Principles of Operations Research Project

Redistricting Project Final Report

Redistricting New Mexico State by minimizing cut edges

By
Arshdeep Kaur, Malay Pandya, Dan Ni Lin

Professor Name
Dr. Hamidreza Validi

Department of Industrial, Manufacturing, & Systems Engineering
Texas Tech University
Lubbock, Texas
December 2023

Contents

1 Executive Summary

The project's goal is to optimize New Mexico's redistricting process while meeting most of the applicable state and federal requirements. To improve compactness and contiguity, the goal is to minimize cut edges by a model proposed by Validi's et Al work. [8] Our group was able to obtaine a feasible solution and an optimal objective value of 19. Our redistricting map output was compared to Validi's et Al work [8] which the main difference is the fact that our smallest district incorporated one more county, while Validi's et Al work smallest district has only 2 county. A possible explanation was the dataset source, our group used the 2020 data while Validi's et Al work used the 2010. Furthermore, while our model aligns well with the criteria of compactness and contiguity, and adheres to the prohibitions regarding prior district cores and incumbents, it exhibits limitations in comprehensively addressing the preservation of political subdivisions and communities of interest. Finally, it's possible by using LP knowledge to propose a redistricting model.

2 Introduction

Redistricting is the process of redefining the district lines from each state and they are redrawn every 10 years following the completion of USA census.[1] Once lines are redefined, it will change the voters population, identity, allegiance and political priorities of a district's representative, and of the legislative as a whole.[2]

Our group is going to choose New Mexico state.

3 Redistricting Criteria

3.1 Federal Criteria [3]

Population Equality: Federal standards address population equality among a state's congressional districts. The U.S. Supreme Court has addressed population size variance among congressional districts within a state, or malapportionment. Under the "equality standard" or "one person, one vote" principle, the Court has found congressional districts within a state should be drawn to approximately equal population sizes. Across states, however, district population sizes can vary.

Racial and Language Minority Protections: Another federal requirement comes from Section 2 of the Voting Rights Act (VRA), as amended, which prohibits states or political subdivisions from imposing any voting qualification, practice, or procedure that results in denial or abridgement of the right to vote based on race, color, or membership in a language minority. Under the VRA, states cannot draw district maps that have the effect of reducing, or diluting, minority voting strength.

Compactness and contiguity: Compactness and contiguity are both related to a district's shape. A compact district represents a geographically consolidated area. Thirty-one states require compact congressional districts, but often, state laws do not specify precise measures of compactness. A district is generally thought to be contiguous if it is possible to travel between any two points in a district without crossing into a different district.

Political Subdivisions and Communities of Interest: Thirty-one states require consideration of existing political subdivisions (e.g., towns, cities, or counties). People within a community of interest generally share a background or characteristics that may be relevant to their legislative representation (e.g., a social, cultural, historical, racial, ethnic, partisan, or economic identity).

Political Competition or Considering Existing District/Incumbent: Some states include measures prohibiting districts intended to unduly favor or disfavor a candidate or political party. Gerrymandering is a term often used for the process of drawing districts to benefit a particular party. Some states expressly allow the use or consideration of party identification information in the redistricting process, whereas others prohibit it.

3.2 New Mexico Redistricting Criteria [4]

Required: Compact, Contiguous, Preserve Political Subdivisions and Preserve Communities Of Interest.

Prohibited: Preserve Cores of Prior Districts and Avoid Pairing Incubents.

4 Problem Statement

The objective of this project is to develop a congressional districting plan for the state of New Mexico. This model will adhere to most of the state and federal criteria and minimizing the cuts from the graph obtained from New Mexico dataset available in lykhovyd [7] while also balancing the population across districts as evenly as possible.

5 New Mexico Districts [5]

Table 1: Amount of Districts in New Mexico State

Description	**Number of Districts**
Amount of districts in 2010	3
Amount of districts after 2020	3

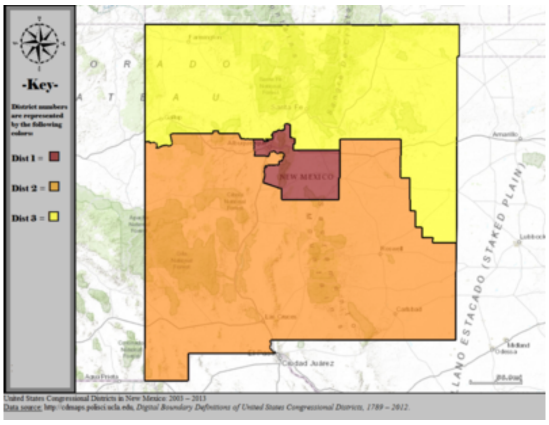
-Key-
District numbers are represented by the following colors:
Dist 1 = 🟥
Dist 2 = 🟧
Dist 3 = 🟨
NEW MEXICO
United States Congressional Districts in New Mexico: 2003 – 2013
Data source: http://cdmaps.polisci.ucla.edu, Digital Boundary Definitions of United States Congressional Districts, 1789 – 2012.

Figure 1: New Mexico congressional districts after 2010

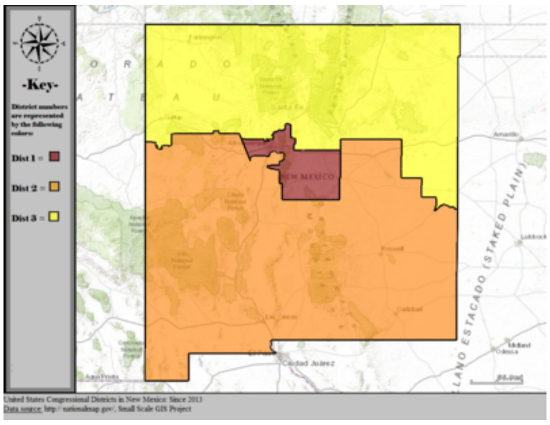
-Key-
District numbers are represented by the following colors:
Dist 1 = 🟥
Dist 2 = 🟧
Dist 3 = 🟨
NEW MEXICO
United States Congressional Districts in New Mexico: Since 2013
Data source: http://nationalmap.gov/, Small Scale GIS Project

Figure 2: New Mexico congressional districts in 2020

According to Wikipedia source [5], 2020 map has been adopted until. The criteria adopted in 2020 map is in section 3.

From 2020 redistricting deviation table [6], we can see that New Mexico districts population had 0.00% deviation.

To get a population bounds of 1% in New Mexico state (2.117.522 total population in 2020 [7]), we define the number of districts k and the allowed population deviation as follows:

$$k = 3 \quad \# \text{ Number of districts}$$
$$\text{deviation} = 0.01 \quad \# \text{ 1\% population deviation}$$

Next, we calculate the lower (L) and upper (U) bounds for the population in each district:

$$L = \lceil (1 - \frac{\text{deviation}}{2}) \times \frac{\text{total\_population}}{k} \rceil = 702.312$$
$$U = \lfloor (1 + \frac{\text{deviation}}{2}) \times \frac{\text{total\_population}}{k} \rfloor = 709.369$$

6 Data source

Data source of New Mexico county population can be taken from lykhovyd [7]. We used the following files available:

File Type	File Name
CSV File	NM_distances.csv
DIMACS Graph File	NM.dimacs
Hash File	NM.hash
Population Data	NM.population
Shapefile	NM_centers.shp

Table 2: Files used in the analysis

7 Redistricting Model in worlds

Description of optimization model(s)'s objective/constraints in words which is understandable by the general public

The Basics

Redistricting is like drawing lines on a map to decide which neighborhoods (counties) belong to which voting areas (districts). This model is a set of rules to help make these decisions fairly.

Key Elements - decision variables

- **Places:** These are specific spots on the map, like landmarks or houses. For our problem, it means the counties of New Mexico State.
- **Connections:** These are the lines between places, like roads or walking paths.

The Main Goal - objective function

- **Minimizing Border Crossings:** The aim is to reduce the number of connections that cross district boundaries. It's like having fewer roads that lead out of your neighborhood.

The Rules - general constraints

1. **Assigning Places:** Each place is put into one voting district, similar to deciding which neighborhood a house is in.
2. **Keeping Neighbors Together:** We try to keep connected areas in the same district, much like keeping adjacent houses in the same neighborhood.
3. **Balancing Population:** Each district should have about the same number of people.
4. **Creating Connected Districts:** The places in a district should be connected, forming a single, unified area.

Special Points - contiguity constraints

- **Central Points (Roots):** Each district has a main point from where the district starts.
- **Internal Connections (Flow):** This is about how the places in a district are connected, ensuring easy movement within the district.

8 Linear Programming Model for Redistricting [8]

The following model was taken from Hamidreza Validi and Austin Buchanan paper which discusses about ”Political districting to minimize cut edges”.

Decision Variables

$$x_{ij} = \begin{cases} 1 & \text{if vertex } i \in V \text{ is assigned to district } j \in \{1, 2, \dots, k\} \\ 0 & \text{otherwise} \end{cases}$$

$$y_e = \begin{cases} 1 & \text{if edge } e \in E \text{ is cut} \\ 0 & \text{otherwise} \end{cases}$$

The model is formulated as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} y_e && (1a) \\ \text{subject to} \quad & x_{uj} - x_{vj} \leq y_e \quad \forall e = \{u, v\} \in E, \forall j \in [k] && (1b) \\ & \sum_{j \in [k]} x_{ij} = 1 \quad \forall i \in V && (1c) \\ & L \leq \sum_{i \in V} p_i x_{ij} \leq U \quad \forall j \in [k] && (1d) \\ & x \in \{0,1\}^{n \times k},\ y \in \{0,1\}^m && (1e) \end{aligned}$$

The objective (1a) minimizes the number of cut edges. Constraints (1b) indicate that edge $e = \{u, v\}$ is cut if vertex $u \in V$ — but not $v \in V$ — is assigned to district $j \in [k]$. Constraints (1c) ensure that each vertex $i \in V$ is assigned to one district. Constraints (1d) ensure that the population of each district is between L and U.

Cut Edges Edges $\{i, j\} \in E$ that are “cut” are those whose endpoints i and j belong to different districts. Intuitively, the cut edges are those that would need to be snipped with a pair of scissors to break the graph into its districts. [8]

In this model is also added contiguity constraints and it’s adopted the single-commodity flow (SCF) formulation. This formulation introduces flow variables f_{ij} and f_{ji} for each edge $\{i, j\} \in E$. Additionally, binary variables r_{ij} are used to indicate if vertex $i \in V$ is the root of district $j \in [k]$.

The model includes the following constraints:

$$\begin{aligned} \sum_{i \in V} r_{ij} &= 1 && \forall j \in \{1, 2, \dots, k\} && (4a) \\ r_{ij} &\leq x_{ij} && \forall i \in V, \forall j \in \{1, 2, \dots, k\} && (4b) \\ \sum_{u \in N(i)} (f_{ui} - f_{iu}) &\geq 1 - M \sum_{j=1}^{k} r_{ij} && \forall i \in V && (4c) \\ f_{ij} + f_{ji} &\leq M(1 - y_e) && \forall e = \{i, j\} \in E && (4d) \\ f_{ij}, f_{ji} &\geq 0 && \forall \{i, j\} \in E && (4e) \\ r_{ij} &\in \{0, 1\} && \forall i \in V, \forall j \in \{1, 2, \dots, k\} && (4f) \end{aligned}$$

Hojny et al. [9] recommends setting $M = n - k + 1$. Constraint (4a) ensures each district has one root. Constraint (4b) dictates that vertex $i \in V$ cannot be the root of a district j it does not belong to. Constraint (4c) requires vertex i to consume flow unless it is a root. Constraint (4d) prohibits flow across cut edges.

9 Python Code for the LP Model

```python
import pandas as pd
import networkx as nx
import gurobipy as gp
from gurobipy import GRB

# Load population data
population_df = pd.read_csv('/Users/dannilin/Desktop/OR_Class/Final_Project/NM/NM.
    population', sep=" ", skiprows=1, names=["Index", "Population"])

# Create a dictionary for population lookup
population_dict = population_df.set_index('Index')['Population'].to_dict()

# Load and parse adjacency data to create a graph G
G = nx.Graph()
with open('/Users/dannilin/Desktop/OR_Class/Final_Project/NM/NM.dimacs', 'r') as file:
    for line in file:
        if line.startswith('e'):
            _, u, v = line.split()
            G.add_edge(int(u), int(v))
G.add_nodes_from(range(max(G.nodes()) + 1))  # Assuming continuous numbering
```

Listing 1: Python code for loading data

```python
import math

# Total population
total_population = population_df['Population'].sum()

# Number of districts and deviation
k = 3  # Number of districts
deviation = 0.01  # 1% population deviation

# Calculate L and U
L = math.ceil((1 - deviation / 2) * total_population / k)
U = math.floor((1 + deviation / 2) * total_population / k)

# Print the results
print("Using L =", L, "and U =", U, "and k =", k)
```

Listing 2: Python code for calculating bounds

```python
# Create the Gurobi model
m = gp.Model()

# Number of districts and deviation
k = 3  # Number of districts from previous calculation

# Variables
x = m.addVars(G.nodes, range(k), vtype=GRB.BINARY)  # x[i, j] equals 1 when county i is
    assigned to district j
y = m.addVars(G.edges, vtype=GRB.BINARY)  # y[u, v] equals 1 when edge {u, v} is cut
```

Listing 3: Python code for model and variables

```python
# Objective: Minimize cut edges
m.setObjective(gp.quicksum(y[u, v] for u, v in G.edges), GRB.MINIMIZE)
```

Listing 4: Python code for objective function

```python
# Constraints

# Each county assigned to one district
for i in G.nodes:
    m.addConstr(gp.quicksum(x[i, j] for j in range(k)) == 1)

# Population constraints for each district
for j in range(k):
    m.addConstr(gp.quicksum(population_dict[i] * x[i, j] for i in G.nodes) >= L)
    m.addConstr(gp.quicksum(population_dict[i] * x[i, j] for i in G.nodes) <= U)

# Edge cut constraints
for i, j in G.edges:
    for v in range(k):
        m.addConstr(x[i, v] - x[j, v] <= y[i, j])

# Update and optimize the model
m.update()
```

Listing 5: Python code for model constraints

```python
# The 'big-M' value as proposed by Hojny et al.
M = G.number_of_nodes() - k + 1

# Add constraints to the model

# Each district j should have one root
for j in range(k):
    m.addConstr(gp.quicksum(r[i, j] for i in DG.nodes) == 1)

# If node i is not assigned to district j, then it cannot be its root
for i in DG.nodes:
    for j in range(k):
        m.addConstr(r[i, j] <= x[i, j])

# Flow constraints: if not a root, consume some flow; if a root, only send out flow
for v in G.nodes:
    m.addConstr(gp.quicksum(f[u, v] - f[v, u] for u in DG.neighbors(v)) >= 1 - M * gp.
        quicksum(r[v, j] for j in range(k)))

# Do not send flow across cut edges
for i, j in G.edges:
    m.addConstr(f[i, j] + f[j, i] <= M * (1 - y[i, j]))

# Update and optimize the model
m.update()
```

Listing 6: Python code for contiguity constraints

```python
m.optimize()
```

Listing 7: Python code for model optimization

```python
# Print the number of cut edges
print("The number of cut edges is", m.objVal)

# Retrieve the districts
districts = [[i for i in G.nodes if x[i, j].X > 0.5] for j in range(k)]

# Map district numbers to counties
# Using the population DataFrame to find the counties in each district
district_counties = [[population_df.loc[population_df['Index'] == i, 'Index'].iloc[0]
    for i in districts[j]] for j in range(k)]

# Summing the populations of the counties in each district
district_populations = [sum(population_df.loc[population_df['Index'] == i, 'Population'
    ].iloc[0] for i in districts[j]) for j in range(k)]

# Print district info
for j in range(k):
    print("District", j, "has population", district_populations[j], "and contains
        counties", district_counties[j])
```

Listing 8: Python code for initial results

```python
import geopandas as gpd
import matplotlib.pyplot as plt

# Load the shapefile
nm_counties_gdf = gpd.read_file('/Users/dannilin/Desktop/OR_Class/Final_Project/NM/maps/
    NM_centers.shp')  # Replace with the actual path

# FIPS code to county name mapping
fips_to_county = {
    '35001': 'BERNALILLO', '35003': 'CATRON', '35005': 'CHAVES', '35006': 'CIBOLA',
    # ... (remaining mappings)}

# Read the file and create the mapping dictionary
node_to_fips = {}
with open('/Users/dannilin/Desktop/OR_Class/Final_Project/NM/NM.hash', 'r') as file:
    for line in file:
        node_number, fips_code = line.strip().split()
        node_to_fips[int(node_number)] = fips_code

# Define the districts based on your model's output
districts = district_counties

# Create a DataFrame from the district assignments
assignment_df = pd.DataFrame([(node, district) for district, nodes in enumerate(
    districts) for node in nodes], columns=['node', 'district'])

# Map node numbers to FIPS codes
assignment_df['GEOID20'] = assignment_df['node'].apply(lambda x: node_to_fips.get(x,
    None))

# Add county names to the assignment DataFrame
assignment_df['CountyName'] = assignment_df['GEOID20'].map(fips_to_county)

# Merge the assignment DataFrame with the shapefile GeoDataFrame
merged_gdf = nm_counties_gdf.merge(assignment_df, on='GEOID20', how='left')

# Plot the map with district assignments and county names
fig, ax = plt.subplots(figsize=(10, 10))
merged_gdf.plot(column='district', ax=ax, categorical=True, legend=True, legend_kwds={'
    bbox_to_anchor': (1, 1)}, cmap='viridis')

# Annotate each county with its name
for idx, row in merged_gdf.iterrows():
    plt.annotate(text=row['CountyName'], xy=(row['geometry'].centroid.x, row['geometry'
        ].centroid.y),
                 horizontalalignment='center', fontsize=8)
plt.title("New Mexico Counties by District")
plt.show()
```

Listing 9: Python code for plotting outcome map

10 Results

10.1 Outputs

Below is the output graph for the New Mexico state as produced by our model:

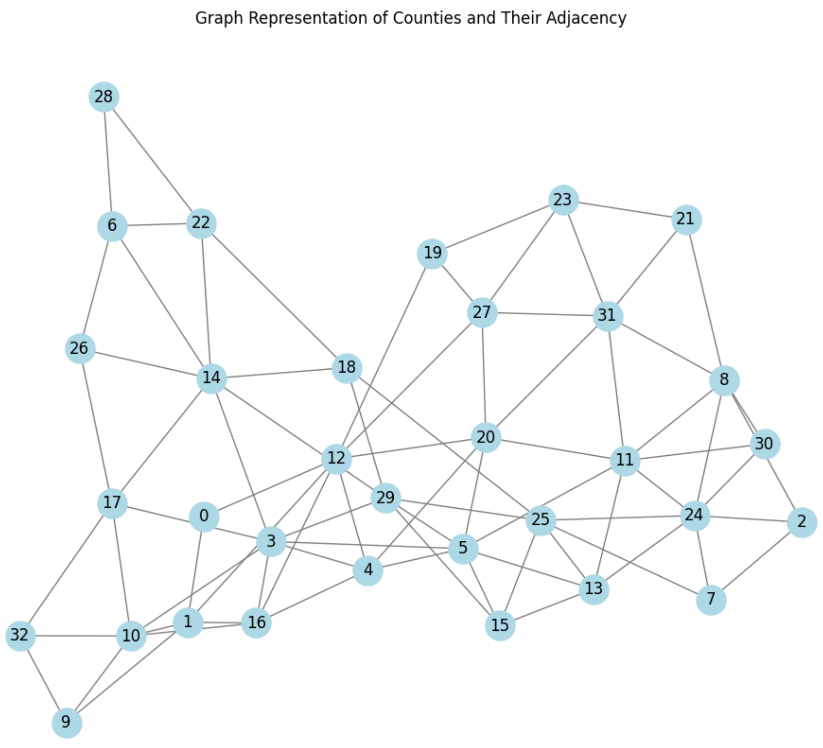

Figure 3: New Mexico State Output Graph

The following output is from the Gurobi Optimizer for our optimized model:

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (mac64[arm])

CPU model: Apple M1
Thread count: 8 physical cores, 8 logical processors, using up to 8 threads

Optimize a model with 486 rows, 432 columns and 1941 nonzeros
Model fingerprint: 0x6a3fbbcd
Variable types: 156 continuous, 276 integer (276 binary)
Coefficient statistics:
  Matrix range     [1e+00, 7e+05]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 7e+05]
Presolve time: 0.00s
Presolved: 486 rows, 432 columns, 1950 nonzeros
Variable types: 156 continuous, 276 integer (276 binary)

Root relaxation: objective 0.000000e+00, 192 iterations, 0.00 seconds (0.00 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0    0.00000    0  108          -    0.00000      -     -    0s
     0     0    3.63573    0  112          -    3.63573      -     -    0s
H    0     0                      33.0000000    3.63573  89.0%     -    0s
     0     0    6.71700    0   89   33.00000    6.71700  79.6%     -    0s
     0     0    8.22580    0  132   33.00000    8.22580  75.1%     -    0s
     0     0    8.22580    0  129   33.00000    8.22580  75.1%     -    0s
     0     0    8.52059    0  126   33.00000    8.52059  74.2%     -    0s
     0     0    8.57478    0  121   33.00000    8.57478  74.0%     -    0s
     0     0    8.58425    0  121   33.00000    8.58425  74.0%     -    0s
     0     0    8.80087    0  134   33.00000    8.80087  73.3%     -    0s
     0     0    8.90434    0  138   33.00000    8.90434  73.0%     -    0s
     0     0    8.98663    0  139   33.00000    8.98663  72.8%     -    0s
     0     0    8.98663    0  139   33.00000    8.98663  72.8%     -    0s
     0     0    9.40517    0  149   33.00000    9.40517  71.5%     -    0s
     0     0    9.41578    0   84   33.00000    9.41578  71.5%     -    0s
     0     0    9.41999    0   80   33.00000    9.41999  71.5%     -    0s
     0     0    9.45855    0  138   33.00000    9.45855  71.3%     -    0s
     0     0    9.45979    0   84   33.00000    9.45979  71.3%     -    0s
     0     0    9.46073    0  139   33.00000    9.46073  71.3%     -    0s
H    0     0                      19.0000000    9.46402  50.2%     -    0s
     0     0    9.46402    0  141   19.00000    9.46402  50.2%     -    0s
     0     0    9.48714    0  141   19.00000    9.48714  50.1%     -    0s
     0     0    9.50469    0  142   19.00000    9.50469  50.0%     -    0s
     0     0    9.50469    0  142   19.00000    9.50469  50.0%     -    0s
     0     0    9.51733    0  143   19.00000    9.51733  49.9%     -    0s
     0     0    9.51733    0  139   19.00000    9.51733  49.9%     -    0s
     0     0    9.51733    0   66   19.00000    9.51733  49.9%     -    0s
     0     0    9.51733    0  105   19.00000    9.51733  49.9%     -    0s
     0     0    9.82046    0  114   19.00000    9.82046  48.3%     -    0s
     0     0   10.24769    0  118   19.00000   10.24769  46.1%     -    0s
     0     0   10.38604    0  118   19.00000   10.38604  45.3%     -    0s
     0     0   11.66891    0  116   19.00000   11.66891  38.6%     -    0s
     0     0   11.66891    0  116   19.00000   11.66891  38.6%     -    0s
     0     2   11.66891    0  116   19.00000   11.66891  38.6%     -    0s
```

```
Cutting planes:
  Gomory: 6
  Cover: 3
  Implied bound: 2
  MIR: 30
  StrongCG: 1
  Flow cover: 5
  Mod-K: 1
  RLT: 31

Explored 329 nodes (12282 simplex iterations) in 0.50 seconds (0.30 work units)
Thread count was 8 (of 8 available processors)

Solution count 2: 19 33

Optimal solution found (tolerance 1.00e-04)
Best objective 1.900000000000e+01, best bound 1.900000000000e+01, gap 0.0000%
```

Below is the redistricting map output for New Mexico, as generated by our model:

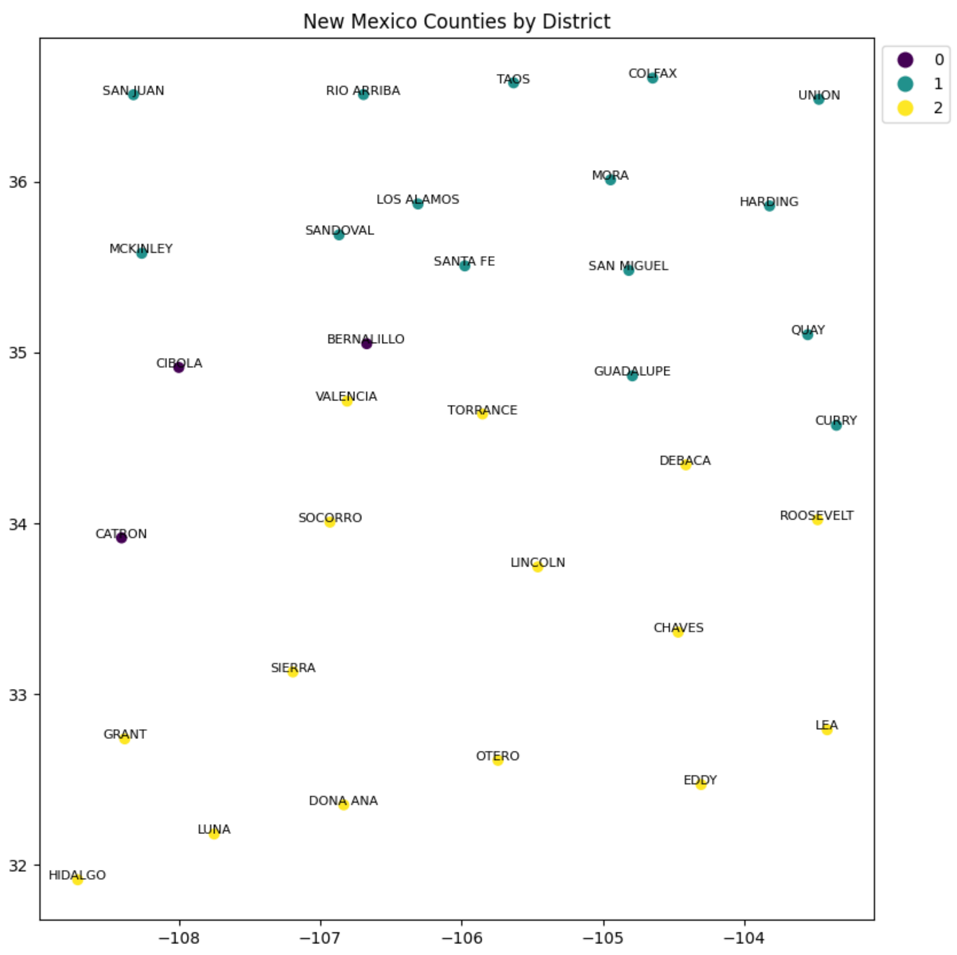

Figure 4: Redistricting Map Output for New Mexico

10.2 Experiments

Computer's Attributes

- **CPU Model**: Apple M1, integrates CPU, GPU, and Neural Engine.
- **Thread Count**: 8 physical cores and 8 logical processors, indicating a capability for efficient parallel task handling. Suitable setup for complex computational tasks, including optimization problems.

Optimization Solver

- **Solver**: Gurobi Optimizer, version 10.0.1.

Model Complexity

- The model consists of 486 rows, 432 columns, and 1941 nonzeros, with 156 continuous and 276 binary integer variables.

Optimization Results

- **Objective Value**: The model achieves an optimal objective value of 19.
- **Solution Time**: Not explicitly stated but likely within a reasonable timeframe since it was solved in less than 1 minutes (when our group tested the code).

Solution Quality and Bounds

- The gap of 0.0000% between the upper and lower bounds indicates that the solution found is optimal.

10.3 Redistricting Map Outcome

This the output map when done in New Mexico State Map. Our group colored by hand based on the generated result from Figure 4.

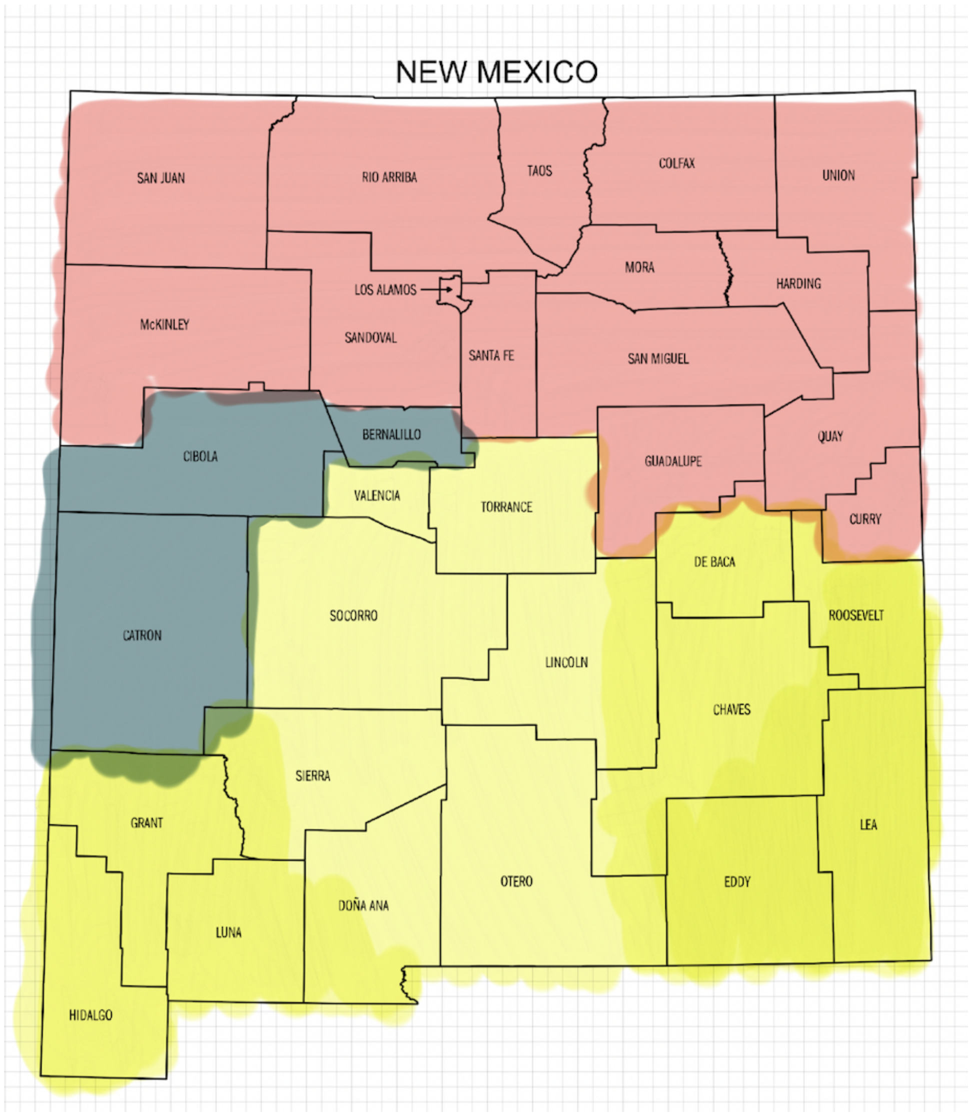

Figure 5: Redistricting Map Output for New Mexico

10.4 Evaluation

Analysis of Proposed Maps

The proposed redistricting maps generated by our optimization model focus on minimizing cut edges, aiming for geographically compact and contiguous districts.

Compactness

Our model's focus on minimizing cut edges aligns well with the requirement for compact districts. This approach helps in maintaining community integrity and ensuring equitable representation.

Contiguity

The implementation of single-commodity flow formulation in our model ensures that each district is contiguous, adhering to the contiguity criteria.

Preservation of Political Subdivisions and Communities of Interest

The model primarily emphasizes mathematical compactness and population equality. Therefore, it shows limitations in addressing the finer details of community and political boundaries, which is a key aspect of the required criteria.

Preserving the Cores of Prior Districts and Avoiding Pairing Incumbents

The model does not focus on preserving the cores of prior districts, aligning with the prohibition of this criteria. It also does not consider the locations of incumbents, thereby adhering to the criteria of avoiding pairing incumbents.

Results Comparison

From Validi et Al New Mexico outcome[8] and our work, it's possible to highlight 2 points for analysis:

Packages utilization

Our redistricting model was developed using NetworkX, a Python library for network analysis. NetworkX allowed us to efficiently build and analyze our graph-based model, which is instrumental in handling the complexities of redistricting. While Validi et Al study on New Mexico state redistricting utilized GerryChain, a Python library specifically designed for redistricting simulations. Therefore, there are other tools available to tackle the redistricting problem.

Maps

Our model was based on the 2020 dataset. In contrast, Validi et Al work was based on the 2010 dataset. When compared the outcomes, we can note that our model incorporated one more county (Catron) in a district. Finally none of the results was same as the official New Mexico redistricting map, mainly because both models didn't considered all of the criteria for redistricting.

Below is the output comparison, as depicted in the image 'comparison.png':

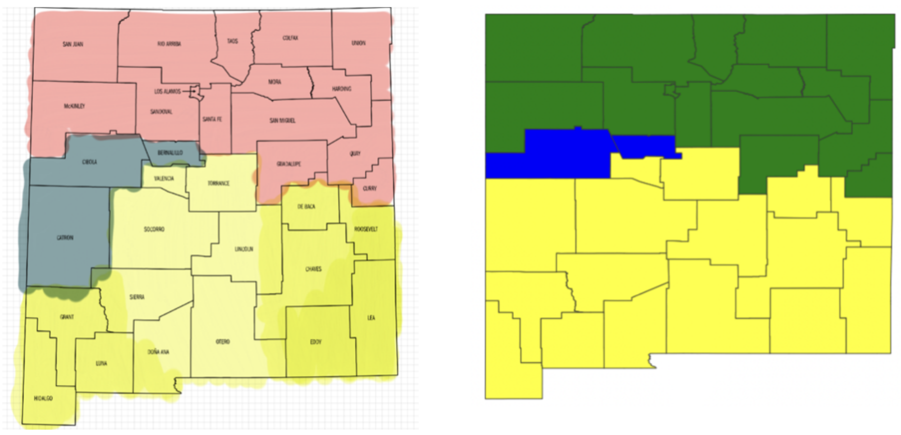

Figure 6: Output Comparison. Where the first image is from our work and the second is from Validi et Al work.

11 Conclusions

The comparison between our work and that of Validi et al. underscores significant differences in methodologies and datasets employed in both approaches. While both studies arrived at optimal solutions for redistricting, the choice of dataset plays a crucial role in influencing the outcome. The distinct datasets used reflect different redistricting landscapes.

In our model, while there is a strong alignment with the criteria of compactness and contiguity, and adherence to the prohibitions regarding preserving the cores of prior districts and avoiding pairing incumbents, there are notable limitations. The model exhibits challenges in comprehensively addressing the preservation of political subdivisions and communities of interest. This aspect is crucial in redistricting, as it ensures representation that reflects the true demographic and political makeup of an area. The need to incorporate more detailed data and methodologies is evident to enhance the model's capability in this problem.

Finally, the knowledge acquired from OR classes were useful when reading papers about problem modelling and our group was capable of finishing a redistricting problem.

References

[1] Redistricting, *Ballotpedia*, Available at: `https://ballotpedia.org/Redistricting`

[2] Why should we care?, *All about Redistricting*, Available at: `https://redistricting.lls.edu/redistricting-101/why-should-we-care/`

[3] Federal redistricting criteria, *Federal Redistricting Criteria*, Available at: `https://crsreports.congress.gov/product/pdf/IN/IN11618#:~:text=Under%20the%20%E2%80%9Cequality%20standard%E2%80%9D%20or,district%20population%20sizes%20can%20vary.`

[4] New Mexico redistricting criteria, *New Mexico redistricting criteria*, Available at: `https://www.ncsl.org/redistricting-and-census/redistricting-criteria`

[5] New Mexico congressional districts, *Iowa congressional districts*, Available at: `https://en.wikipedia.org/wiki/Iowa%27s_congressional_districts`

[6] New Mexico congressional districts, *2020 redistricting deviation tables*, Available at: `https://www.ncsl.org/elections-and-campaigns/2020-redistricting-deviation-table`

[7] New Mexico county population, *New Mexico county population*, Available at: `https://lykhovyd.com/files/public/districting/2020/NM/counties/graph/`

[8] H. Validi and A. Buchanan, *Political districting to minimize cut edges*, Computational and Applied Mathematics, Rice University, hamidreza.validi@rice.edu; Industrial Engineering & Management, Oklahoma State University, buchanan@okstate.edu.

[9] C. Hojny, I. Joormann, H. Lüthen, and M. Schmidt, *Mixed-integer programming techniques for the connected max-k-cut problem*, Mathematical Programming Computation, 13(1):75–132, 2021.