**Introduction**

Childcare has extremely important significance for families and even society. For young children, high-quality early childhood education helps shape a favorable environment for their early growth, which in turn helps them achieve long-term success in life. In addition, good early childhood education can alleviate parents' stress, help them focus on their work, and enhance their professional competitiveness. For society as a whole, sufficient childcare resources help employers retain employees and effectively reduce future crime rates among children. It can be seen that providing sufficient childcare has multiple positive implications, but due to various factors such as income, childcare resources are relatively insufficient. According to relevant data, including many parts of New York, 51 of the United States is facing the dilemma of childcare deserts. The National Governors Association (NGA) believes that the factors that hinder teenagers from accessing mental health services are high nursing costs, imbalanced resources in mental health services among states. In the survey on the causes of the children's fox desert, the most commonly cited reasons are insufficient operating funds from state and federal governments, as well as the high financial costs required to operate childcare centers. In addition, previous studies have shown that a lack of economic support for providers is one of the main reasons for childcare in deserts. Therefore, using optimization methods to assist the New York City government in developing plans that can provide sufficient childcare resources with minimal budget constraints and ensure fairness has positive practical significance.

**Background**

Based on the known information, this article will transform how to design the best solution for childcare in desert environments with limited budget into a linear problem for processing. Due to the limitations of practical conditions and relevant policies, multiple constraints were assumed during the problem-solving process. Firstly, to address the childcare crisis, New York City needs to construct new childcare facilities. To avoid excessive concentration of resources, there needs to be distance restrictions between facilities. This issue is discussed in the first part of this article. Secondly, according to differences in household income and parental employment, different areas of New York City are divided into high demand areas for childcare and normal demand areas for childcare: regions where at least 60 % parents are employed, or the average income is $60000 or less per year is considered as high demand areas. For high demand areas, if the number of childcare places exceeds half of the children aged 2 to 12, they are not considered as childcare deserts. For areas with normal demand, it only needs to reach one-third of the number of children in the same category. In addition, due to the policy requirements of New York City, the available quota must reach at least two-thirds of the number of children aged 0 to 5. When expanding the number of childcare quotas, the government faces two choices. One is to select a location to establish new facilities, which come in three types with different capacities and construction costs. The second is to expand old facilities, with a maximum capacity of 500 or 0.2 times the original capacity. As the capacity increases, the cost also increases. This part of the problem will be discussed in the second part of the main body of this article. Finally, considering the maintenance of social equity, the New York City government hopes to maintain relative fairness in childcare resources among different regions within the budget and maximize the social coverage index of childcare. This part of the problem will be discussed in the third section of the main body of this article. In summary, this article simplifies the solution to the New York City childcare desert dilemma into three linear optimization problems and uses Gurobi and Python languages to solve the three optimization problems and discusses the practical significance of the problem results.

**Problem 1 Assumptions**: We make the fundamental assumption that the scenario of childcare in NYS can be formulated into a linear program with a finite, feasible, and convex region of solutions. This assumption is necessary as it forms the basis for our linear program. Another assumption we make is that if an existing facility currently has a total capacity of 0, it will not be expanded. This makes sense because the maximum number of slots a facility can have is the minimum of 500 and 1.2 times its current capacity. A 1.2 multiplier on 0 is still 0, so it cannot be expanded. This assumption is necessary because there are a handful of existing facilities with a total capacity of zero.

**Data Preparation:** The data is provided in the project instructions and needs only a little cleaning after downloading. We first drop unnecessary data from our population, such as the number of people aged 40-45 in a zip code. Then, we calculate the 0-12 population using proportions as follows, rounding down at the end:

Then, we use a similar methodology to calculate the 0-5 capacity of an existing facility.

Finally, to wrap up the data cleaning, we drop existing facilities that have a total capacity of zero, because they are of no use. They cannot be expanded, as per our assumptions.

**Problem 1 Decision Variables**: The first set of decision variables are non-zero integers that represent how many small, medium, or large facilities to build in each zip code. Hence, we have 3 \* number of zip codes, as the quantity of these decision variables.

For expansions, we make two more sets of decision variables: one representing the number of 0-5 slots we add to each existing facility in each zip code, and one representing the number of 5-12 slots we add to each existing facility in each zip code. It can be defined as follows:

**Problem 1 Cost Function**: The cost function is built piece by piece. First, we will add the cost of building new buildings. This is simple, as the cost is fixed for each size and given to us:

For expanding existing facilities, the cost is more involved. It considers the current capacity of the existing facility as well as the number of slots added of each type. The relationship for a single facility is described below:

To get our final cost function, we simply add our cost of building new facilities in zip code with the cost of expanding existing facilities in that same zip code. Then, we sum over all zip codes in NYS to get the total cost. That results in our final objective (second summation is still inside the first one):

**Problem 1 Constraints**: The first constraints we can add are on the decision variables. Since we can only build a positive integer number of new buildings in each zip code, we must restrict that variable to be non-negative and integer. Same goes for expanding slots in existing facilities, they must be integer slots greater than zero.

Next, we add the constraint that a facility can only be expanded by up to 20%, or until it reaches 500 slots. This is represented as below:

Then we set demand constraints, where depending on whether the zip code is high demand or normal, our facilities must provide the required number of slots. To do this, we first calculate the new number of slots available in each zip code. This is simply the existing slots plus the slots we gain from building new facilities plus the slots we gain from expanding existing facilities, all in a particular zip code. The number of slots we gain by making new buildings are provided in the instructions.

Then, we obtain the corresponding populations in that particular zip code:

Then, we make the final demand constraints, changing based on high demand status for total capacity but keeping the same 0-5 capacity constraint regardless of high demand status. High demand in a zip code is defined as having employment over 60% or income under 60,000:

**Problem 1 Results**: After optimization, we can successfully meet all demands and constraints by using $316,248,854. To analyze our data we first begin by grouping our zip codes into groups of 11 by their proximity. This makes the data less granular and easier to view in charts. Once the data is aggregated into groups of 11, we first look at how often each size of building was built in each group:

A graph of a number of buildings

Description automatically generated

From the chart, we can see that the overwhelming majority of the time, large buildings are built in our zip codes. This makes sense as they provide a very large number of slots for less than double the cost of small facilities that are built. We also see that the amount of small and medium buildings is mostly flat across all the zip code groups, while the large buildings frequency has a discernible shape. One unique point is that the peak of the large buildings being built happens in zip code group 25, which corresponds to zip codes in the Northern Poughkeepsie area. This area may be susceptible to childcare deserts due to its low socioeconomic status in its resident population. Next, we look at expansion vs building new facilities. The results are seen below:

A graph of different numbers

Description automatically generated with medium confidence

In this chart we can see that most slots are added by building new facilities rather than expanding current facilities. Looking deeper, we can see that both distributions roughly have the same shape. This implies that when slots are added, usually you must build new facilities and expand existing ones, you just obtain more slots by building. It also implies that there is no zip code or geography where it is disproportionally better to expand vs build new slots, the shapes are roughly the same. Finally, we look at expansion in detail, analyzing which type of slots are usually added when facilities are expanded. The results are below:

A graph of numbers and numbers

Description automatically generated with medium confidence

In this chart, we see a density of points on the x-axis, implying that most of the time you will be adding 0-5 slots. This makes sense as most existing facilities had a shortage of 0-5 slots. We also see that the range of the 0-5 slots is a lot higher, reaching 4000 in some zip code groups. Whereas for 5-12 slots, we add at most 250 in a zip code group. The last deduction that can be made from the data is that 5-12 slots are only really built when not a lot of 0-5 slots are built. This indicates that perhaps facilities may have a shortage in one type of slot or the other.

**Problem 1 Mathematical Formulation:**

**Notation Guide**

**Decision Variables:**

**Handy Expressions:**

**Mathematical Formulation: (2nd summation is still inside first summation)**

**Subject to**

**Problem 2**

**Problem 2 Assumptions**: We make the fundamental assumption that the scenario of childcare in NYS can be formulated into a linear program with a finite, feasible, and convex region of solutions. This assumption is necessary as it forms the basis for our linear program. Another assumption we make is that if an existing facility currently has a total capacity of 0, it will not be expanded. This makes sense because the maximum number of slots a facility can have is the minimum of 500 and 1.2 times its current capacity. A 1.2 multiplier on 0 is still 0, so it cannot be expanded. This assumption is necessary because there are a handful of existing facilities with a total capacity of zero.

**Data Preparation:** The data is provided in the project instructions and needs only a little cleaning after downloading. We first drop unnecessary data from our population, such as the number of people aged 40-45 in a zip code. Then, we calculate the 0-12 population using proportions as follows, rounding down at the end:

Then, we use a similar methodology to calculate the 0-5 capacity of an existing facility.

Finally, to wrap up the data cleaning, we drop existing facilities that have a total capacity of zero, because they are of no use. They cannot be expanded, as per our assumptions.

**Problem 2 Decision Variables**: The first set of decision variables are binary and represent whether a new building of a particular size will be built in a potential location. For each zip code, we obtain a list of all potential locations, and then create three binary decision variables (small, medium, large) that indicate whether a building of that size is built there. This process is repeated for all the potential locations in that zip code and then for all zip codes. The total number of these decision variables is 3 \* number of potential locations in a zip code, added up for all zip codes.

For expansions, we make two more sets of decision variables: one representing the number of 0-5 slots we add to each existing facility in each zip code, and one representing the number of 5-12 slots we add to each existing facility in each zip code. It can be defined as follows:

**Problem 2 Cost Function**: The cost function is built piece by piece. First, we will add the cost of building new buildings. This is simple, as the cost is fixed for each size and given to us:

For expanding existing facilities, the cost is more involved. It considers the current capacity of the existing facility as well as the number of slots added of each type. For problem 2, the cost is piecewise, with the cost per slot changing with the size of your expansion. The brackets for size of expansion are 0-10%, 10-15%, and 15-20%. If our expansion rate covers two brackets, we apply the respective cost for the portion in those brackets. For example, if we expand a facility by 13%, we apply the first brackets cost to the first 10% expansion, and the second brackets cost for the remaining 3% in our expansion. Before we implement this, we first calculate the expansion rate for a given facility:

Then, we use additional decision variables to model our piecewise cost as discussed in the prior section. The first variables we declare are positive continuous variables that represent the portion of our expansion that will “fit” in each of the three cost brackets. We also declare three binary variables that indicate whether a certain cost bracket is active or not.

Then, we introduce the logical constraints that force to fit within their cost brackets and so that all add up to our total expansion rate for that facility.

Next, we introduce more logical constraints using the big M method to make sure that if a certain cost bracket is active, then the preceding bracket must be fully utilized. For example, we can only apply the cost associated with a (10, 15) % expansion if we have already expanded by 10%. We ensure that in the following way:

Finally, we build our final cost function for expanding existing facilities in a given zip code by combining our logical decision variables with the fixed cost and per slot cost given in the instructions. We make sure not to forget the constant but additional cost of adding 0-5 slots:

To get our final cost function, we simply add our cost of building new facilities in zip code with the cost of expanding existing facilities in that same zip code. Then, we sum over all zip codes in NYS to get the total cost. That results in our final objective (second summation is still inside the first one):

**Problem 2 Constraints**: The first constraints we can add are on the binary decision variables that indicate whether we build a facility of a certain size in a certain potential location. Since we can only build one building of a certain size in a given location, we must constrain the sum of our binary variables to be at most 1.

We also ensure that our expansion variables, the ones that indicate how many slots for each age group are added to existing facilities, are non-negative integers.

Then, we can focus on a new limitation that is required in problem 2, the one of location. It states that any new location where a facility is built must not be less than 0.06 miles from another new location or existing location. We can set these constraints in two parts, first by looking at distances between new facilities, and next by looking at distances between new facilities and existing facilities. To set constraints between just the new facilities, we first get all the unique pairs of potential locations from our zip code. In Python, this can be done using itertools.combinations().

Next, we use the haversine package in Python to calculate the distance between two locations given their latitudes and longitudes. Using this distance, we set the constraints, making sure that if they are too close, then you can only build one facility of a certain size in both of those two locations.

Next, we set location constraints that enforce distance between newly built locations and existing locations. First, we get the list of all combinations between the existing facilities in a zip code and potential locations in a zip code. This is done in Python by using itertools.product().

Then, similar to before, we use the haversine package in Python to calculate the distance between each item in each tuple in our set. If a potential location is too close to an existing location, we set the constraint that we cannot build a facility at that location.

Next, we add the constraint that a facility can only be expanded by up to 20%, or until it reaches 500 slots. This is represented as below:

Then we set demand constraints, where depending on whether the zip code is high demand or normal, our facilities must provide the required number of slots. To do this, we first calculate the new number of slots available in each zip code. This is simply the existing slots plus the slots we gain from building new facilities plus the slots we gain from expanding existing facilities, all in a particular zip code. The number of slots we gain by making new buildings are provided in the instructions.

Then, we obtain the corresponding populations in that particular zip code:

Then, we make the final demand constraints, changing based on high demand status for total capacity but keeping the same 0-5 capacity constraint regardless of high demand status. High demand in a zip code is defined as having employment over 60% or income under 60,000:

**Problem 2 Results**: After optimization, we can successfully meet all childcare demands and facility constraints by using $320,532,485.

**Problem 2 Final Mathematical Formulation:**

**Notation Guide**

**Decision Variables:**

**Handy Expressions:**

**Mathematical Formulation:**

**Subject to**