Albuquerque Fire Department Wait Times

New Mexico Supercomputing Challenge
Final Report

Team: 56
Cottonwood Classical and Del Norte High School

Team members:

Ayvree Urrea ayvreeurrea@gmail.com

Kiara Onomoto kiaraonomoto@gmail.com

Sponsor Teacher:

Karen Glennon

Patty Meyer

Mentor:

Nick Bennett

Don Nguyen

Table of Contents:

Executive Summary

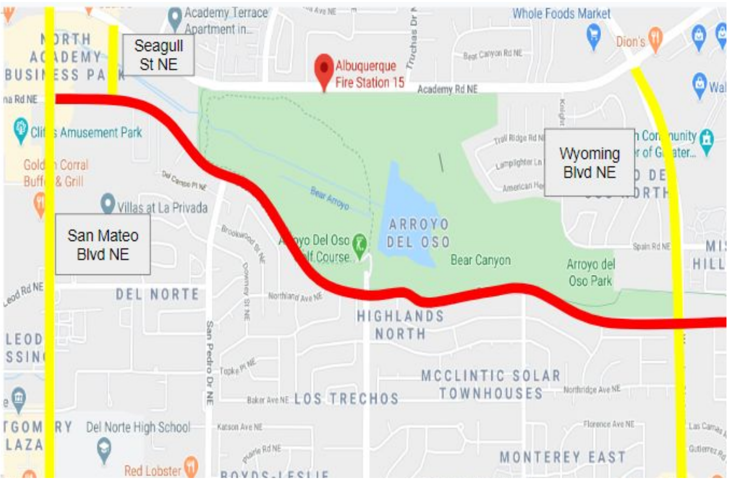
Fires and emergency situations are common around the world and in turn, exceptionally hazardous. The bigger issue, however, is that Fire Departments in Albuquerque, New Mexico have prolonged response times. Every single day, fire departments all around Albuquerque are called to save people from emergencies. They are tasked with the job of saving lives, protecting people from danger, and stopping fires. In order to get the job done efficiently and effectively, they need to arrive at their desired destination in less than five minutes. There are 22 fire departments scattered around Albuquerque, but all have problems getting to their destinations due to extended travel, lack of accessibility routes, compact communities, etc.

This is the second year building on the current project and we have expanded our knowledge of response times and traffic flow through meetings with traffic engineer designers and fire departments who informed us of important variables and statistics. This year we decided to switch our program language from Netlogo to Python in order to better address the variables of response times and gain more accurate mathematical solutions that can lead to advanced statistics for our project solution. A maze works well in our code because it models what an actual fire department trip would look like. Our program utilizes Pygame with a maze to represent a neighborhood, a firetruck, fires, and streetlights. The arrow keys control navigation of the firetruck and give coordinates and direction with each move, the maze cells are blocked. As the firetruck moves around the maze, fires show up randomly in a Poisson distribution to simulate fires that appear unexpectedly. There is a running clock that goes up 1 every tick and jumps depending on the severity of the fire for the time it takes to put the fire out.

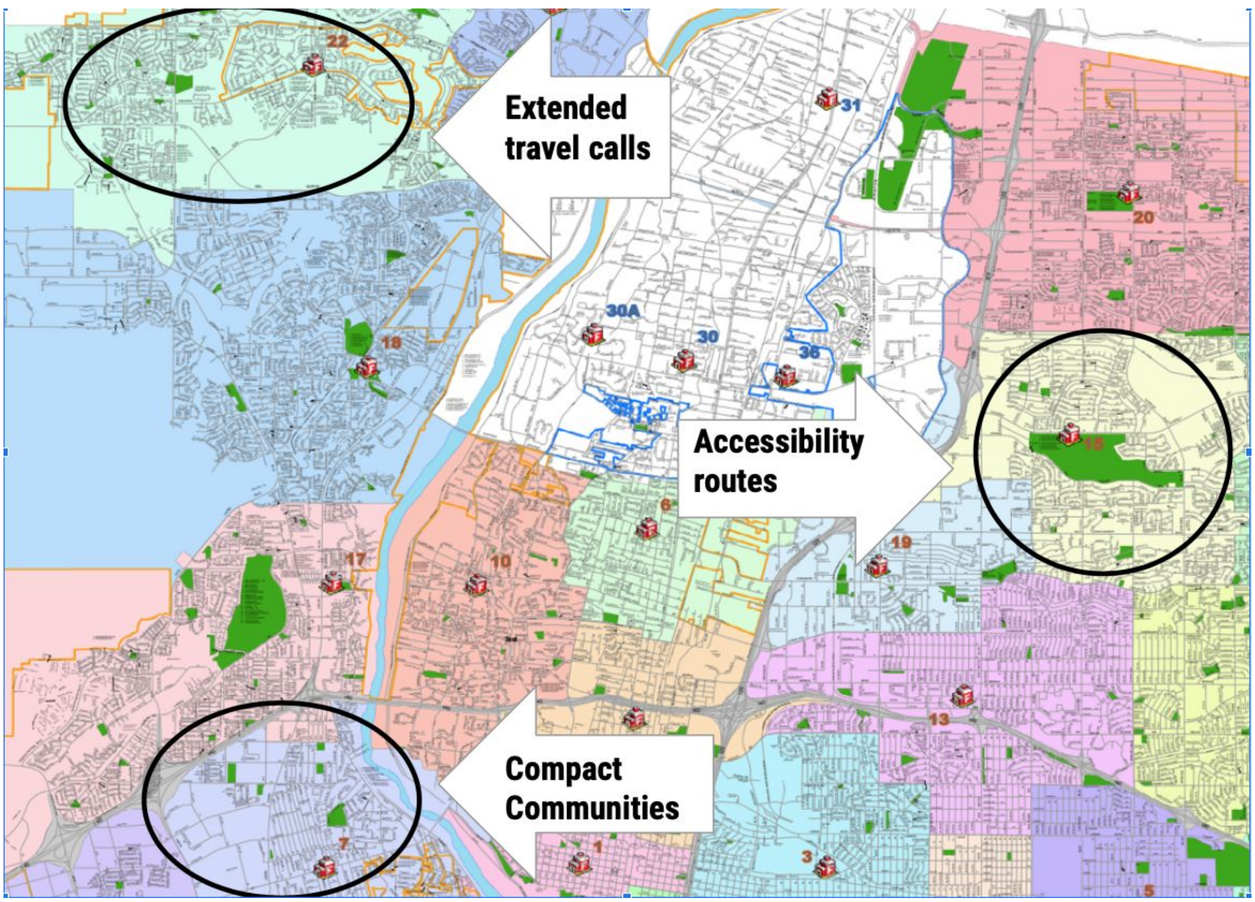
To make the most accurate representative model, we interviewed traffic directors and a Fire Battalion Chief within the community. For example, Battalion Chief Justin Staley informed us about fire department regulations and Amanda Herrera, who is a traffic engineer, taught us about the way roads are made in relation to optimizing response times. Through our research, we gained new knowledge for variables like traffic, Opticoms, and road changes. Using this data, we implemented a code base in the hope that its solutions could be used in place to lessen fire department wait times.

Problem

In Albuquerque, fire departments take too long to get to their destinations once they are called. This is inadequate because it can impact communities, where the emergencies are taking place. This can be a danger to the houses on fire and the neighborhood around it. In 2017, on average, it took around 6 minutes and 40 seconds for the firemen to arrive at their destinations, while in 2018 wait times took about 6 minutes and 54 seconds. This information indicates that Albuquerque's wait times are increasing due to the growth of the population and vehicles on the road. This makes it difficult for firemen to get to some locations. The area around Arroyo Del Oso golf course and surrounding houses has only one main fire station for such a big neighborhood. Because of that reason we decided to focus our research on that area to investigate if we could model a better path for firefighters in order to reduce response time. Factors that can affect these times can range from other vehicles to the traffic lights. Another problem this station faces is that there are only three main streets/points that the fire station can be accessed: San Mateo, Wyoming, and Seagull. This can be a boundary problem because only 3 roads leading to Osuna (the road that this neighborhood connects to) can cause more traffic and delays for the emergency systems. Firemen also have to make extended travel of the corresponding neighborhood sometimes. District calls that are made by individuals outside of the fire station region help other stations that are too busy or lack materials. This can make the firemen really busy, also resulting in slower response times. Because of all these factors, finding the fastest path to a location and navigating through unexpected vehicles or accidents also complicates the problem. This leaves time for fires to get bigger and become more of a danger to the people nearby. Fires affect many people and those extra few minutes can make a big difference. The fire stations must have a turnout time of 80 seconds or less, meaning they must be informed of the call, its location, and when the unit leaves the fire station. The first engine must come to the scene in a minimum of 4 minutes, 90% of the time with four personnel. The second unit needs to come to the scene within 6 minutes for 90% of responses, also with 4 personnel (7)



This image shows Fire station 15 and its corresponding roads it must take to get to neighborhoods south of the golf course such as Highlands North and Los Trechos. The yellow highlighted lines are the streets that can get to the red street of Osuna (the main road that the neighborhood connects to). This is a boundary problem because there are only three roads that are available for this large neighborhood, as well as a golf course in between.



The picture above visualizes the different traffic problems that the fire department faces in Albuquerque. (1)

Objectives

Research:

- Find data on Fire Departments in Albuquerque
- Learn more about traffic flow
- Learn more about the Poisson distribution
- Find resources to help with fire departments
- Find resources from traffic control.

Code:

- Learn Github
- Learn how to use Python
- Learn how to use Pycharm
- Find mentors to help with code
- Model neighborhoods
- Add events and obstacles

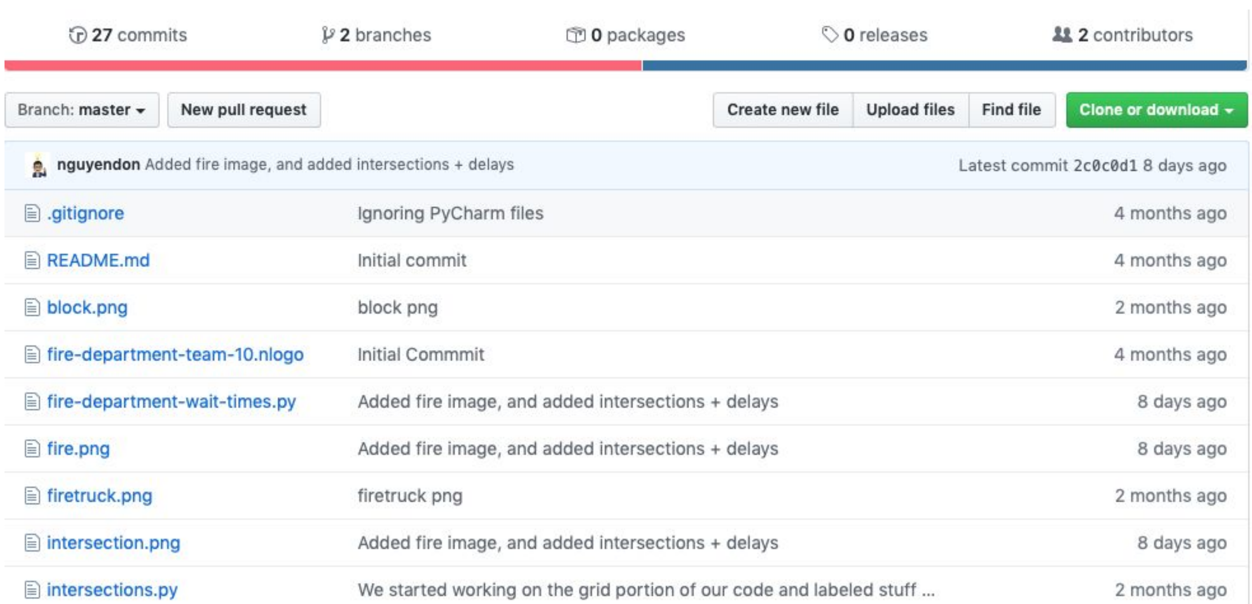
Research

Fire departments are aware of the problem in Albuquerque but have been unsuccessful in lowering response times. Many variables can impact the time it takes to get from one place to another. Because there are so many variables that affect response times and fires occur randomly, we wanted to study probability distribution in our code. In order to gain knowledge on traffic, how roads are made in relation to fire departments, average response times, and the precautions fire departments already take we sought out help from Zak Cottrell, who is a civil engineer designer that has now worked with us for about four years. This year, he advised us on finding the right sources to help us learn more about fire departments. (3)

Battalion Chief, Justin Staley of the Albuquerque Fire Department advised about the different fire departments throughout Albuquerque. He provided us with a map of Albuquerque fire departments corresponding districts they respond to. Staley suggested that we might want to look at variables that affect response times, such as traffic in general, extended travel (when a fire unit has to go to a house that is not in their specific district), navigation with how fire departments choose the specific route with the shortest distance, construction, geographical locations, etc. Fire departments that we found to be the most problematic were stations 7, 15, and 22. All of these stations have a specific issue that causes their response times to be slower. For example, station 15 lacks accessibility routes since the Arroyo Del Oso Golf course is in the middle of the district, causing slower response times with the southern houses. San Mateo, Wyoming, and Seagull road are the only available roads to get to the neighborhood houses that reside on the other side of the golf course. Station 7 tends to many compact communities. Since it is located in Central, slow traffic problems occur often due to the population being quite dense. Lastly, fire station 22 has to make a lot of extended travel due to their district being very spread out amongst western Albuquerque. Overall, all of these locations demonstrate some of the types of variables that can greatly affect all response times. (1)

Amanda Herrera is in the Department of Municipal Development Traffic Division of Albuquerque. Amanda taught us what variables cause traffic to slow down, and how roads are changed/made to help fire departments and other emergency systems. She told us about OptiCom, which is a type of system that changes the streetlight to green when an emergency system is near a traffic signal. Unfortunately, these lights aren't always reliable and are not installed at all the traffic lights in Albuquerque. One thought that we investigated as a possible solution was to model if emergency systems could detect from further away would it be helpful in improving response times by improving the current delay of streetlights. (2)

This year we worked with a Software Engineering Designer at Roku in Texas. He introduced Python and the Pycharm platform to us and led us in reworking our code from last year into Python. He lives in Texas, so we communicated through google hangouts and Github. Google hangouts allowed us to video call with him and share our screen to effectively collaborate. We also communicated through Github, which is a platform that allows you to share and collaborate on code with other people regardless of their location. This allowed us to work more efficiently as a team and work on the code under configuration management. It also allowed us to view the changes each team member made and subsequently build off of them.



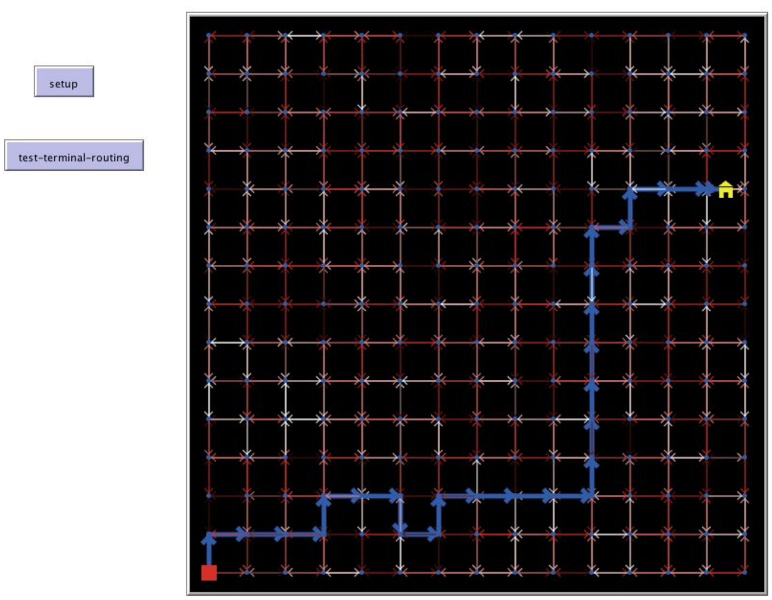
The photo above shows the Github repository, where we can collaborate on our code with each other and mentor.

Lastly, we learned a little bit about the Poisson theory for our code. In the Poisson theory, it describes a series of discrete events that are known, but the exact time of the events are random. For example, in a local Starbucks while one customer orders and receives their order, other customers continue to come in at random times to repeat the process. In other words, the customers have a specific procedure that happens like getting their drink before the next person can order and this happens at a randomly generated time and then repeats. This theory allows us to model in our code that fires can randomly be distributed and will stay there until managed, while other fires can start as well in the meantime. This allows for a very accurate “real” simulation with probability.

Methods

Last Years project:

We started this project last year only looking at station 15, where the Arroyo Del Oso golf course is in the middle of the district. In our Netlogo program, we used a network extension that used the minimum weight to give us the shortest route possible from one node to another. This is a case of Dijkstra's algorithm which is when you find the shortest distance from one node to another. We randomly generated the “accident” around the map and kept the “station” on the bottom left corner of the screen. While we learned a lot from this method, we wanted actual data from the code such as coordinates and fire severity making python a better choice to try.



This image shows our program last year that used the networking extension and Dijkstra's algorithm.

Start:

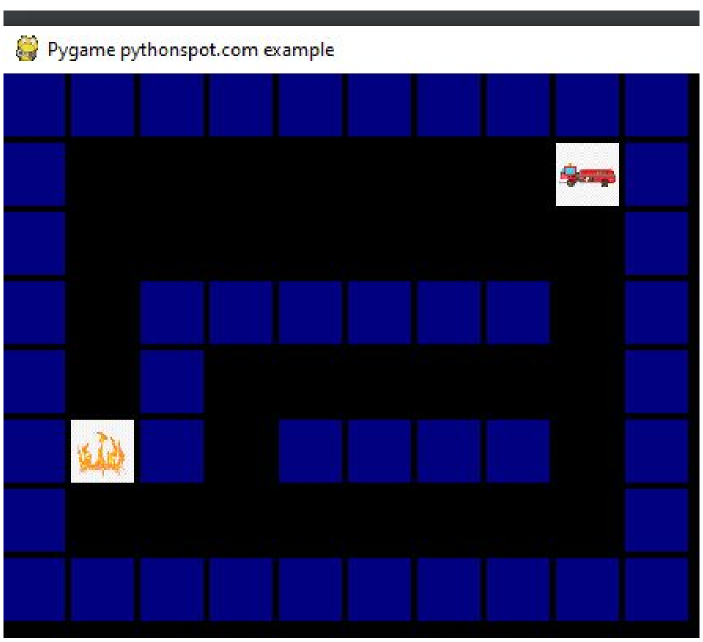
We knew coming into this year that we wanted to build on last year's work and develop a better understanding and improve upon what we did last year. We wanted to learn more about how to control certain functions in our program to make our project more math-based with more factors that are applicable to the real world. One example is the ability to utilize streetlights and time to see how the model predicts how long it takes for a fire unit to get to a house fire with different obstruction variables in place. To do this, our mentor advised us to migrate our code to Python instead of Netlogo because it is more flexible and able to simulate the factors we wanted to investigate. We wanted to add more realistic variables like a neighborhood, firetruck, intersections, and fires. At the beginning of the year, we worked with our contact, Don Nguyen, to help us with our code and decided to use Github as a way to stay in touch. This year communication and teamwork were very important for our team as we wanted to be more effective while writing the code and conducting research. Both of us had other extracurriculars at the beginning of the year so we had to plan in advance and meet outside of Monday's supercomputing session and utilize our resources to communicate effectively and efficiently.

After Evaluations:

After evaluations, we got more help with our code from Nick Bennett and Don Nguyen and started adding some of the important variables. We learned about a random distribution theory called Poisson and decided to use that as a base for adding fires and intersections to our code. The importance of this project and how we can make an impact with it was also brought to our attention. We believe that continuing this project for another year will allow us to add all the necessary layers and variables to make a difference in Albuquerque Fire Departments and achieve our ultimate goal of providing a valuable resource to the community improving emergency response times for those in need.

Pygame/Python

In our program, we are trying to symbolize the fire truck and its surrounding neighborhoods. We are using Python as opposed to Netlogo this year because it can make our project more math-based with variables that can be realistic and applicable to the real world. We also imported a maze Pygame to best visualize and simulate a neighborhood that a fire could be taking place. At first, we had the “player” or firetruck automatically be navigated around the map, while the fires were randomly distributed at random times and can only be stopped when the fire truck goes on top of it. This was when the fire truck was being moved by the arrow keys. However, we changed the program by pressing the spacebar, the fire truck will not move until it sees a fire, then it will go to the closest fire with the highest severity using flood fill. There are also stoplights incorporated in the code to slow down the firetruck at random points. We also have a clock that counts every time the spacebar is pushed. The clock is also impacted by the streetlights and the severity of the fire which can add a random amount of time depending on how long it takes the fire to be extinguished.



This is the interface of the code and shows how we use the firetruck as navigation and the fires that are randomly generated throughout the map.

```python
from pygame.locals import *
import pygame
import random
```

This image shows how we imported a Pygame that helped generate the maze

This shows how we added navigation to the player to make it move based on the arrows on the keyboard

```python
class Player:
    speed = 1
    position = Point(1, 1)

    def moveRight(self):
        self.position.x = self.position.x + self.speed

    def moveLeft(self):
        self.position.x = self.position.x - self.speed

    def moveUp(self):
        self.position.y = self.position.y - self.speed

    def moveDown(self):
        self.position.y = self.position.y + self.speed
```

```python
class Fire:
    def __init__(self, x, y, start):
        self.position = Point(x, y)
        self.start = start
        self.severity = 1
        while True:
            if random.randrange(MEAN_FIRE_FIGHT_DURATION) == 0:
                break
            self.severity += 1
```

This image shows how we added fires and with random severity, which affects the clock

```
class Maze:
    def __init__(self):
        self.M = 10
        self.N = 8
        self.obstacles = []
        self.maze = [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                     [1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
                     [1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
                     [1, 0, 1, 1, 1, 1, 1, 1, 0, 1],
                     [1, 0, 1, 0, 0, 0, 0, 0, 0, 1],
                     [1, 0, 1, 0, 1, 1, 1, 1, 0, 1],
                     [1, 0, 0, 0, 0, 0, 0, 0, 0, 1],
                     [1, 1, 1, 1, 1, 1, 1, 1, 1, 1], ]

    def draw(self, display_surf, obstacle_surf, intersection_surf):
        for x in range(0, self.M):
            for y in range(0, self.N):
                if self.maze[y][x] == 1:
                    display_surf.blit(obstacle_surf, (x * BLOCK_SIZE, y * BLOCK_SIZE))

                if self.maze[y][x] == 2:
                    display_surf.blit(intersection_surf, (x * BLOCK_SIZE, y * BLOCK_SIZE))

    def is_cell_clear(self, position):
        if self.maze[position.y][position.x] != 0:
            return False

        for obstacle in self.obstacles:
            if position.x == obstacle[1].x and position.y == obstacle[1].y:
                return False

        return True

    def is_cell_drivable(self, position):
        return self.maze[position.y][position.x] == 0

    def add_obstacle(self, type, position):
        self.obstacles.append((type, position))
```

This photo shows how we established what the maze will look like and how the cell blocks were placed and to make certain spaces unavailable to go through

```python
    def on_init(self):
        pygame.init()
        self._display_surf = pygame.display.set_mode((self.windowWidth, self.windowHeight), pygame.HWSURFACE)

        pygame.display.set_caption('Pygame pythonspot.com example')
        self._running = True
        # Load the image files
        self._firetruck_surf = pygame.image.load("firetruck.png").convert()
        self._obstacle_surf = pygame.image.load("block.png").convert()
        self._fire_surf = pygame.image.load("fire.png").convert()
        self._intersection_surf = pygame.image.load("intersection.png").convert()
        # Resize the images so they are the same size
        self._firetruck_surf = pygame.transform.scale(self._firetruck_surf, (40, 40))
        self._obstacle_surf = pygame.transform.scale(self._obstacle_surf, (40, 40))
        self._fire_surf = pygame.transform.scale(self._fire_surf, (40, 40))
        self._intersection_surf = pygame.transform.scale(self._intersection_surf, (40, 40))
```

The above image shows how we upload certain images to symbolize the firetruck, fires, intersections, and maze.

```python
    def advance_clock(self, interval=1):
        for i in range(interval):
            self.clock += 1
            if random.randrange(MEAN_FIRE_INTERVAL) == 0:
                while True:
                    x = random.randrange(self.maze.M)
                    y = random.randrange(self.maze.N)
                    fire_point = Point(x, y)
                    if self.maze.is_cell_clear(fire_point):
                        self.fires.append(Fire(x, y, self.clock))
                        self.maze.add_obstacle('Fire', fire_point)
                        break
        print("clock={}".format(self.clock))
```

This image above shows how we made the clock and how the severity of the fire increases it and how for every move the player takes, time goes up.

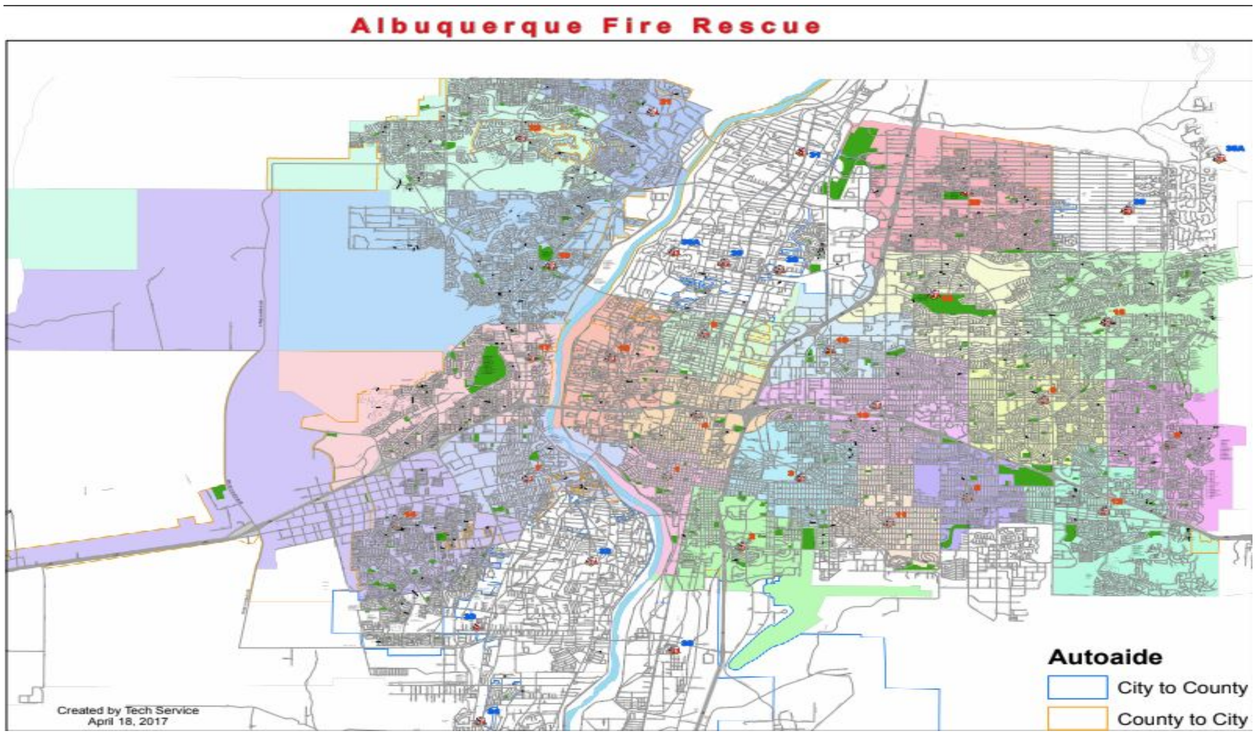
```python
        self.floodfill(maze.M * maze.N)

    def floodfill(self, num_cells):
        self.maze[self.position.y][self.position.x] = -num_cells
        boundary_set = set(self.boundary(self.position))
        while boundary_set:
            p = boundary_set.pop()
            self.maze[p.y][p.x] = self.fill_value(p)
            boundary_set = boundary_set.union(set(self.boundary(p)))

    def boundary(self, position):
        return filter(lambda p: self.maze[position.y][position.x] + 1 < self.maze[p.y][p.x] <= 0,
                      self.base_neighborhood(position))

    def base_neighborhood(self, position):
        neighborhood = []
        neighborhood.append(Point(position.x - 1, position.y))
        neighborhood.append(Point(position.x + 1, position.y))
        neighborhood.append(Point(position.x, position.y - 1))
        neighborhood.append(Point(position.x, position.y + 1))
        return filter(lambda p: (0 <= p.x < self.columns) and (0 <= p.y < self.rows), neighborhood)

    def fill_value(self, position):
        best_neighbor = 1 + min(self.maze[p.y][p.x] for p in self.base_neighborhood(position))
        return min(best_neighbor, self.maze[position.y][position.x])

    def min_neighbor(self, position):
        return min(self.base_neighborhood(position), key=lambda p: self.maze[p.y][p.x])
```

This image shows how we used a flood fill in our program to get the fire truck to move on its own and to find the shortest possible time to get to fires with the highest severity.

Tables/Graphs

Movement	EBT	EBR	WBL	WBT	NBL	NBR
Lane Configurations	↱		↰	↑	↖↗	
Volume (veh/h)	188	13	68	62	18	134
Sign Control	Free			Free	Stop	
Grade	0%			0%	0%	
Peak Hour Factor	0.89	0.41	0.45	0.74	0.38	0.52
Hourly flow rate (vph)	211	32	151	84	47	258
Pedestrians						
Lane Width (ft)						
Walking Speed (ft/s)						
Percent Blockage						
Right turn flare (veh)						
Median type	None			None		
Median storage veh)						
Upstream signal (ft)						

This is what standards traffic management is looking for when designing roads. On the left, it explains the types of movement many lanes and intersections that are present have. Examples can have different configurations, such as the lane may have several intersections or the volume/number of vehicles present in the lanes. The top of the table shows examples of lanes and how they are graded according to these standards (2).



This picture shows the fire departments in Albuquerque and the corresponding area they respond to fires in. This has helped us get a general idea of what fire department to study and the neighborhoods that specific fire departments service (1).

Results

In our results, we were able to figure out how the clock in our program works and how obstacles such as the severity of the fire and traffic lights impact the time. We found out that the clock can jump when the fire truck is held up especially by the level of severity the fire has. Our next step is to come up with multiple programs featuring different obstacles like other cars, stoplights, and boundary problems to see how these impact the times. We want to have a function in our program that will file out the difference between the current time and when the fire started, getting us a response time. Then try out different maps to see how they invoke the fire and if it will change with new barriers. The main solutions that we believe could help decrease the response times can be some sort of emergency lane that could only be used by emergency vehicles, Opticoms that can turn a stoplight automatically green when an emergency system is there, more functioning departments around the city, or even a lane that goes through some boundaries. For example, Station 15 can be benefitted from an extra lane only for specific emergencies because access can be hard in this area due to only three streets available for the station. Another solution can be a lane going through the boundary, since the Arroyo Del Oso golf course prevents easy access for the emergency vehicles to go through, due to only having three streets around the course that they can go on. The lane can somehow go through the course whether it be using a bridge above the course or just a street going through, creating more accessibility. Station 7 can be benefited with an Opticom because due to the populations being so compact, an Opticom can make the process of traffic speed up and turn the lights green when fire emergency systems need to get through. Lastly, station 22 can get faster response times by creating more departments in areas more spread out to make sure these areas can be safe from having a fire that can cause destruction too fast before a fire truck can get there.

```
clock=42
You moved the truck  to (1,1)
clock=43
You moved the truck  to (2,1)
clock=44
You moved the truck  to (3,1)
clock=45
You moved the truck  to (4,1)
```

This image shows how the clock normally function for every time the space bar is pushed

```
You moved the truck  to (4,1)
clock=83
clock=84
You moved the truck  to (6,1)
```

This image shows how the severity of the fire impacts the fire, in this case, it made the time move from 45 in the previous image to 83 after the fire was fought.

Conclusion

Emergency systems need a solution to help them find a way to get to their destinations in a timelier manner. We believe that possible solutions to aid in lengthy fire department wait times are adding Opticoms, taking the shortest determined path, or accounting for more variables. Adding OptiComs to more street lights would provide emergency systems the option to change the lights more efficiently, therefore getting rid of one of the variables affecting response times. In addition, we learned from our code that the Fire Departments have to be more efficient in rating fire severity and taking the shortest and fastest path to a destination. We think that by putting more thought into the variables that affect their response times and changing them one by one, fire departments will be able to shorten response times. It is extremely important to find a solution for emergency vehicles to get to their locations more efficiently and effectively to potentially save someone or something at risk. We plan to continue this project next year to use our code to gather statistics and limit the variables that have a huge impact on response times. Once we are able to do this, we will gather real information that can apply to Fire Departments and shorten response times for our community.

Significant Achievements:

The most significant achievements our team has had this year are communicating with each other and other resources and learning a completely new coding language. This year we used multiple platforms to stay in touch and work together, including Github, Google Hangouts and screen share, Zoom, and meeting outside of supercomputing classes. We have learned to stay in constant communication with each other to work on our project more efficiently and effectively. In addition, we found ways to stay in touch with our mentor from another state and worked well with our resources. Through communication with our resources, we were able to gain valuable data and variables that can make our project more realistic and useful to our community. Going into this year, we have also learned a completely new coding language as we had previously only been familiar with Netlogo. We were able to find people to help us grow as coders and teach us how to work with the language. This allowed us to effectively create a model with more of the variables that we had hoped for and possible mathematical solutions.

Ayvree Urrea:

A significant achievement I had this year was becoming more confident in my presenting and coding. As this is my third year doing the project I am starting to learn how to project my voice and present in a way that is easier to understand. I am now able to explain our research and respond to questions on the spot in a confident and knowledgeable way. Throughout this year in particular I have also gotten alot better at coding and understanding what each part of the code does. I feel more comfortable explaining the code and adding new parts to it. In addition, I have learned how to take advantage of my opportunities as a woman in STEM and put myself out there more as I applied for an Aspirations in Technology Award at NCWIT and got the Honorable Mention Award.

Kiara Onomoto:

My significant achievement this year in the Supercomputing challenge is presenting myself to more opportunities and communicating with others. This year I applied for an Aspirations in Technology Award at NCWIT and got the Rising Star Award. I think I was able to become more confident and willing to apply myself to more opportunities that come with being a woman in STEM. I was also able to improve my communication skills throughout the year, whether it be communicating with my partner, mentors, resources, or communicating our project to an outside source. Throughout the year I have learned to put my thoughts together and explain information in a more logical and understandable way.

Acknowledgments

Amanda Herrera, Department of Municipal Development Traffic Engineering Division

Civil Engineering Designer, Sgt. Zak Cottrell, Albuquerque Police Department, Traffic Division/Motor Unit.

Don Nguyen, Software Engineer Designer at Roku in Texas

Justin Staley, Battalion Chief Special Operations / FOC Albuquerque Fire Rescue

Karen Glennon, Retired Teacher

Nick Bennett

Patty Meyer, Retired teacher

Bibliography

1. Justin Staley, Battalion Chief Special Operations / FOC Albuquerque Fire Rescue

2. Amanda Herrera, Department of Municipal Development Traffic Engineering Division

3. Civil Engineering Designer, Sgt. Zak Cottrell, Albuquerque Police Department, Traffic Division/Motor Unit.

4. "Albuquerque Fire Rescue." *City of Albuquerque*, www.cabq.gov/fire.

5. "Albuquerque Fire Rescue 2018 Annual Report." *Albuquerque Fire Rescue,* https://www.cabq.gov/fire/documents/annual-report-2-corrected.pdf

6. "Poisson Theory." *Statistics How To, https://www.statisticshowto.datasciencecentral.com/poisson-distribution/*

7. *Moore-Merrell, Lori. "Understanding and Measuring Fire Department Response Times." Lexipol, 16 July 2019, www.lexipol.com/resources/blog/understanding-and-measuring-fire-department-response-times/ .*