Arkansas Redistricting Final Report

28 April 2022

Team BAD

IEM 4013-24520

Dr. Austin Buchanan

Ali AlMulla, Landon Bakhsh, Logan Davis

EXECUTIVE SUMMARY

Political redistricting is an issue of great importance within the United States. With the country's strong belief in democratic practices, it is of utmost importance that political districts are designed fairly. This preserves the voting integrity of Americans and ensures a more equal society. This report will detail how Team BAD tackles the issue of political redistricting within the state of Arkansas.

Team BAD first analyzed federal and state redistricting criteria, then acquired data to assist in further development. A linear program was then developed to create a model to illustrate a potential districting plan for the state of Arkansas.

A redistricting plan was created based around districts having ~750,000 people per district. The population deviation of this plan is 0.28% and is true to the redistricting criteria put in place at the state and federal government levels.

INTRODUCTION

The United States Constitution requires that a census is taken every ten years. The results of this poll then determines whether each governing body of the federal and state governments must see redistricting, redelegation, or redistribution.

Redistricting is a term used to describe the process of taking a political body and grouping it into smaller, nearly equal population pieces. This is done to ensure there exists no bias in favor of specific demographics or spatial locations. Any given districting plan must follow redistricting criteria as outlined by both federal and state governments.

Due to different amounts of people living in different spatial locations in addition to the limitations posed by geographic, political, and social constraints, redistricting is an incredibly complex issue that is often left to optimization models to solve. This solution was created using the minimum cut edges technique.

REDISTRICTING CRITERIA

Federal Criteria

Every state within the United States of America must abide by the following criteria points:

- Each district must have a nearly equal population in relation to one another.
    - *Apportionment Clause, Article , Section 2, U.S. Constitution*
- Each district must not intentionally discriminate racially.
    - *Voting Rights Act, 1965*

These points are enforced at the federal level. In addition to this, each state has the freedom to impose their own criteria.

State of Arkansas Criteria

The state of Arkansas requires that the following criteria points are followed:

- Each district must maintain a +/- 5% population deviation.
- Each district must not intentionally discriminate or favor any race.
- Districts cannot be redrawn strictly based on race.

Source: https://arkansasredistricting.org/about-the-process/redistricting-criteria-and-goals/

Arkansas does not impose very many of its own constraints within the redistricting process. It is generally desirable that districts are compact and contiguous. It is also preferred that districts preserve counties and other political subdivisions, communities of interest, and prior districts. If it is at all possible, incumbents should not be paired within a district. Other considerations that are taken when selecting a districting plan include whether the plan favors or disfavors an incumbent, if the plan utilizes partisan data, or if the plan is generally competitive.

PROBLEM STATEMENT

Team BAD will be utilizing operations research methodology to develop a districting plan for the state of Arkansas that follows the constraints imposed by the federal and state government.

OPERATIONS RESEARCH (OR) MODEL (PROCESS 1)

The models for Process 1 and 2 are very similar. For that reason we will only cover the model which produced the best redistricting map.

The goal of this process is to minimize the distance of the cut edges in order to ensure compactness for each district. Counties will be prioritized to be kept whole for the process.

Sets: $C\ is\ the\ set\ of\ counties\ (Nodes)\ 1, 2,..., 74$

$J\ is\ the\ set\ of\ districts\ 1, 2,..., k$

$E\ is\ the\ set\ of\ edges\ 1, 2,..., e$

$N(i)\ is\ the\ set\ of\ neighbors\ of\ county\ (i)$

Indices: $i\ represents\ a\ county$

$j\ represents\ a\ district$

$u\ and\ v\ are\ counties\ that\ are\ being\ checked\ for\ contiguity$

Parameters: $P_i = population\ of\ county\ i$

$totpop_j = total\ population\ of\ district\ j$

$e = total\ number\ of\ edges$

$n = number\ of\ counties\ (nodes)$

$k = total\ number\ of\ desired\ districts$

$L = lower\ bound\ maximum\ deviation$

$U = upper\ bound\ maximum\ deviation$

$M = number\ of\ counties - number\ of\ districts + 1$

Variables: $x_{ij} = \{ 1\ county\ is\ a\ part\ of\ district\ j$

$\{ 0\ Otherwise$

$$y_{uv} = \begin{cases} 1 \; there\ is\ a\ district\ boundary\ between\ county\ u\ and\ v \\ 0 \; Otherwise \end{cases}$$

$$r_{ij} = \begin{cases} 1 \; county\ i\ is\ the\ root\ of\ district\ j \\ 0 \; Otherwise \end{cases}$$

$$f_{ij} = \begin{cases} 1 \; flow\ is\ being\ sent\ from\ i\ to\ j \\ 0 \; Otherwise \end{cases}$$

In Words	**In Math**
Minimize cut edges	Min $\sum_{u}^{n}\sum_{v}^{n} y_{uv}$
Each county i is assigned to a district j	S.T. $\sum_{i}^{n} x_{ij} = 1 \mid \forall_j \in J$
Each district j has a population at most U	$\sum_{i}^{n} P_i x_{ij} \leq U \mid \forall_j \in J$
Each district j has a population at least L	$\sum_{i}^{n} P_i x_{ij} \geq L \mid \forall_j \in J$
An edge is cut if u is assigned to district j but v is not.	$x_{uj} - x_{vj} \leq y_{uv} \mid \forall_u \in E, \forall_v \in E, \forall_j \in J$
Each districts should have one root	$\sum_{i}^{n} r_{ij} = 1 \mid \forall_j \in J$
If node i isn't assigned to district j, then it cannot be its root	$r_{ij} \leq x_{ij} \mid \forall_j \in J, \forall_i \in C$
Only send flow to a root	$\sum_{j \in N(i)} (f_{ij} - f_{ji}) \leq 1 - M \times \sum_{j}^{k} r_{ij} \mid \forall_i \in C$
Do not send flow across cut edges	$f_{ij} + f_{ji} \leq M \times (1 - y_{ij}) \mid \forall_i \in E, \forall_j \in E$

PYTHON/GUROBI CODE

Process 1: Minimum Cut Edges

1. **Importing Packages**

```
In [85]: #import all necessary packages
         import gurobipy as gp
         from gurobipy import GRB
         from gerrychain import Graph
         import networkx as nx
         import geopandas as gpd
         import math
```

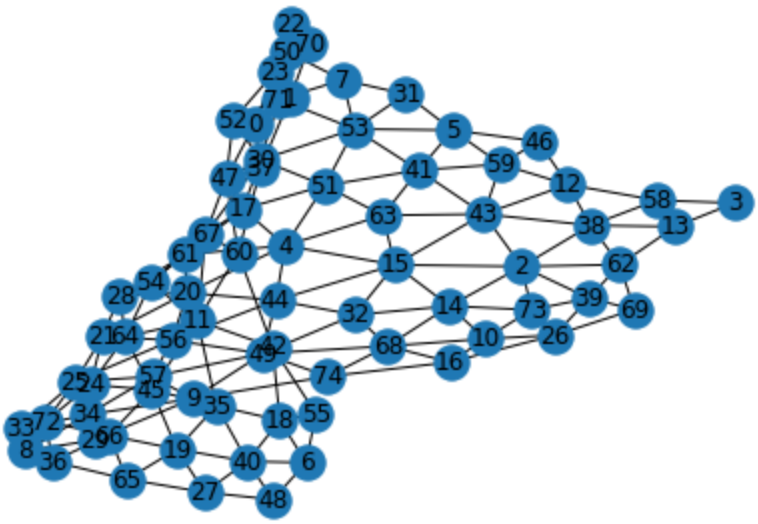
2. **Contiguity**

After importing packages, we call AR_County.json. This file has the necessary data to create a graph of all the nodes.

```
In [86]: #Set filepath and filename equal to the path/name of the data used respectively
         filepath = 'C:/Users/Logan/Desktop\College/IEM4013Project/'
         filename= 'AR_county.json'
         #Create a new Graph object G from the file
         G = Graph.from_json(filepath + filename)

In [87]: #Set each node in G to be equal to the population of their respective county
         for node in G.nodes:
             G.nodes[node]['TOTPOP'] = G.nodes[node]['P0010001']
```

```
In [89]: #draw the graph of nodes
         nx.draw(G, with_labels=True)
```



From this data, we can find which counties touch. This can then be represented as a node-and-edge graph to better visualize the data. The nodes are then labeled so it is easier to discern which county is within which district.

```
In [117]: #Print each node, the county it represents, and their 2020 population
          for node in G.nodes:
              name = G.nodes[node]['NAME20']
              population = G.nodes[node]['TOTPOP']
              print("Node",node,"represents",name,"County with 2020 population of",population)
```

```
Node 0 represents Franklin County with 2020 population of 17097
Node 1 represents Crawford County with 2020 population of 60133
Node 2 represents Jackson County with 2020 population of 16755
Node 3 represents Clay County with 2020 population of 14552
Node 4 represents Faulkner County with 2020 population of 123498
Node 5 represents Baxter County with 2020 population of 41627
Node 6 represents Little River County with 2020 population of 12026
Node 7 represents Boone County with 2020 population of 37373
Node 8 represents Ashley County with 2020 population of 19062
Node 9 represents Desha County with 2020 population of 11395
Node 10 represents St. Francis County with 2020 population of 23090
Node 11 represents Montgomery County with 2020 population of 8484
```

3. Create the Model

In [120]:
```
#create a new model object and create variables
m = gp.Model()

x = m.addVars(G.nodes, k, vtype=GRB.BINARY)
y = m.addVars(G.edges, vtype=GRB.BINARY)
```

In [121]:
```
#set objective to minimize cut edges
m.setObjective( gp.quicksum( y[u,v] for u,v in G.edges ), GRB.MINIMIZE )
```

In [122]:
```
# each county i is assigned to a district j
m.addConstrs(gp.quicksum(x[i,j] for j in range(k)) == 1 for i in G.nodes)
# each district j has a population at least L and at most U
m.addConstrs( gp.quicksum( G.nodes[i]['TOTPOP'] * x[i,j] for i in G.nodes) >= L for j in range(k))
m.addConstrs( gp.quicksum( G.nodes[i]['TOTPOP'] * x[i,j] for i in G.nodes) <= U for j in range(k))
# an edge is cut if u is assigned to district j but v is not.
m.addConstrs( x[u,j] - x[v,j] <= y[u,v] for u,v in G.edges for j in range(k))
m.update()
```

In [123]:
```
# add root variables: r[i,j] equals 1 if node i is the root of district j
r = m.addVars( G.nodes, k, vtype=GRB.BINARY)

import networkx as nx

DG = nx.DiGraph(G)

f = m.addVars(DG.edges)
```

In [124]:
```
# The big-M proposed by Hojny et al.
M = G.number_of_nodes() - k + 1
# each district should have one root
m.addConstrs( gp.quicksum( r[i,j] for i in G.nodes ) == 1 for j in range(k) )
# If node i isn't assigned to district j, then it cannot be its root
m.addConstrs( r[i,j] <= x[i,j] for i in G.nodes for j in range(k) )
# If not a root, consume some flow
# If a root, only send out (so much) flow
m.addConstrs( gp.quicksum( f[j,i] - f[i,j] for j in G.neighbors(i) )
             >= 1 - M * gp.quicksum( r[i,j] for j in range(k) ) for i in G.nodes )
# Do not send flow across cut edges
m.addConstrs( f[i,j] + f[j,i] <= M * (1-y[i,j] )for i,j in G.edges)

m.update()
```

Each constraint is labeled. Constraints are taken from the OR model previously detailed in this report. The objective is to minimize the amount of cut edges, meaning that we want the fewest number of counties split into two separate districts.

4. Optimize

```
 157392 12347   11.94760   37  167   12.00965   11.11272  7.47%  87.6  195s
 163365 11253   11.89880   38  205   12.00965   11.17020  6.99%  87.1  200s
 169906  9748   11.78766   32  304   12.00965   11.24342  6.38%  86.5  205s
 176265  8011     cutoff   32        12.00965   11.32273  5.72%  85.9  210s
 182575  5808     cutoff   39        12.00965   11.42010  4.91%  85.3  215s
 190816  1592     cutoff   35        12.00965   11.63483  3.12%  84.2  220s

Cutting planes:
  Gomory: 16
  Cover: 2
  MIR: 2
  Flow cover: 16
  Zero half: 5
  RLT: 24

Explored 194359 nodes (16385690 simplex iterations) in 221.99 seconds (257.78 work units)
Thread count was 12 (of 12 available processors)

Solution count 4: 12.0096 12.7364 12.8332 12.8994

Optimal solution found (tolerance 1.00e-04)
Best objective 1.200964554654e+01, best bound 1.200964554654e+01, gap 0.0000%
```

The model was then optimized. The statistics for this optimization can be seen above. 194,259 nodes were explored. 16,385,690 simplex iterations were computed. The process took 221.99 seconds.

Process 2: Min Moment of Inertia

Many of the steps in Process 2 are similar to Process 1. Repeated processes will be skipped.

1. **Setup**

```
In [20]: #Set filepath and filename equal to the path/name of the data used respectively
filepath = 'C:/Users/Logan/Desktop\College/IEM4013Project/'
filename= 'AR_county.json'

#Create a new Graph object G from the file
G = Graph.from_json(filepath + filename)

#
for node in G.nodes:
    G.nodes[node]['TOTPOP'] = G.nodes[node]['P0010001']
    G.nodes[node]['C_X'] = G.nodes[node]['INTPTLON20']
    G.nodes[node]['C_Y'] = G.nodes[node]['INTPTLAT20']
```

More information is taken from AR_county.json.

2. **Distance Dictionary**

```
In [21]: # create distance dictionary
dist = { (i,j) : 0 for i in G.nodes for j in G.nodes }
for i in G.nodes:
    for j in G.nodes:
        loc_i = ( G.nodes[i]['C_Y'], G.nodes[i]['C_X'] )
        loc_j = ( G.nodes[j]['C_Y'], G.nodes[j]['C_X'] )
        dist[i,j] = geodesic(loc_i,loc_j).miles
```

A dictionary is created for each node's distance from one another. This will be used in the moment of inertia calculation.

3. Create Model

In [23]:
```
m = gp.Model()

x = m.addVars(G.nodes, G.nodes, vtype=GRB.BINARY)
```

In [24]:
```
m.setObjective( gp.quicksum( dist[i,j] * dist[i,j] * G.nodes[i]['TOTPOP'] * x[i,j] for i in G.nodes for j in G.nodes ), GRB.MINI
```

In [25]:
```
# add constraints saying that each county i is assigned to one district
m.addConstrs( gp.quicksum( x[i,j] for j in G.nodes ) == 1 for i in G.nodes )

# add constraint saying there should be k district centers
m.addConstr( gp.quicksum( x[j,j] for j in G.nodes ) == k )

# add constraints that say: if j roots a district, then its population is between L and U.
m.addConstrs( gp.quicksum( G.nodes[i]['TOTPOP'] * x[i,j] for i in G.nodes ) >= L * x[j,j] for j in G.nodes )
m.addConstrs( gp.quicksum( G.nodes[i]['TOTPOP'] * x[i,j] for i in G.nodes ) <= U * x[j,j] for j in G.nodes )

# add coupling constraints saying that if i is assigned to j, then j is a center.
m.addConstrs( x[i,j] <= x[j,j] for i in G.nodes for j in G.nodes )

m.update()
```

In [26]:
```
# add contiguity constraints
import networkx as nx
DG = nx.DiGraph(G)

# add flow variables
#    f[i,j,v] = flow across arc (i,j) that is sent from souce/root v
f = m.addVars( DG.edges, G.nodes )

# add constraints saying that if node i is assigned to node j,
#   then node i must consume one unit of node j's flow
m.addConstrs( gp.quicksum( f[u,i,j] - f[i,u,j] for u in G.neighbors(i) ) == x[i,j] for i in G.nodes for j in G.nodes if i != j )

# add constraints saying that node i can receive flow of type j
#   only if node i is assigned to node j
M = G.number_of_nodes() - 1
m.addConstrs( gp.quicksum( f[u,i,j] for u in G.neighbors(i) ) <= M * x[i,j] for i in G.nodes for j in G.nodes if i != j )

# add constraints saying that node j cannot receive flow of its own type
m.addConstrs( gp.quicksum( f[u,j,j] for u in G.neighbors(j) ) == 0 for j in G.nodes )

m.update()
```

Constraints are labeled in the image above, and taken from the OR Model section of this paper. Notably, in this version of the optimization we are minimizing the Moment of Inertia. This focuses on keeping the districts compact while also balancing district population.

4. **Optimize**

```
     0     0 6.7966e+09    0  247 6.8351e+09 6.7966e+09  0.56%     -    9s
     0     0 6.7992e+09    0  247 6.8351e+09 6.7992e+09  0.53%     -   10s
     0     2 6.7992e+09    0  247 6.8351e+09 6.7992e+09  0.53%     -   10s

Cutting planes:
  Gomory: 1
  Lift-and-project: 2
  Cover: 31
  Implied bound: 4
  Clique: 1
  MIR: 9
  StrongCG: 2
  Flow cover: 47
  GUB cover: 10
  Network: 4
  RLT: 2

Explored 75 nodes (15169 simplex iterations) in 10.82 seconds (7.26 work units)
Thread count was 12 (of 12 available processors)

Solution count 3: 6.8351e+09 6.88348e+09 6.99888e+09

Optimal solution found (tolerance 0.00e+00)
Best objective 6.835095216711e+09, best bound 6.835095216711e+09, gap 0.0000%
```

Model optimization led to an improved solution time of 10.82 seconds. Only 75 nodes and 15,169 simplex iterations were explored. Further statistics are shown above.

5. **Additional information**

An additional method of optimization was computed using a minimum perimeter. However, this produced the same map as the min cut edges method so it was excluded from the final paper.

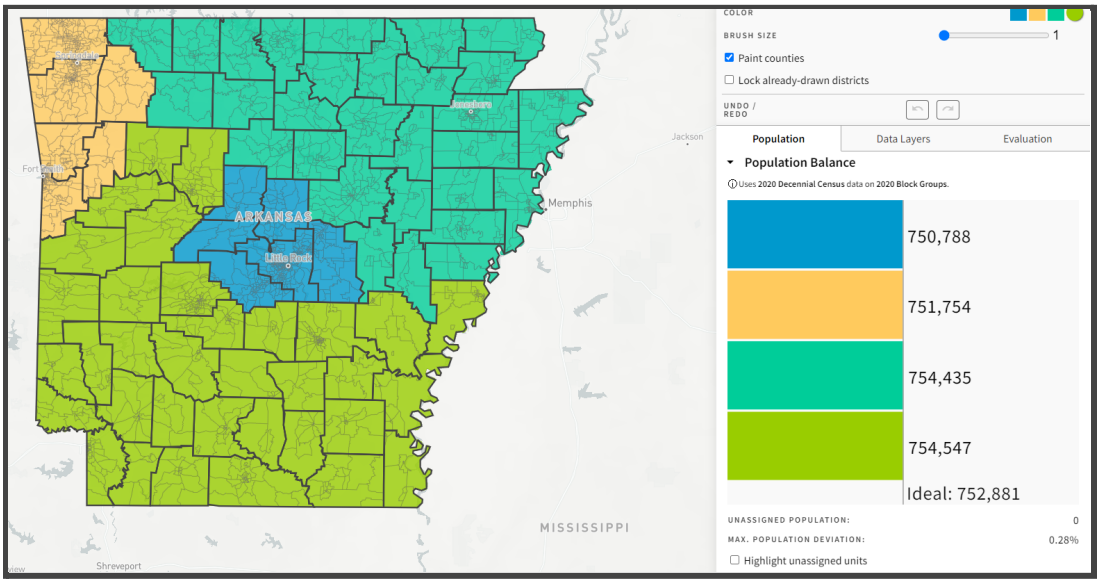
EXPERIMENTS

The program was written and optimized using the Gurobi Application (9.5.0) and Jupyter Notebook coded in Python. The program was solved on a Home Built PC which contains 32 GB of RAM, and an Intel(R) Core(TM) i5-10600K CPU @ 4.10GHz processor. The two programs were solved in 221.99 and 10.82 seconds respectively.

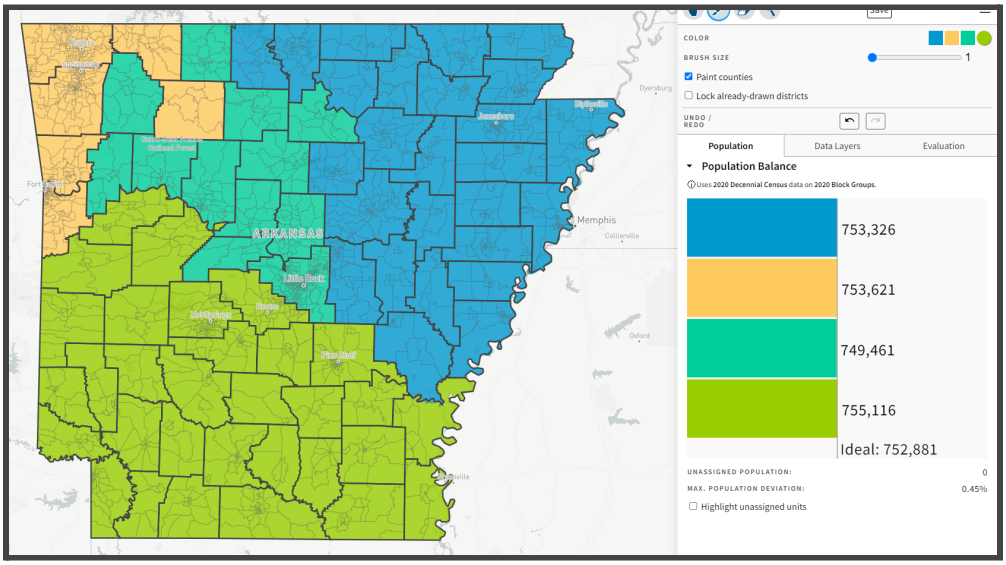
PLANS AND MAPS

Below you can find the two plans that were created in the two programs.

Process 1: Min Cut Edges
https://districtr.org/plan/126891



Process 2: Min Moment of Inertia
https://districtr.org/plan/126890



EVALUATION OF PLANS

Team BAD created two new district plans for Arkansas. The first plan utilized minimum cutting edges and got the maximum population deviation of 0.28%. Process two utilized the minimum moment of inertia. The maximum population deviation was determined to be 0.45%. Process one is recommended because it covers all federal and state criteria.

CONCLUSIONS

In conclusion, Team BAD first saw the federal and state redistricting criteria then gathered data to develop a linear program illustrating potential districting plans for the state of Arkansas. Team BAD covered the criteria presented in the proposed districting plan. It was determined that each district has a population of 750,000 people. Here it is mentioned the population for each district with the which counties it covers:

- District 1 has population 751754 and contains counties ['Franklin', 'Crawford', 'Benton', 'Madison', 'Sebastian', 'Washington']
- District 2 has a population of 754435 and contains counties ['Jackson', 'Clay', 'Baxter', 'Boone', 'St. Francis', 'Sharp', 'Greene', 'Woodruff', 'White', 'Lee', 'Crittenden', 'Marion', 'Prairie', 'Lawrence', 'Poinsett', 'Stone', 'Independence', 'Fulton', 'Carroll', 'Van Buren', 'Searcy', 'Randolph', 'Izard', 'Craighead', 'Cleburne', 'Monroe', 'Mississippi', 'Newton', 'Cross']
- District 3 has population 750788 and contains counties ['Faulkner', 'Conway', 'Pulaski', 'Saline', 'Lonoke', 'Perry']
- District 4 has population 754547 and contains counties ['Little River', 'Ashley', 'Desha', 'Montgomery', 'Howard', 'Nevada', 'Grant', 'Dallas', 'Cleveland', 'Lafayette', 'Chicot', 'Pope', 'Bradley', 'Drew', 'Pike', 'Union', 'Hempstead', 'Polk', 'Clark', 'Logan', 'Miller', 'Arkansas', 'Johnson', 'Garland', 'Sevier', 'Jefferson', 'Lincoln', 'Scott', 'Hot Spring', 'Columbia', 'Ouachita', 'Yell', 'Calhoun', 'Phillips'].

The districts mapped are contiguous according to the guidelines. The maximum population deviation of 0.28% from the min cut edge method, tells us that the population is close to equivalent.

REFERENCES

Github: https://github.com/logandavis2518/IEM4013_2020RedistrictingProject.git

Data Source: https://github.com/AustinLBuchanan/Districting-Examples-2020

Redistricting Sources:
https://arkansasredistricting.org/about-the-process/redistricting-criteria-and-goals/

Voting Rights Act, 1965

Apportionment Clause, Article , Section 2, U.S. Constitution