

# **Fast Population Maximization in Designating Areas of Substantial Unemployment (ASUs) in Utah with A\* Search and Binary Integer Programming**

Joyce Cao (University of Utah)  
Advisor: Jeff Phillips (University of Utah)

# Designation of ASUs

- **Input**

- Graph  $G = (V, E)$
- Each vertex has population, employment, unemployment

- Threshold = 6.5%

Threshold = 0.0645 for  
rounding to thousandths

- **Problem Definition**

- Maximize  $\sum_{i=1}^k p_i$  , here  $p_i$  is the  $i$ -th ASU's population
- s.t. each ASU is contiguous
- the unemployment rate of each ASU  $\geq 6.5\%$

- **Motivation**

- Federal Funding is based on the total population of the  $k$  ASUs

# Dataframe

- **Data**
  - Shape Data (tl\_2016\_49\_tract.shp)
  - Attribute Data (UT\_asu\_exampleData.csv)
- **Dataframe**

Index	...	Geoid	Geometry	EMP	UNEMP	POP
0		49057210900	<pysal obj>	3638	77	6813
1		49045131200	<pysal obj>	1393	75	3515
2		49045131100	<pysal obj>	4053	205	7987
3		49045130600	<pysal obj>	1241	37	2704

# Introduction to Linear Programming

## Example from Wikipedia:

Index	Name	Fertilizer $kg/km^2$	Pesticide $kg/km^2$	Price $$/km^2$	Decision Var $km^2$
1	Wheat	F1	P1	S1	X1
2	Barley	F2	P2	S2	X2
Farmer has:		F (kg)	P (kg)		L( $km^2$ )

- A farmer has a land of size L for Wheat and Barley ... ([https://en.wikipedia.org/wiki/Linear\\_programming](https://en.wikipedia.org/wiki/Linear_programming))
- Step 2: Identify objective
  - Farmer wants to maximize the revenue
  - Maximize  $S_1x_1 + S_2x_2$
- Step 1: Identify decision variables
  - X1 the area of land planted with wheat
  - X2 the area of land planted with barley
- Step 3: Identify constraints
  - Area constraints:  $x_1 + x_2 \leq L$   
 $x_1, x_2 \geq 0$
  - Fertilizer limit:  $F_1x_1 + F_2x_2 \leq F$
  - Pesticide limit:  $P_1x_1 + P_2x_2 \leq P$

# BIP Step 1: Identify Decision Variables

Index	...	Geoid	Geometry	EMP	UNEMP	POP
0		49057210900	<pysal obj>	3638	77	6813
1		49045131200	<pysal obj>	1393	75	3515
2		49045131100	<pysal obj>	4053	205	7987
3		49045130600	<pysal obj>	1241	37	2704

- For each census tract in the dataframe, we need to make a decision.

- Do we choose census tract 0?
  - Do we choose census tract 1?

- In Math,

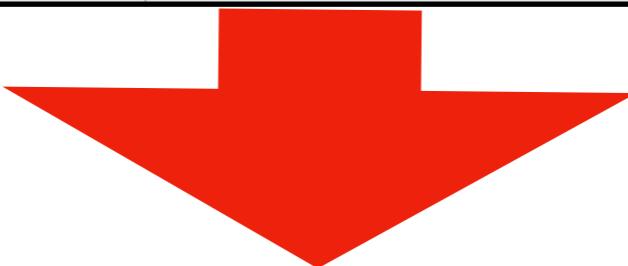
$$x_0 = \begin{cases} 1 & \text{if we decide to choose tract 0} \\ 0 & \text{otherwise} \end{cases}$$

Decision Variable

$$x_1 = \begin{cases} 1 & \text{if we decide to choose tract 1} \\ 0 & \text{otherwise} \end{cases}$$

# BIP Step 1: Identify Decision Variables (cont.)

Index	...	Geoid	Geometry	EMP	UNEMP	POP
0		49057210900	<pysal obj>	3638	77	6813
1		49045131200	<pysal obj>	1393	75	3515
2		49045131100	<pysal obj>	4053	205	7987
3		49045130600	<pysal obj>	1241	37	2704



Decision Number	Yes or No (1 or 0)	Decision Variables	EMP	UNEMP	POP
0	Choose 0?	$x_0$	3638	77	6813
1	Choose 1?	$x_1$	1393	75	3515
2	Choose 2?	$x_2$	4053	205	7987
3	Choose 3?	$x_3$	1241	37	2704

# BIP Step 2: Identify Objective

Decision Number	Yes/No (1 or 0)	Decision Variables	EMP	UNEMP	POP
0	<b>Choose 0?</b>	$x_0$	<b>3638</b>	<b>77</b>	<b>6813</b>
1	<b>Choose 1?</b>	$x_1$	<b>1393</b>	<b>75</b>	<b>3515</b>
2	<b>Choose 2?</b>	$x_2$	<b>4053</b>	<b>205</b>	<b>7987</b>
3	<b>Choose 3?</b>	$x_3$	<b>1241</b>	<b>37</b>	<b>2704</b>

- We want to maximize the total population of these decision variables
- In math, maximize  $6813x_0 + 3515x_1 + 7987x_2 + 2704x_3$

If this decision variable is 1, then the population is 7,987

If this decision variable is 0, then the population is 0

# BIP Step 3: Identify Constraints (1)

Decision Number	Yes/No (1 or 0)	Decision Variables	EMP	UNEMP	POP
0	<b>Choose 0?</b>	$x_0$	<b>3638</b>	<b>77</b>	<b>6813</b>
1	<b>Choose 1?</b>	$x_1$	<b>1393</b>	<b>75</b>	<b>3515</b>
2	<b>Choose 2?</b>	$x_2$	<b>4053</b>	<b>205</b>	<b>7987</b>
3	<b>Choose 3?</b>	$x_3$	<b>1241</b>	<b>37</b>	<b>2704</b>

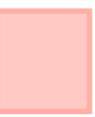
- The average unemployment rate of an ASU should be  $\geq 6.5\%$ .
- In Math,

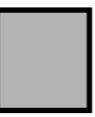
$$\frac{x_0 77 + x_1 75 + x_2 205 + x_3 37}{x_0(3638 + 77) + x_1(75 + 1393) + x_2(205 + 4053) + x_3(37 + 1241)} \geq 6.5\%$$

# BIP Step 3: Identify Constraints (2)

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

**Seed (tract added)** 

**1st-order Neighbor** 

**2nd-order Neighbor** 

**Radius = 2**

- An ASU should be a contiguous area.
- The way we do: if we decide to choose a tract only if at least one of its previous order neighbors has been chosen.
- Take  $x_{21}$  for example
- In Math,  $x_{21} \leq x_{16} + x_{17} + x_{22}$

The real contiguous constraint is as long as there is a path from the tract to the seed

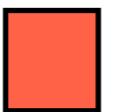
For example, when  $x_{21} = 1$ , at least one of  $x_{16}, x_{17}, x_{22} = 1$  such that the inequality is valid

# BIP Put Everything Together

- First, we construct a region around the seed with a specified radius

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Seed (tract added)



1st-order Neighbor



2nd-order Neighbor



Radius = 2

- Second, we formulate the BIP model

$$\begin{aligned} & \text{maximize } \sum_{i=1}^{25} p_i x_i \\ \text{s.t. } & \frac{\sum_{i=1}^{25} u_i x_i}{\sum_{i=1}^{25} (u_i + e_i) x_i} \geq 0.0645 \end{aligned}$$

$p_i$  : pop of the  $i$ -th tract

$u_i$  : unemp of the  $i$ -th tract

$e_i$  : emp of the  $i$ -th tract

$c$  : order of neighborhood

$q$  : queen neighbors

$$x_{12} = 1$$

$$x_{13} = 1$$

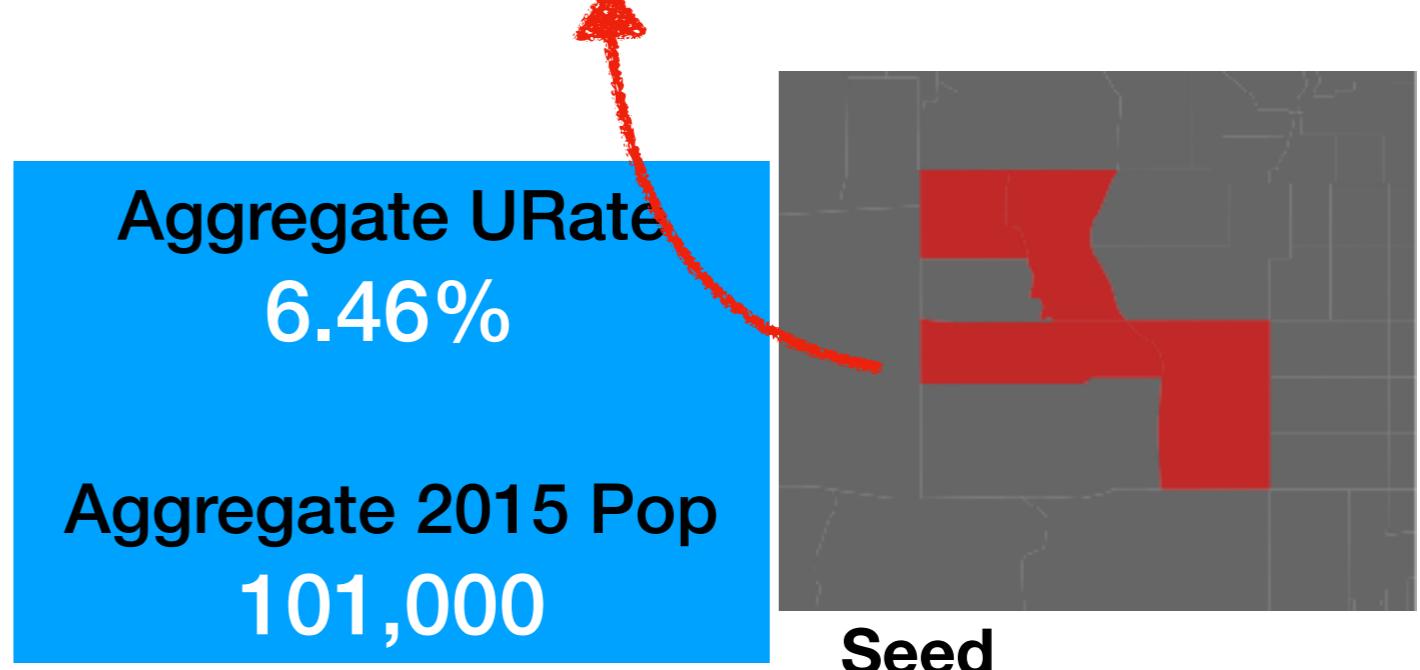
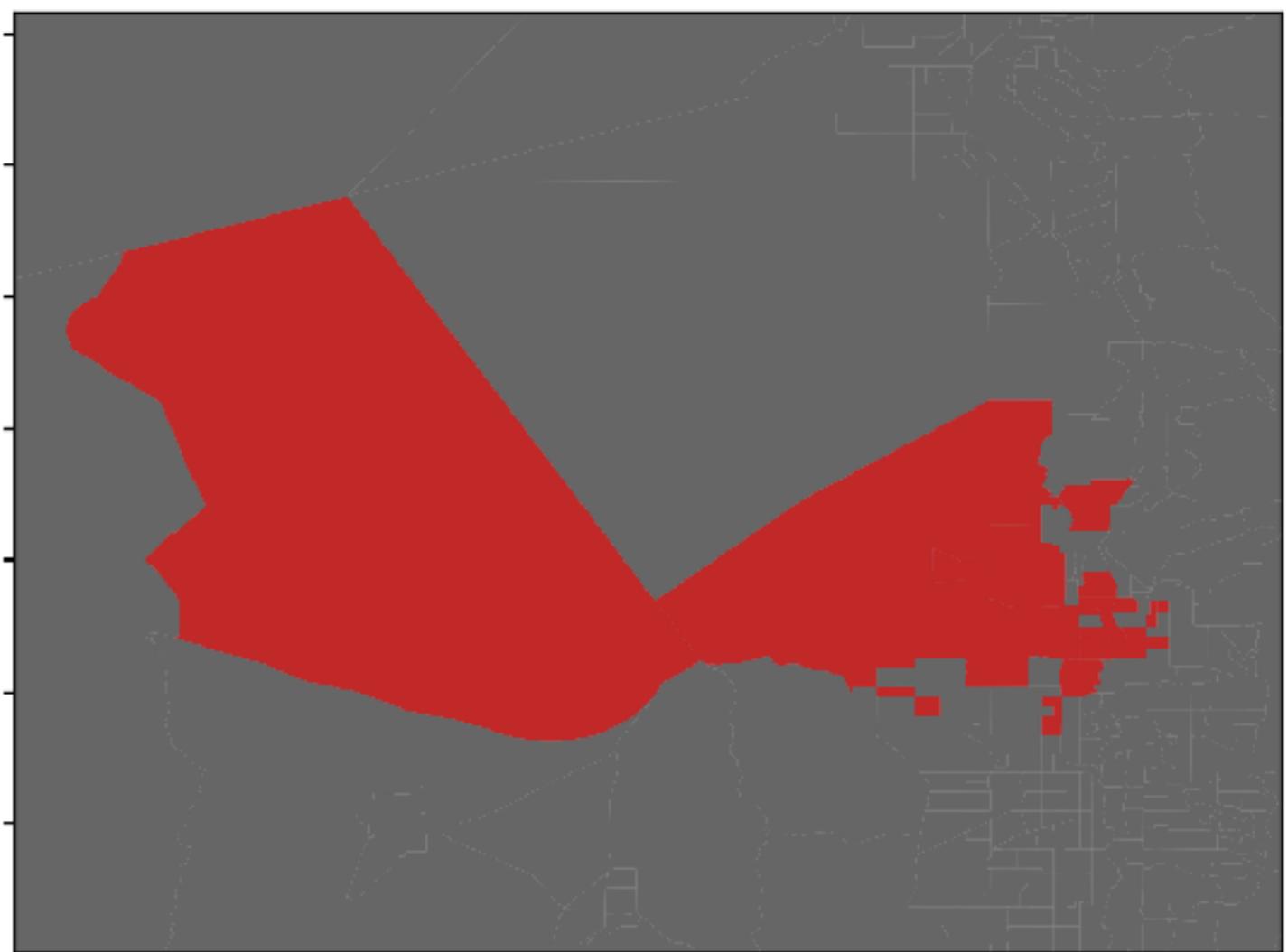
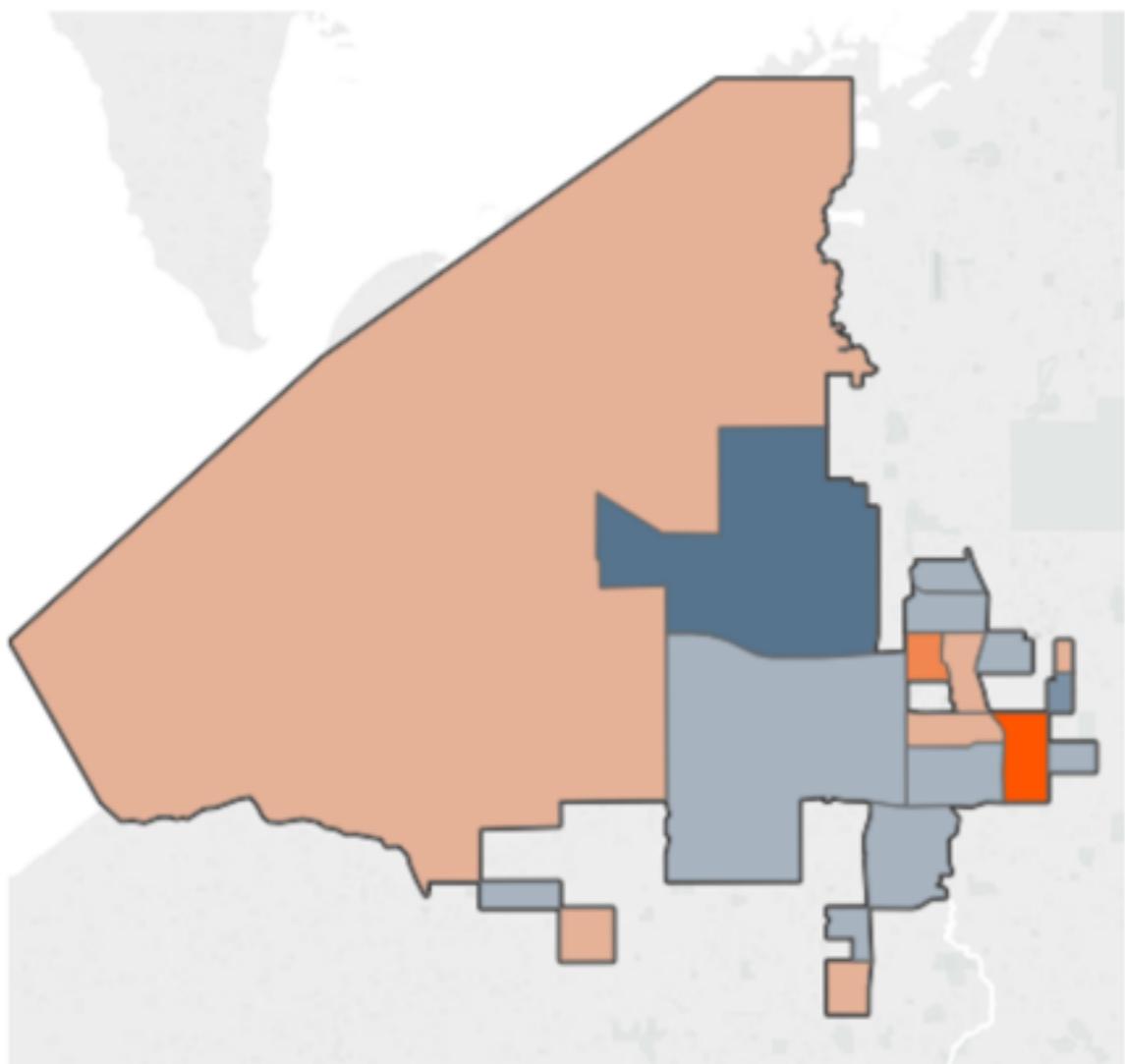
$$x_{18} = 1$$

$$x_i^c \leq \sum_q x_j^{c-1}$$

$x_i$  = binary

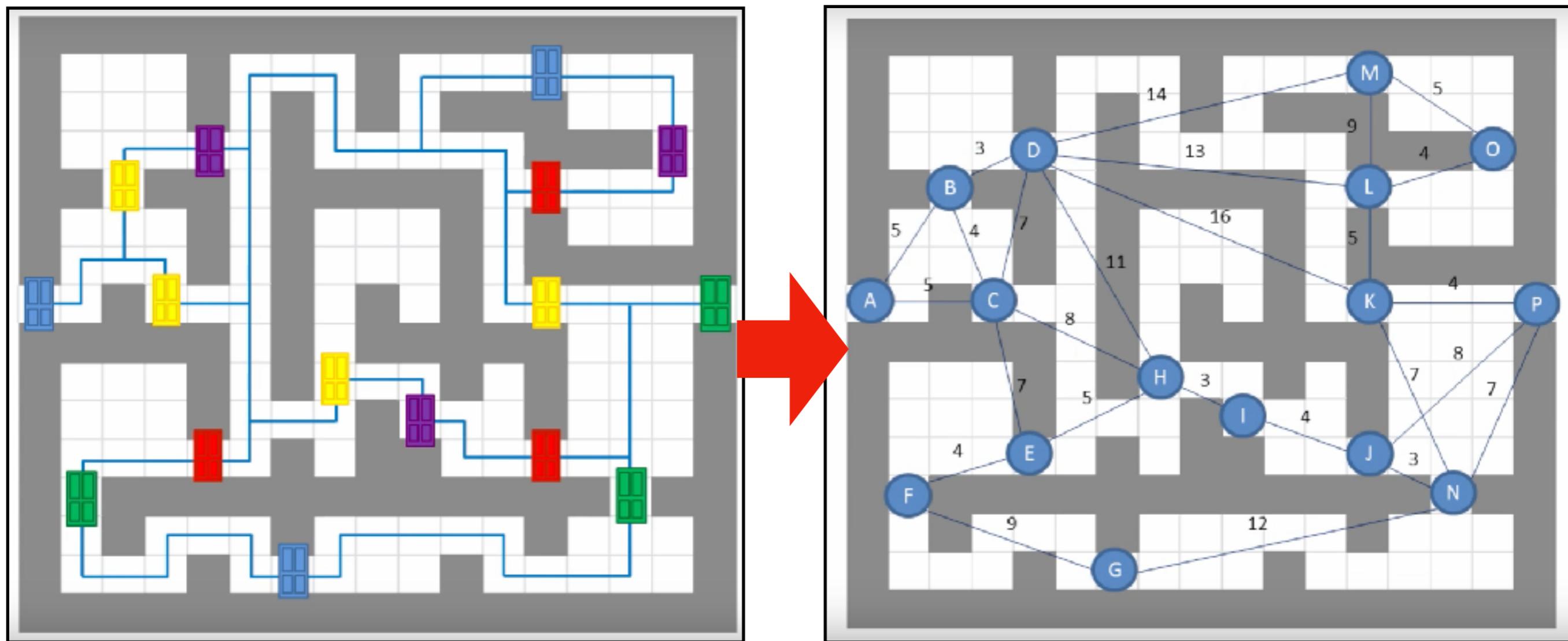
- Third, we use Python and PuLP to solve it.

# Marginal Results - BIP



# Introduction to A\*

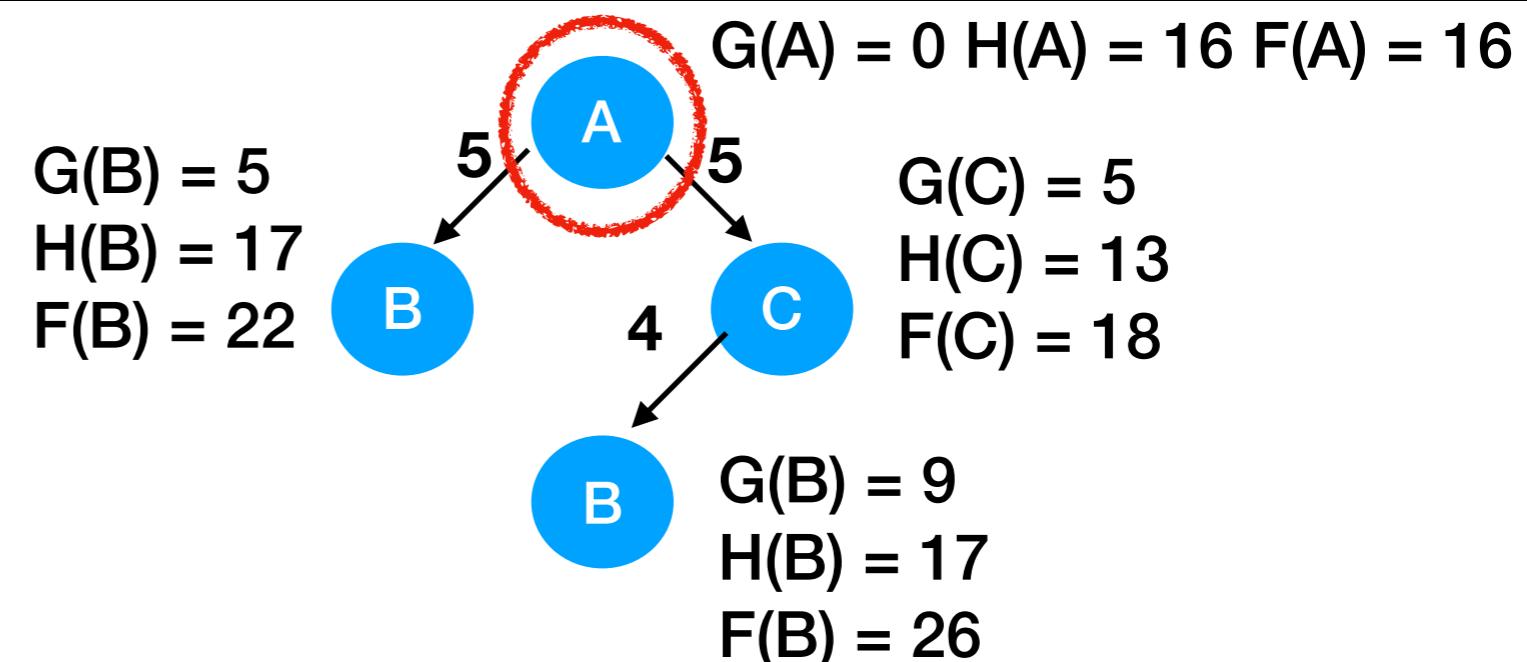
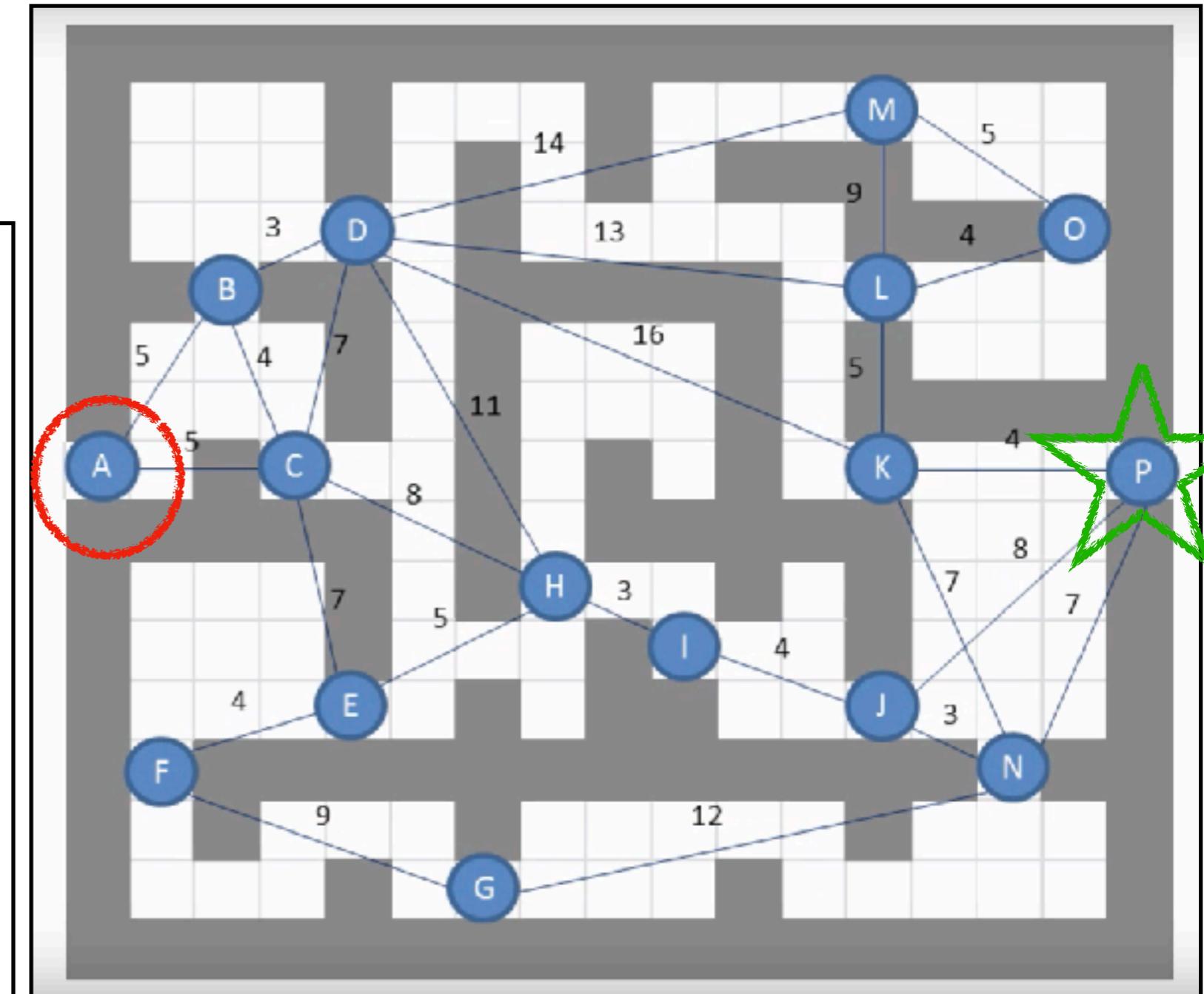
- A\* search algorithm was invented by Hart, Nilsson and Raphael in 1968 to find paths through graphs [Hart et al. 1968].
- Example:
  - Navigation through a maze [Kevin Drumm 2017]



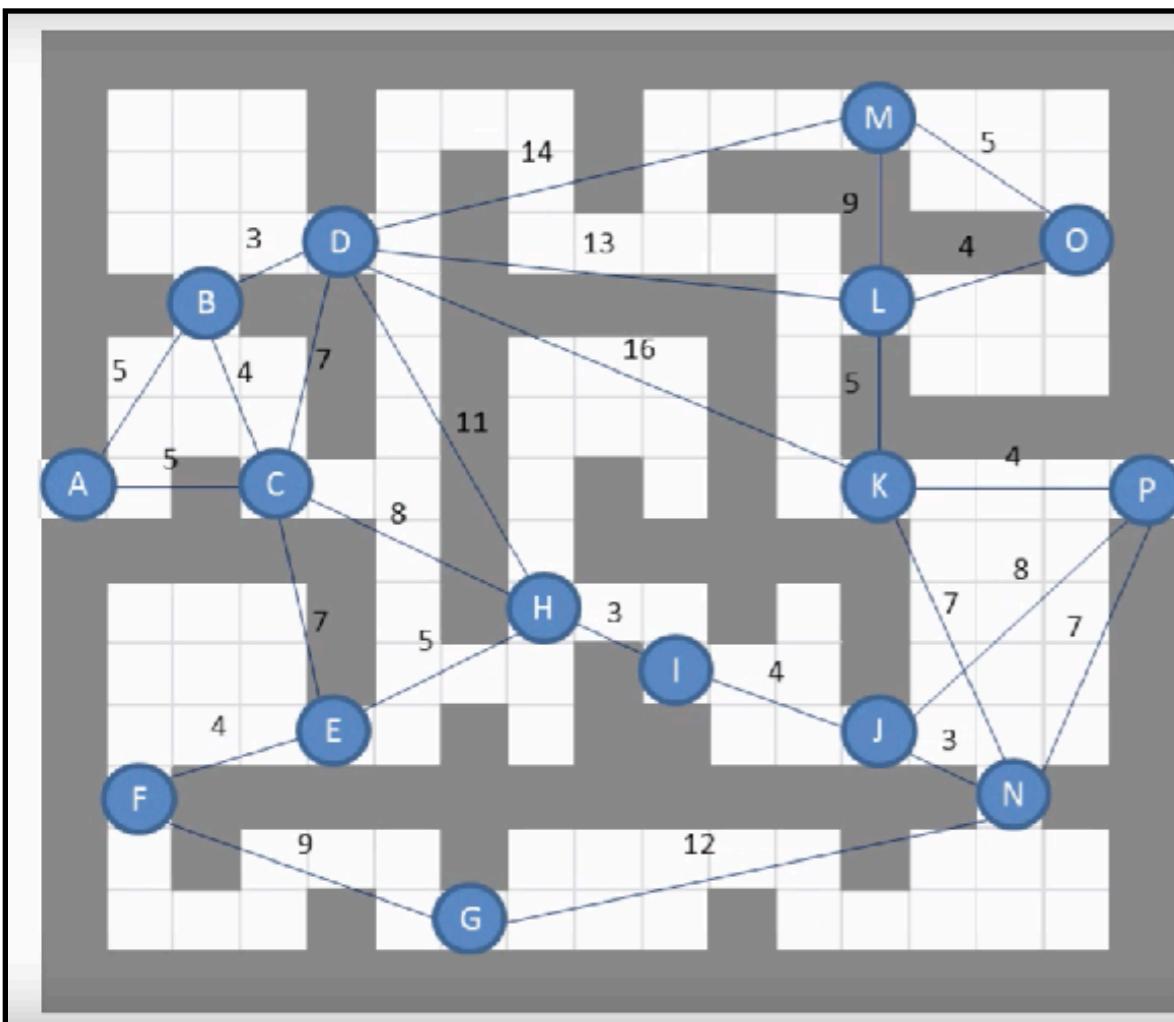
# Components of A\*

- **Initial State:** A
- **Goal State:** P
- **G-cost:** the cost from initial state (A) to current, e.g. time
- **H-cost:** the cost from current state to goal (P) e.g. straight line distance
- **F-cost:** the cost from Initial state (A) to current to goal (P)
- $F = G + H$

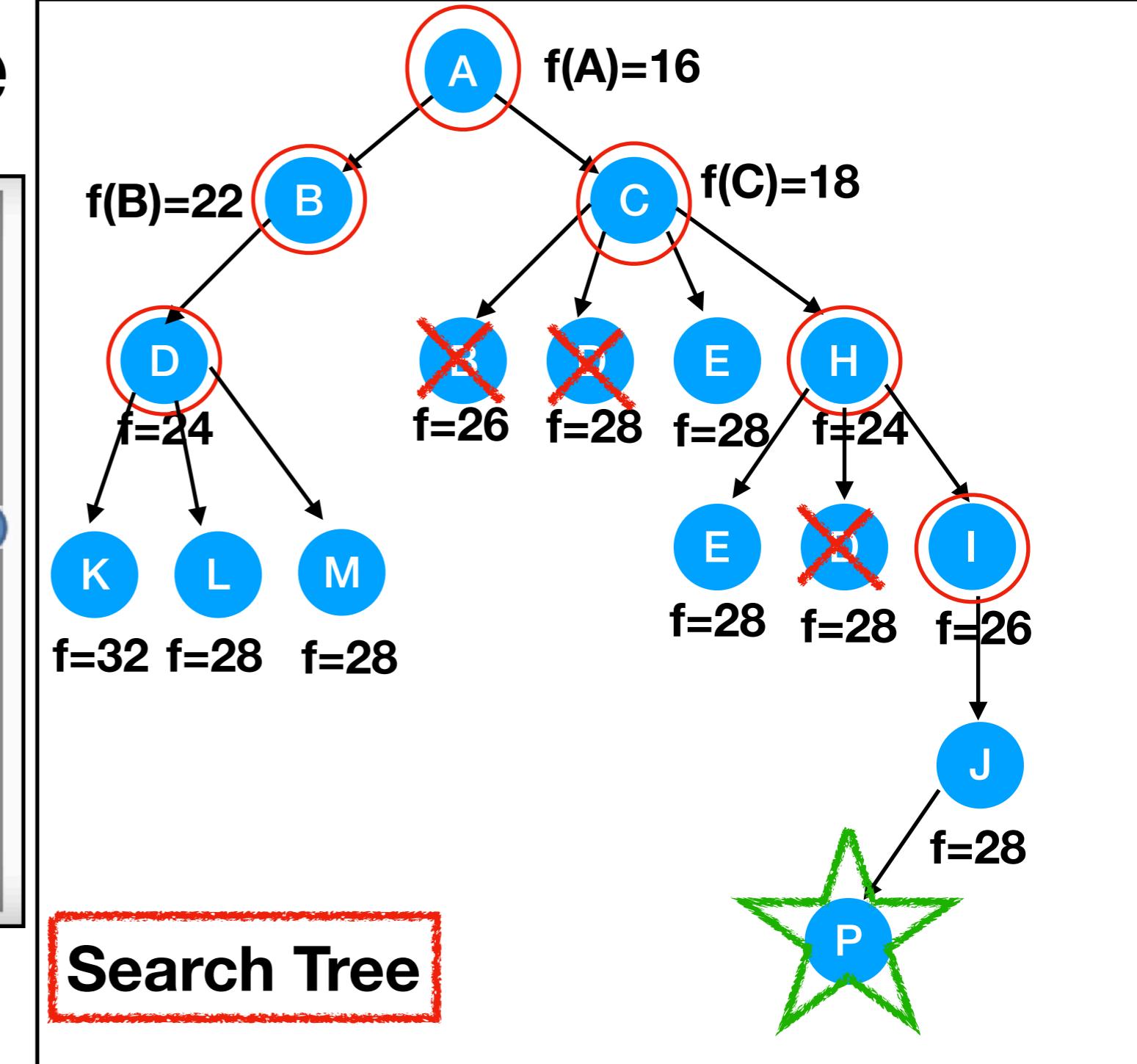
Suppose SLD(AP) = 16  
SLD(BP) = 17  
SLD(CP) = 13



# A\* Maze Example



Maze as Graph



- **State Space:** search tree
  - all possible locations that can be reached from A
- **State:** nodes in the search tree
  - one possible location in State Space, for example, E
- **Action:** edges in the search tree
  - at a location (say J), choose which way to go, for example, P

# A\* Components in the ASU Problem

- **Initial State:** the seed
- **Goal State:** an ASU with goal pop and goal ur
- **G-cost:** 
$$G = w_0 \frac{(\text{start pop} - \text{current pop})}{\text{start pop}} + w_1 \frac{(\text{start ur} - \text{current ur})}{\text{start ur}}$$
- **H-cost:**  $H = 0$
- **F-cost:** 
$$F = G + H = w_0 \frac{(\text{start pop} - \text{current pop})}{\text{start pop}} + w_1 \frac{(\text{start ur} - \text{current ur})}{\text{start ur}}$$

- How do we set the goal for an ASU grown from a seed?
  - The goal ur is 6.5%
  - The goal pop is estimated using BIP
- Why do we set the F-cost that way?
  - F is a weighted summation of percentage changes in pop and ur
  - When choosing the next node to expand
    - We want pop to increase as much as possible
    - We want ur to decrease as little as possible

# A\* Search State Space

**Initial State**  
 $f=0$

Add 6  
 $f=24$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 7  
 $f=26$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 8  
 $f=29$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 1  
 $f=29$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 2  
 $f=30$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

We repeat this process until a goal node is reached. The goal node is returned as the resulting ASU.

# A\* Search State Space

Add 6  
 $f=24$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 7  
 $f=26$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 8  
 $f=29$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Add 1  
 $f=29$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

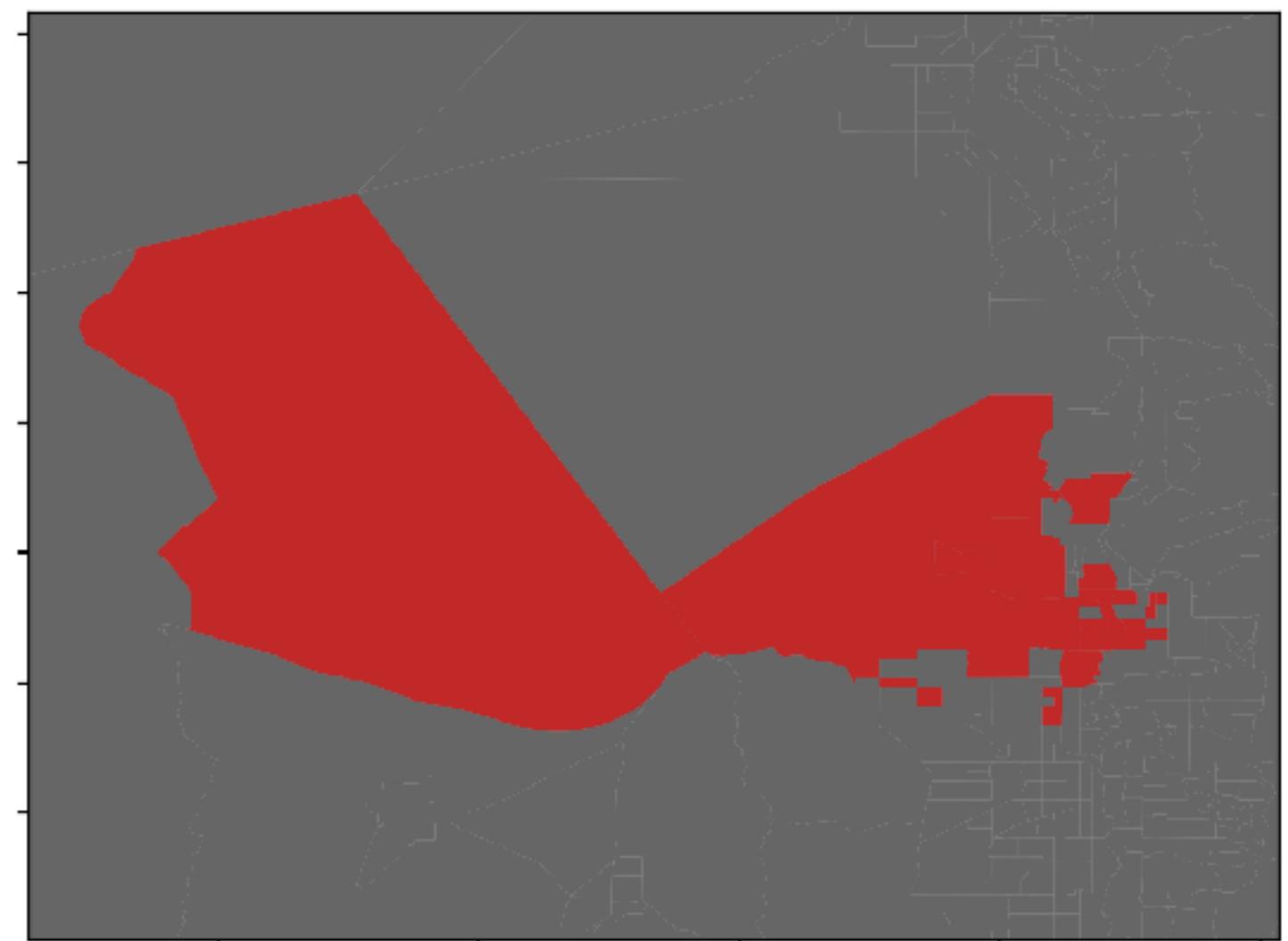
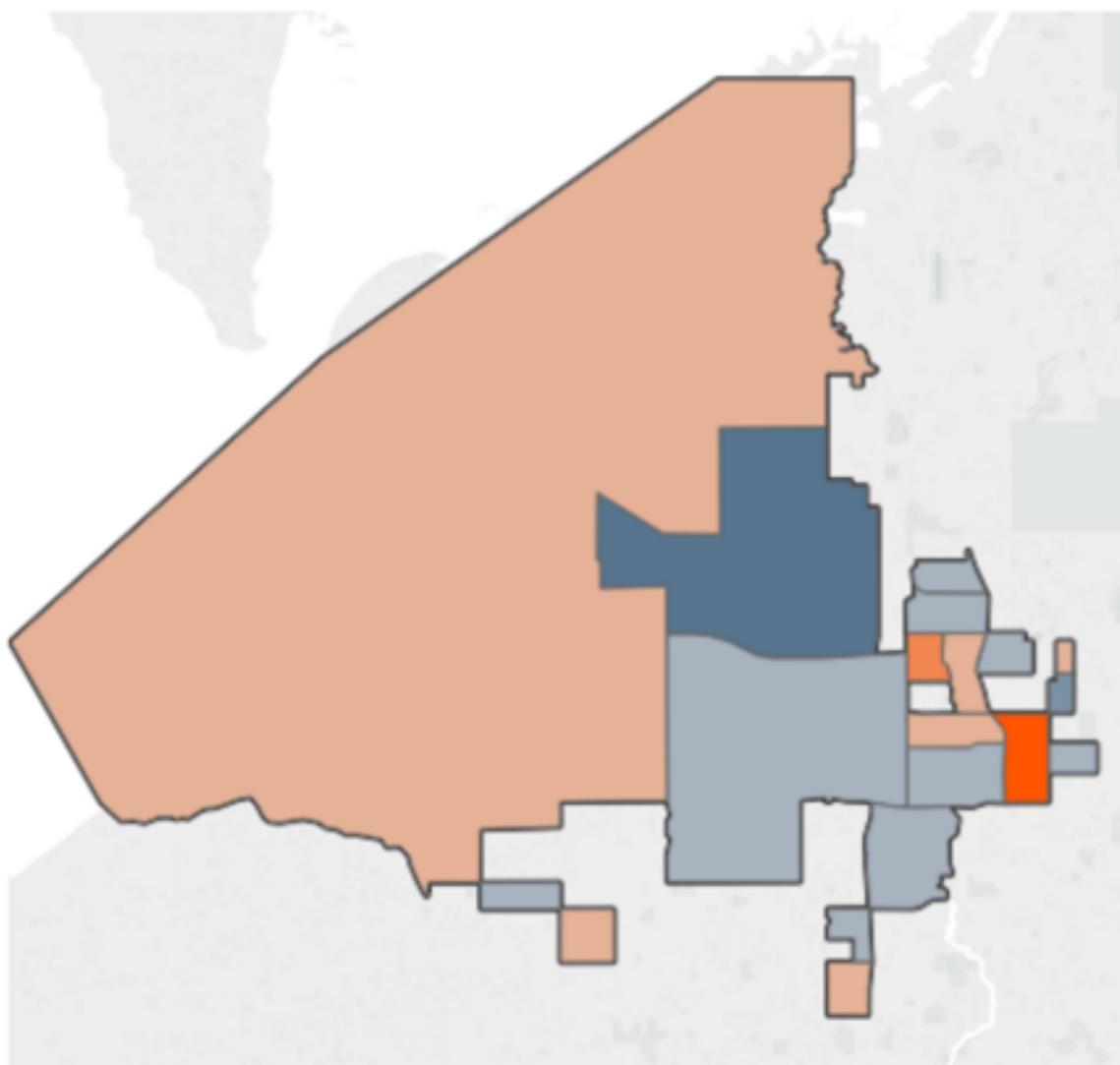
Add 2  
 $f=30$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$x_6$	$x_7$	$x_8$	$x_9$	$x_{10}$
$x_{11}$	$x_{12}$	$x_{13}$	$x_{14}$	$x_{15}$
$x_{16}$	$x_{17}$	$x_{18}$	$x_{19}$	$x_{20}$
$x_{21}$	$x_{22}$	$x_{23}$	$x_{24}$	$x_{25}$

Initial State  
 $f=0$

- **State Space:** all possible ASUs can be grown from seed
- **State:** one possible ASU in the Space
- **Action:** at one ASU, choose which one to add

# Marginal Results - A\*



**[Workforce Services, 2017]**      Aggregate URate  
6.53%

Aggregate 2015 Pop  
91,202

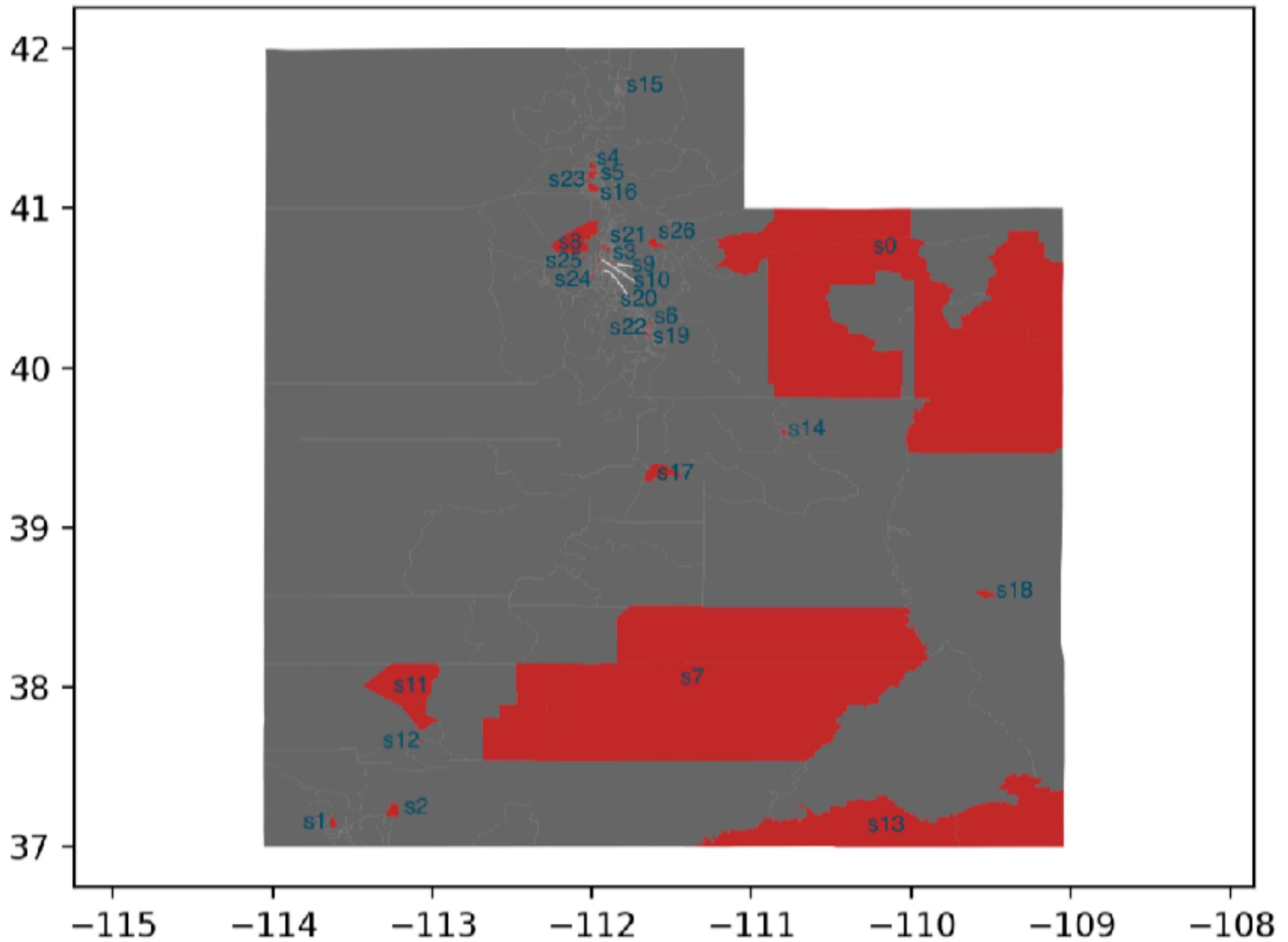
**Obtained using A\***  
Aggregate URate  
6.46%

Aggregate 2015 Pop  
101,000

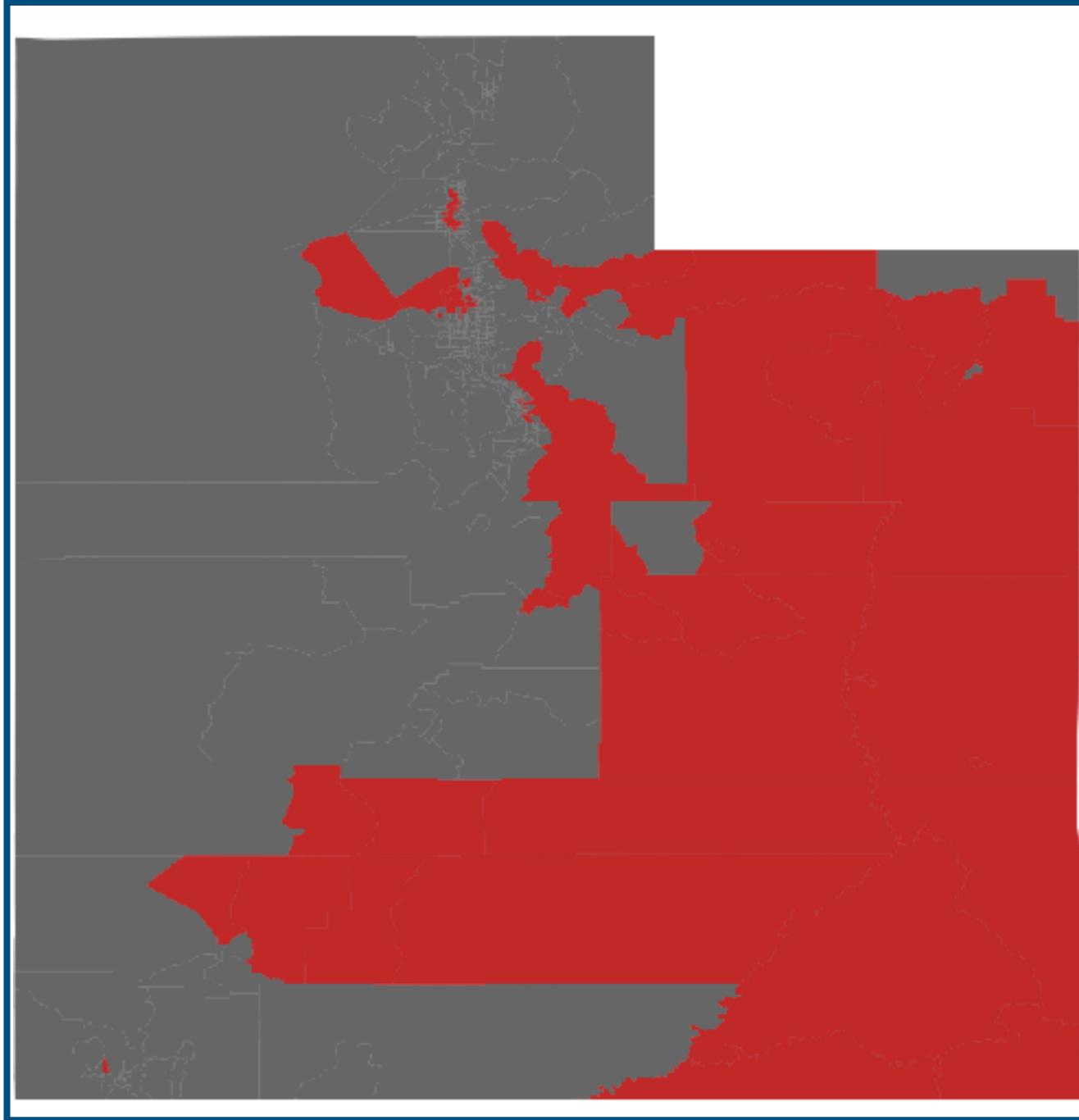
# Find All ASUs

- So far, we have been discussing how to find a single ASU using BIP or A\*
- Now, we will explain how to find all ASUs in the state of Utah
  - Initialize a result set A
  - Store all census tracts whose  $ur \geq 6.5\%$  in a seed set S
  - For each seed in S
    - If the seed is not in A
      - Grow an ASU from the seed using BIP or A\* without repetition
      - Add the resulting ASU to A
    - After all seeds in S are processed, we return A
- In the next slide, we will show the seed set S on a map
- In the slide after the next, we will show all ASUs (the result set A) found by BIP and A\* respectively

# ASU Seeds



# Joint Results from BIP and A\*



Obtained using BIP

Total 2015 Pop  
**371,742**



Obtained using A\*

Total 2015 Pop  
**369,352**

# Code and References

- **Code**
  - <https://github.com/splashmountain/asu>
- **Tools**
  - PySAL for Spatial Data Analysis (<http://pysal.readthedocs.io/en/latest/users/installation.html>)
  - PuLP for Linear Programming (<https://pythonhosted.org/PuLP/>)
  - Geopandas for Spatial Data Plot (<https://automating-gis-processes.github.io/2016/Lesson2-geopandas-basics.html>)
- **Papers**
  - Redistricting Using Constrained Polygonal Clustering [Joshi et al. 2012]
  - Contiguity Constraints for Single-Region Site Search Problems [Cove and Church 2000]
  - A Formal Basis for the Heuristic Determination of Minimum Cost Paths [Hart et al. 1968]