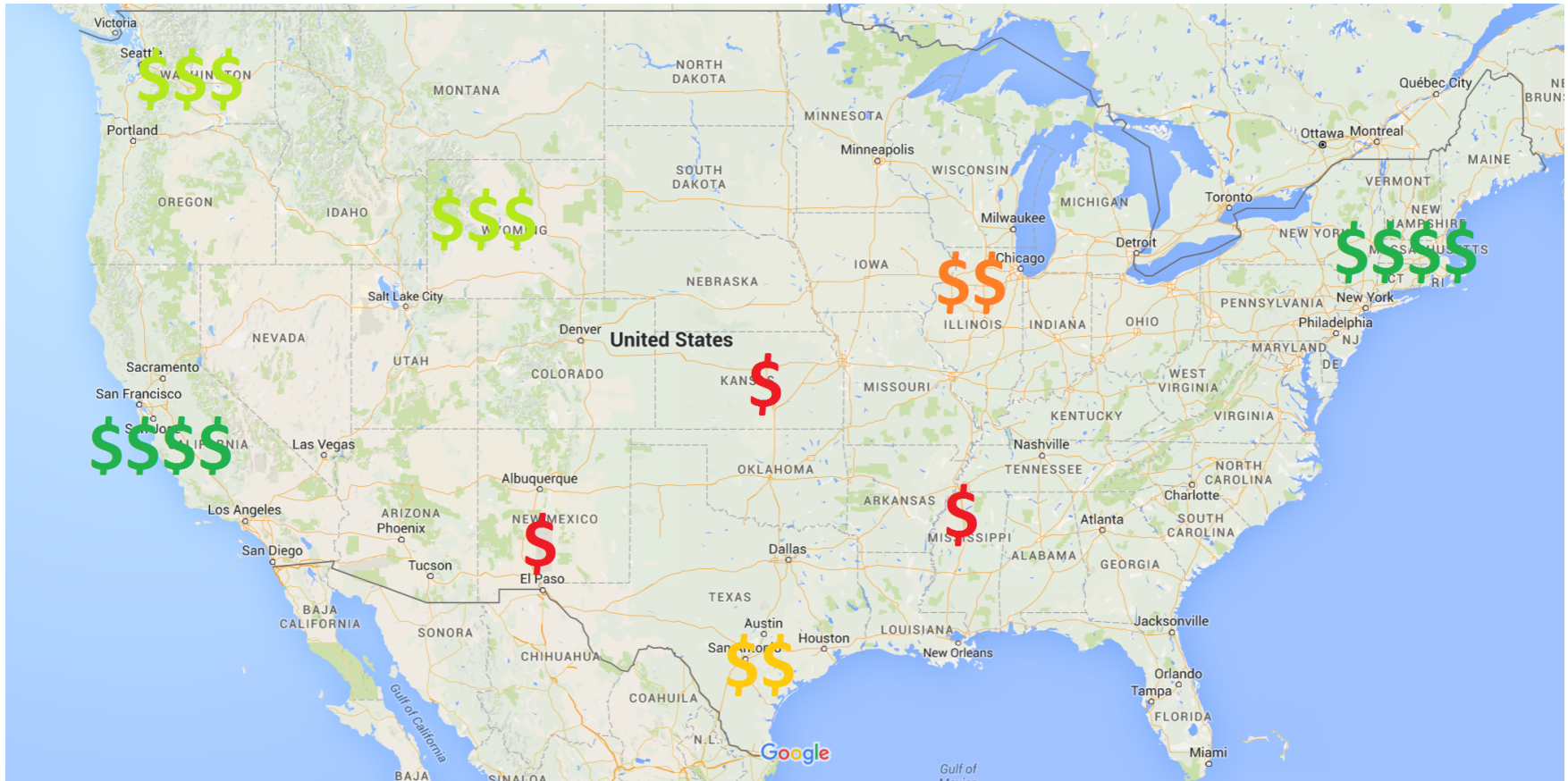
Steven Palmer & Ali Reda

CS 2XB3

L04 GROUP 32

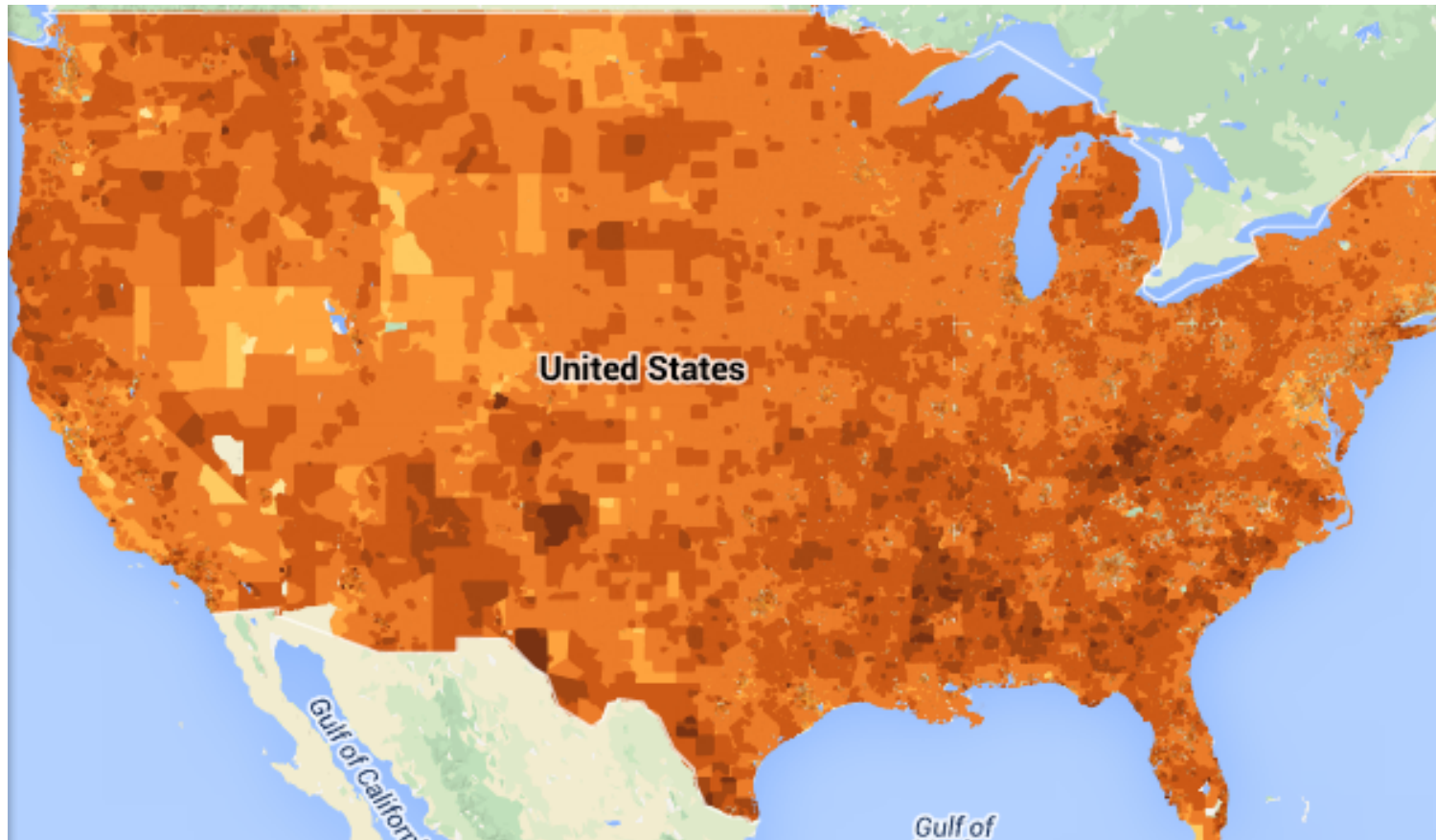
Problem

Compensation for a particular career can vary dramatically based on your location.



Problem

BUT: Cost of living also varies based on location!



Which locations are best from a financial point of view?

OBJECTIVE & SCOPE

The Analysis Tool will:

- Assess and rank the most viable locations to start a career.
- Produce a list of the best locations based on cost of living and employment statistics.
- **Motivation** is to help new graduates find a good balance between cost of living and average income in their wanted location.

Data Sets

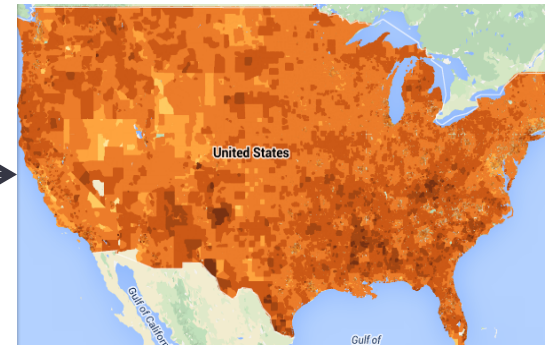
User
Query ↓

Location Data

Employment Data

Output

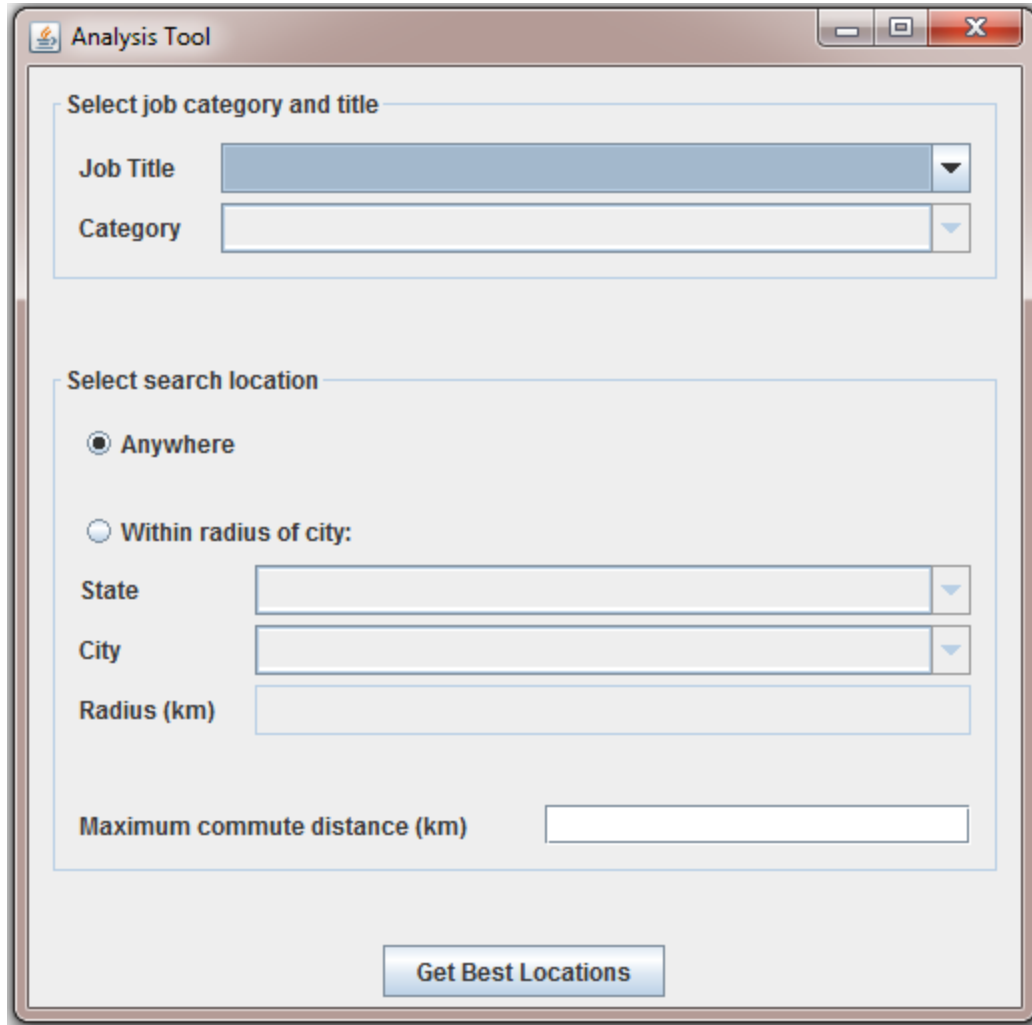
↑
Affordability Data



Datasets Used:

- Zillow Real Estate Research Data
- Occupational Employment Statistics published by The US Department of Labor
- Geographic location data published by the GeoNames database

Functional Requirements



The screenshot shows a window titled "Analysis Tool" with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains two main sections. The first section, "Select job category and title", has two dropdown menus: "Job Title" and "Category". The second section, "Select search location", contains two radio buttons: "Anywhere" (which is selected) and "Within radius of city:". Below the radio buttons are three input fields: "State" (a dropdown menu), "City" (a dropdown menu), and "Radius (km)" (a text input field). At the bottom of the window, there is a "Maximum commute distance (km)" label next to a text input field, and a "Get Best Locations" button.

- User can input wanted job title, category, location, commute distance, and desired residence square footage
- Application must use the Implemented datasets to list outputs of users inputs.
- Application must sort results In an proper manner

Non-Functional Requirements

- Must look appealing and user friendly.
- Easy to use, and easy to read.
- Results must be fast, and must be shown in a clear manner
- Application must be available at all times.

Algorithmic Challenges

- Sorting and searching algorithms were used to navigate the loaded data sets
- A graph was used to represent paths between cities:
 - nodes = cities
 - edges = paths between cities, weighted for distance
- The main challenge was creating a graph of connections between cities: needed to keep edge count “reasonable”
- Solution: connect all major cities within 800 km of each other, then for each major city connect all small cities within 200 km

Verification & Validation Methods

- The correctness of the data loaders/structures and the analysis methods were verified using JUnit test cases
- Manual testing was also carried out throughout the implementation process to ensure that the application functioned properly

Demonstration & Questions.