Optimization for Sustainable Waste Management in Ogemaw County

_15.071 Homework 7_Lisa Wan

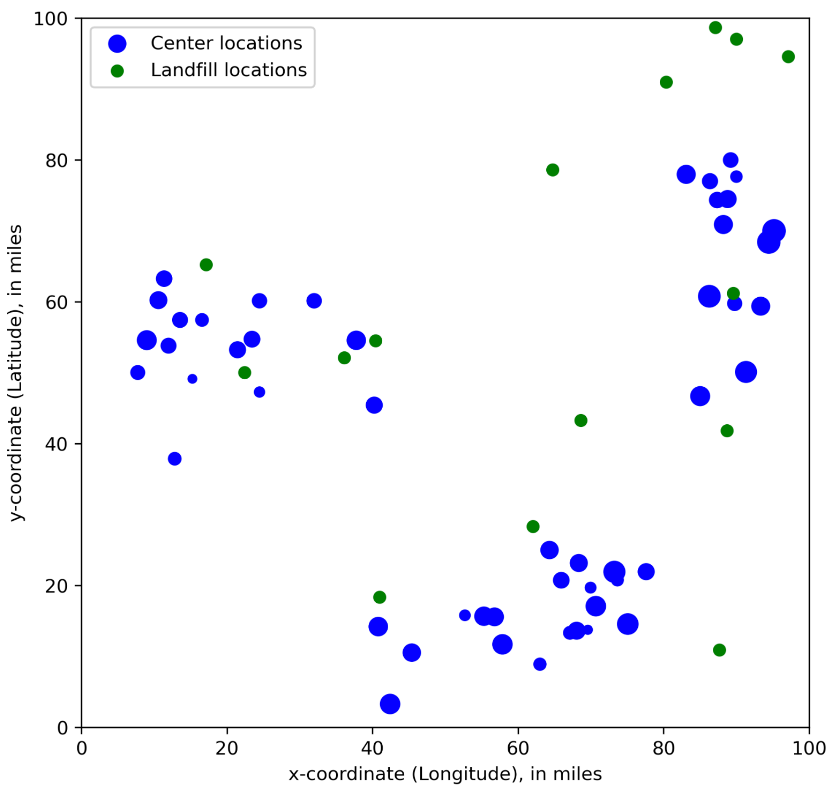
Problem Context

Waste management refers to the collection, transportation, and processing/disposal of household and light industrial trash and related materials. In most countries, waste management is a logistics function that is typically managed at a regional level -- at the county/province/prefecture/canton level in most countries. Sustainable solutions to waste managemnt problems are even more important as the cost of sanitary landfills has increased considerably. The decisions of where to construct sanitary landfills (and possible waste compaction transfer stations) -- in order to minimize total system costs -- is now tackled using modern optimization modeling in conjunction with other business analytics modeling tools.

KSA (Kendall Square Analytics) is an analytics-based consulting firm that has recently expanded its practice scope to include serving government and government-related organizations using business analytics for the delivery of a wide variety of public services, including waste management.

KSA has recently been retained by Ogemaw County to assist them in the development of a cost- and environmentally-efficient plan for the siting of sanitary landfills (and possibly waste compaction stations) and associated transportation and logistics decision planning. As shown in Figure 1, Ogemaw County is a "square" region 100 miles by 100 miles, with 50 town/city centers shown in the blue discs in the figure (with the size of each disc being proportional to the population of the town/city center). Also shown in the figure in green are the candidate locations of 15 candidate sanitary landfill locations.

Up until last year, Ogemaw County focused its waste collection efforts on acquiring and operating waste transportation trucks and efficiently designing collection routes. Waste was collected, transported, and disposed of in open-air waste dumps which burned the waste. Burning waste is one of the environmentally worst ways to dispose of waste, and so Ogemaw County was mandated by court order to construct sanitary landfills for waste disposal. They chose three sites for the construction of landfills as shown in Figure 2, whose names are Lake Lorraine, Muhlenberg Park, and Renfrow. (In the figure the gray lines show the transportation flows of waste tonnage from the town/city centers to the three landfill sites.) The total transportation cost of the plan in Figure 2 is estimated to be **$560,000 per week** based on the current weekly waste produced in each town/city center and using the unit waste transportation cost of **$0.97 per ton per mile**.


Center locations
Landfill locations
y-coordinate (Latitude), in miles
x-coordinate (Longitude), in miles
0
20
40
60
80
100

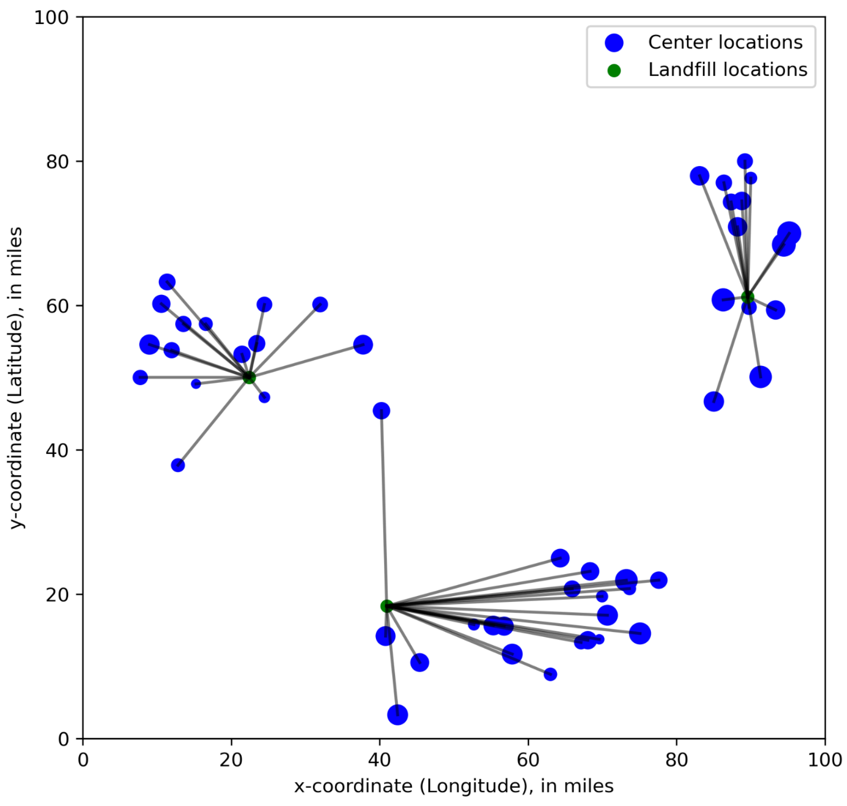

Center locations
Landfill locations
y-coordinate (Latitude), in miles
x-coordinate (Longitude), in miles
0
20
40
60
80
100

Figure 1 (left): Map of Ogemaw County showing town/city centers (in blue) and candidate landfill sites (in green).

Figure 2 (right): Recent plan for Ogemaw County siting of 3 landfills at the sites (i) Lake Lorraine, (ii) Muhlenberg Park, and (iii) Renfrow. The total weekly transportation cost of this plan is $560,000.

Ogemaw County has had some discussions with the US EPA (Environmental Protection Agency) in the hopes of receiving some sort of federal funding from the EPA for a more comprehensive waste management solution. The EPA thought that they might be able to provide direct funding for the construction of up to four (4) landfills, depending on the EPA's own budget and also depending on well Ogemaw County can make the case that this would be an efficient use of such federal funds. At this point Ogemaw County decided to retain KSA to assist them in determining a least-system-cost plan for choosing which candidate landfill sites to build, as well as the transportation flows of waste tonnage from town/city centers to landfills.

In the questions and exercises below, you should consider yourself to be one of the consultants on the KSA team working on this engagement for Ogemaw County....

Task 1: Where to build landfills to optimize transportation costs

The first task is to decide **which candidate sanitary landfills to build**, and then determine the transportation of waste tonnage from the 50 town/city centers to the chosen landfills, in order to minimize transportation costs.

> **A Quick Note on Optimization Modeling Workflow.** In general, the workflow when working with optimization models is as follows:

1. Bring in the relevant data
2. Transform the data into convenient data structures
3. Formulate the optimization model
4. Solve the model
5. Examine the solution

Often, upon studying the solution, you may decide to modify the optimization model. Thus, you may repeat steps 3-4-5 until the results are satisfactory. We will generally follow this process as we work through this assignment.

With the help of recently hired Research Assistants, the KSA team has constructed the following three dataset files:

```
**center_location.csv**: This file contains the names an
d locations of the 50 town/city centers. The first row c
ontains the descriptor for the columns in the file: name
, the center's location horizontal value (x-coordinate,
in miles), and the center's location vertical value (y-c
oordinate, in miles). Note that the units are shown in m
iles instead of longitude or latitude units, as miles ar
e more intuitive to work with. The first column contains
the name for each of the 50 town/city centers. The secon
d column displays the horizontal location coordinate, na
mely the $x$-coordinate value in miles, and the third co
lumn displays the vertical location coordinate, namely t
he $y$-coordinate value in miles.
```

```
    **landfill_location.csv**: This file contains the nam
    es and locations of the 15 *candidate* landfill sites
    . The first row contains the descriptor for the colum
    ns in the file: name, the candidate site's location h
    orizontal value (x-coordinate, in miles), and the can
    didate site's location vertical value (y-coordinate,
    in miles). Again, the units are shown in miles instea
    d of longitude or latitude units, as miles are more i
    ntuitive to work with. The first column contains the
    name for each of the 15 candidate sites, the second c
    olumn displays the horizontal location coordinate, na
    mely the $x$-coordinate value in miles, and the third
    column displays the vertical location coordinate, nam
    ely the $y$-coordinate value in miles.
```

```
        **center_info.csv**: This file some contains addi
        tional data for the 50 town/city centers that wil
        l be used in this exercise.  The second column co
        ntains the weekly quantity of waste produced by e
        ach town/city center, in tons per week.  The thir
        d column shows the average per capita income of e
        ach town/city center, in dollars.  And the fourth
        column contains the population of each town/city
        center, which will be used later in this modeling
        exercise.
```

Using the above datasets, we now construct an optimization in order to caluclate the best set of landfills to build in order to meet all the county's waste demand

Optimization Model Formulation

Let us consider the following optimization model:

Sets (“Lists”)

We first describe two sets ("lists"), namely the list of town/city centers, and the list of candidate landfill locations:

$$\begin{array}{ll} \mathcal{C}: & \text{The set ("list") of the 50 town/city centers,} \quad \mathcal{C} = \{\text{Sioux Falls, Apison, ...,} \\ \mathcal{L}: & \text{The set ("list") of the 15 candidate landfill sites,} \quad \mathcal{L} = \{\text{Avondale Estates,} \end{array}$$

Parameters (Data)

Next we define some of the data ("parameters") for the model, namely the weekly waste quantity produced by each town/city center, and the straight-line distances between each town/city center and each landfill site:

$$\begin{array}{rl} q_i: & \text{amount of waste generated (in tons/week) at town/city center } i \text{, for ea} \\ [\text{DCL}]_{ij}: & \text{straight-line Distance (in miles) between each Center } i \in \mathcal{C} \text{ and each L} \end{array}$$

Regarding the straight-line distances $[\text{DCL}]_{ij}$ above, the straight-line distance between a center i and a landfill site j is given by the well-known formula:

$$[\text{DCL}]_{ij} := \sqrt{|\text{x-coord. of center } i - \text{ x-coord. of landfill } j|^2 + |\text{y-coord. of center } i}$$

Decision Variables

We define two groups of decision variables. One group is the quantities of waste transported from each town/city center to each candidate landfill, and the other is the group of binary variables that indicate whether a candidate landfill site is assigned to be built or not.

$$[\text{XCL}]_{ij}: \quad \text{amount of waste transported from center } i \text{ to landfill } j \text{ in tons/week, fo}$$

$$z_j = \quad 1 \text{ or } 0 \text{, if landfill site } j \text{ is chosen to be built or not, for each } j \in \mathcal{L}$$

Optimization Model Formulation

With the above lists, parameters, and decision variables, we formulate our optimization model as follows:

$$
\begin{aligned}
\text{minimize} \quad & 0.97 \times \sum_{i \in \mathcal{C}} \sum_{j \in \mathcal{L}} [\text{DCL}]_{ij} \times [\text{XCL}]_{ij} \\
\text{subject to} \quad & \sum_{j \in \mathcal{L}} [\text{XCL}]_{ij} = q_i && \text{for each center } i \in \mathcal{C} \\
& [\text{XCL}]_{ij} \le M \times z_j && \text{for each center } i \in \mathcal{C} \text{ and each landfill site } j \in \mathcal{L} \\
& \sum_{j \in \mathcal{L}} z_j \le 4 \\
& [\text{XCL}]_{ij} \ge 0 && \text{for each center } i \in \mathcal{C} \text{ and each landfill site } j \in \\
& z_j \text{ is binary 0/1} && \text{for each landfill site } j \in \mathcal{L}
\end{aligned}
$$

Recall from the introduction section that the cost of transporting waste in Ogemaw County is estimated to be $0.97 per ton per mile.

Note that the second constraint group above uses a "big M" constant. You will shortly be asked below to compute an appropriate value of this constant.

Question 1(a): Describing and understanding the above model formulation

Describe and explain the above model formulation. Note that the model has three groups of constraints, and two groups of decision variables. In the questions below you should provide a detailed explanation of each of the constraints, the objective function, and the decision variables. We recommend that each answer be 1-3 sentences. Your answers should convince the grader that you have a thorough understanding of this optimization model formulation.

What is the purpose of each of the two groups of decision variables? What are the units of these decision variables?

The first group of decision variables, labeled as [XCL]$_{ij}$, serve the purpose of determining the amount of waste to be transported from each center i to each candidate landfill j, measured in tons per week. There are 50 town/city centers in Ogemaw County, and each landfill j is chosen from a set of 15 candidate landfills. The second group of decision variables, denoted as Z$_j$, have the purpose of indicating whether a candidate landfill j is selected for construction or not. These variables are binary, taking the value 1 if the landfill is chosen to be built, and 0 if it is not chosen.

Describe the objective function in one sentence. What are the units of the objective function?

The objective function aims to minimize the transportation cost of the amount of waste from all centers i to all candidate landfills j, measured by 0.97 multiplied by the sum of waste transportation quantities ([DCL]$_{ij}$) from all centers i to candidate landfills j ([XCL]$_i$). 0.97 is the unit cost of transporting waste in Ogemaw County, measured in ton per mile. The units of the objective function are dollars per week.

What is the purpose of the first group of constraints? What are units of these constraints?

The first group of constraints serve the purpose of ensuring that the total amount of waste transported from each center i to all candidate landfills j is equal to the amount of waste generated (in tons/week) at center i. The units are tons per week.

What is the purpose of the second group of constraints? What are the units of these constraints? Explain how the "big M" method works in this group of constraints.

The second group of constraints serve the purpose of limiting the amount of waste transported from each center i to each candidate landfill j based on whether the landfill j is selected or not, as indicated by the binary decision variable Z$_j$. For example, if Z$_j$ is 0, the value of [XCL]$_{ij}$ should be 0 since no waste can be transported to a non-selected landfill. The “big M” method is used to transform “if-then” binary constraints into linear constraints. By selecting the parameter M as a large positive constant, [XCL]$_{ij}$ can be a non-zero positive value when Zj is 1.

Calculate a valid, data-dependent value of M for the "big M" constant used in the second group of constraints.

The parameter M is a large positive constant, which is chosen to be greater than or equal to the upper bound of the decision variable [XCL]$_{ij}$ for all potential combinations of centers i and landfills j, which should be the maximum value among the volume of waste produced in each town center.

What is the purpose/role of the third constraint group? What are the units of the constraints?

The third constraint group serves the purpose of limiting the total amount of selected landfills to less than 4, suggesting that at most 4 landfills can be selected out of 15 candidate landfills for waste transportation. The units are measured in volume.

Question 1(b): Constructing, Solving, and Using the Optimization Model

In this question you will use gurobipy to construct, solve, and then use the optimization model formulated above. Gurobipy is an optimization modeling software library for building and solving linear, binary, and mixed-integer optimization models.

Step 1: We will import gurobipy and other data-manipulation and visualization libraries

In [1]:
```python
# !pip install gurobipy
import gurobipy as gp
from gurobipy import GRB
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
```

Step 2: Let us now read in the three data files described earlier.

In [2]:
```python
center_locations = pd.read_csv("center_location.csv", index_col="Ce
landfill_locations = pd.read_csv("landfill_location.csv", index_col
center_info = pd.read_csv("center_info.csv", index_col="Center")
```

Practical Tip: Whenever you read in data, we recommend that you take a quick look at the top few rows to check if the data "looks right". Here, we use .head() to examine the first few rows, but you can also use .tail() to examine the bottom few rows.

In [3]:
```
center_locations.head()
```

Out[3]:

	x	y
Center		
Sioux Falls	24.47	47.26
Apison	24.47	60.12
Grandin	11.97	53.79
Southern Gateway	16.57	57.42
Seattle	15.25	49.12

In [4]:
```
landfill_locations.head()
```

Out[4]:

	x	y
Landfill		
Avondale Estates	40.41	54.48
Hurt	36.11	52.11
Lake Lorraine	40.94	18.37
Muhlenberg Park	22.41	50.01
Arcadia	87.68	10.90

In [5]:
```
center_info.head()
```

Out[5]:

	waste tonnage	average income	population
Center			
Sioux Falls	260	44430	6466
Apison	525	38929	15929
Grandin	571	32742	13866
Southern Gateway	408	49612	11569
Seattle	183	31373	4578

The data looks good, so let us proceed to the next step.

Step 3: We define the set (or list) of centers and the set of landfills

In [6]:
```
centers = list(center_locations.index)
landfills = list(landfill_locations.index)
```

Let us check our work by printing the first five names in each list.

In [7]:
```
centers[:5]
```

Out[7]:
```
['Sioux Falls', 'Apison', 'Grandin', 'Southern Gateway', 'Seattle'
]
```

In [8]:
```
landfills[:5]
```

Out[8]:
```
['Avondale Estates', 'Hurt', 'Lake Lorraine', 'Muhlenberg Park', '
Arcadia']
```

The lists look good, so let us proceed to the next step.

Step 4: Let us compute the straight-line ("SL") distances between each of the centers and each of the candidate landfill sites. We will of course use these distances in our model formulation.

In [9]:
```
# compute the straight-line distances between all centers and all l
sl_distance_centers_landfills = pd.DataFrame(index=centers, columns
for i in centers:
    for j in landfills:
        center_x = center_locations.loc[i, "x"]
        center_y = center_locations.loc[i, "y"]
        landfill_x = landfill_locations.loc[j, "x"]
        landfill_y = landfill_locations.loc[j, "y"]
        distance = ((center_x - landfill_x)**2 +
                    (center_y - landfill_y)**2)**(1/2)
        sl_distance_centers_landfills.loc[i, j] = distance
```

Practical Tip: Once again, whenever you do a computation, we recommend you take a quick look at the output to make sure it "looks right

In [10]:
```
sl_distance_centers_landfills.head()
```

Out[10]:

	Avondale Estates	Hurt	Lake Lorraine	Muhlenberg Park	Arcadia	Ottumwa	Weiser	
Sioux Falls	17.498914	12.61	33.25497	3.436001	72.921559	64.471615	42.047297	1
Apison	16.908377	14.129745	44.881214	10.317737	80.113123	66.801198	49.198492	
Grandin	28.448369	24.198388	45.758467	11.103243	87.01469	77.671022	56.14911	
Southern Gateway	24.020599	20.248647	46.030418	9.434707	84.974952	73.811679	53.959277	
Seattle	25.724603	21.073199	40.069173	7.215102	81.895502	73.823804	51.177511	1

Step 5: With the preliminaries completed, we can now construct our optimization model in gurobipy

Please review the model formulation below carefully and ensure you understand how the mathematical model above has been "translated" into programmatic form.

In [11]:
```
# calculate big-M value
bigM = max(center_info["waste tonnage"])

# Create the model object (for question 1b)
model_question1b = gp.Model("question1b_model")

# Add the decision variables
xcl = model_question1b.addVars(centers, landfills, name="xcl")
z = model_question1b.addVars(landfills, vtype=GRB.BINARY, name="z")

# Construct the objective function
objective_function = model_question1b.setObjective(
    0.97 * sum(sl_distance_centers_landfills.loc[i, j] * xcl[i, j]
    GRB.MINIMIZE)

# Waste collection constraints:
# We must transport all waste away from each center
collect_waste_constraint = model_question1b.addConstrs(
    (sum(xcl[i, j] for j in landfills) == center_info.loc[i, 'waste
    name="collect_waste")

# We can only transport waste to landfill locations that have been
bibig_M_constraint = model_question1b.addConstrs(
    (xcl[i, j] <= bigM * z[j] for i in centers for j in landfills),
    name="big_M")

# We can build at most 4 landfills
max_num_landfills_constraint = model_question1b.addConstr(
    sum(z[j] for j in landfills) <= 4,
    name="max_num_landfills")
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2024-04-0
1
```

Step 6: We solve this seemingly-complicated optimization model with a simple command!

In [12]:
```
# Solve the model
model_question1b.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (mac64[rosetta2])

CPU model: Apple M2
Thread count: 8 physical cores, 8 logical processors, using up to
8 threads

Optimize a model with 801 rows, 765 columns and 2265 nonzeros
Model fingerprint: 0x83936481
Variable types: 750 continuous, 15 integer (15 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+03]
  Objective range  [1e+00, 1e+02]
  Bounds range     [1e+00, 1e+00]
  RHS range        [4e+00, 1e+03]
Presolve time: 0.01s
Presolved: 801 rows, 765 columns, 2265 nonzeros
Variable types: 750 continuous, 15 integer (15 binary)
Found heuristic solution: objective 1008948.0939

Root relaxation: objective 4.150351e+05, 54 iterations, 0.00 secon
ds (0.00 work units)

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

*    0     0               0    415035.06365 415035.064  0.00%
-    0s

Explored 1 nodes (54 simplex iterations) in 0.06 seconds (0.00 wor
k units)
Thread count was 8 (of 8 available processors)

Solution count 2: 415035 1.00895e+06

Optimal solution found (tolerance 1.00e-04)
Best objective 4.150350636482e+05, best bound 4.150350636482e+05,
gap 0.0000%
```

Examine the Solution

Now it is time to examine the output of the optimization model to see what the optimal solution looks like.

Which landfill sites are chosen in the optimal solution of the model?

In [13]:
```
built_landfills = []
for j in landfills:
    if z[j].x >= 0.9999:
        print("Build landfill at", j)
        built_landfills.append(j)
```
```
Build landfill at Lake Lorraine
Build landfill at Muhlenberg Park
Build landfill at Weiser
Build landfill at Renfrow
```

What is the total weekly cost of the optimal solution?

Since the objective function of the model is the total weekly cost, we can simply "read off" the optimal value.

In [14]:
```
model_question1b.ObjVal
```
Out[14]:
```
415035.0636482479
```

Let us "hand calculate" the total weekly cost of the optimal solution and see if it matches the optimal objective function value.

In [15]:
```
total_cost = 0.97 * sum(sl_distance_centers_landfills.loc[i,j]*xcl[
print("Total weekly cost is: $ %f" % (total_cost))
```
```
Total weekly cost is: $ 415035.063648
```

They match - excellent!

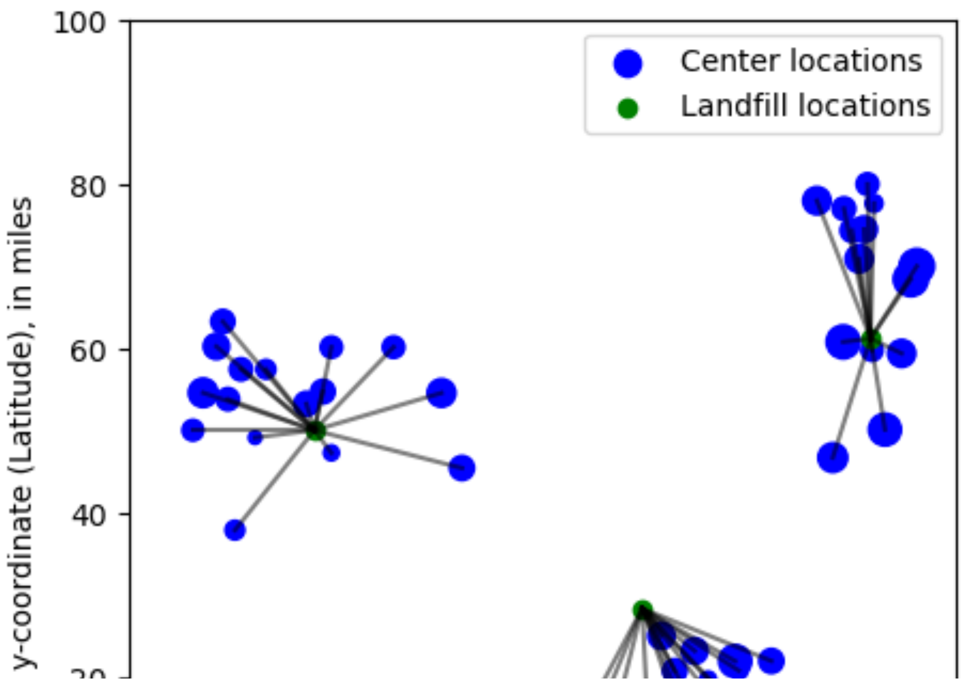
Practical Tip: Hand-calculating values and checking that they match the output of the model is a great way to ensure clear understanding of the model and to check for typos or command errors.

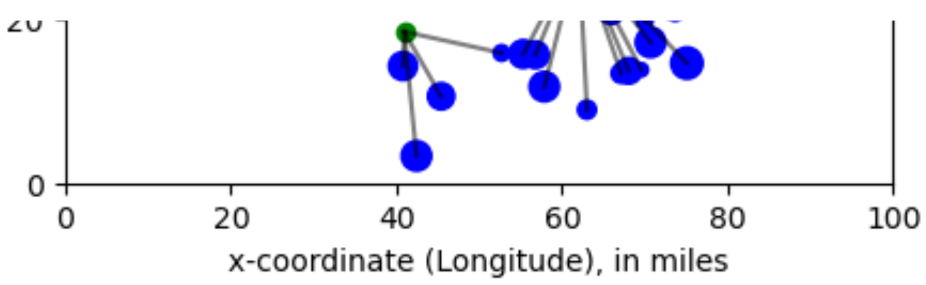
Visualize the Solution

To help visualize the solution, let us plot the locations of the chosen landfills and the assignments of landfills for each center.

In [16]:

```
from IPython.core.display import HTML
HTML("""
<style>
.output_png {
    display: table-cell;
    text-align: center;
    vertical-align: middle;
}
</style>
""")

built_landfills_for_plot1 = built_landfills.copy()
all_vars = model_question1b.getVars()
values = model_question1b.getAttr("X", all_vars)
names = model_question1b.getAttr("VarName", all_vars)
optimal_solution_for_plot1 = dict(zip(names, values))

fig, ax = plt.subplots(figsize=(5.,5.),dpi = 100)
for i in built_landfills_for_plot1:
    for j in centers:
        index_name = "xcl["+ j + ',' + i + ']'
        if optimal_solution_for_plot1[index_name] > 1e-6:
            x_coordinates = [landfill_locations.loc[i,'x'], center_
            y_coordinates = [landfill_locations.loc[i,'y'], center_
            ax.plot(x_coordinates, y_coordinates, color='black',lin
ax.scatter(center_locations['x'], center_locations['y'], color='blu
ax.scatter(landfill_locations.loc[built_landfills,'x'], landfill_lo
ax.legend()
ax.set_xlim(0, 100);
ax.set_ylim(0, 100);
ax.set_xlabel('x-coordinate (Longitude), in miles')
ax.set_ylabel('y-coordinate (Latitude), in miles')
plt.show()
```





Examine the sites of the landfills chosen in the optimal solution and the transportation flows from the centers to the landfills. What do you observe? What might you want to bring to the attention of your client?

Four landfills are selected in the optimal solution, namely Lake Lorraine, Muhlenberg Park, Weiser, and Renfrow. The weekly costs of waste transportation from the 50 town centers to the four landfills are calculated to be $415,035, which is significantly less than the planned costs of $560,000 if three landfills - Lake Lorraine, Muhlenberg Park, and Renfrow are selected. However, our current model does not include the cost of landfill construction, which might mitigate the cost saved by the optimal transportation.

Question 1(c): A new model that includes the costs of landfills

Recall that the EPA indicated that they might be able to provide direct funding for the construction of up to four landfills, depending on the EPA's own budget and also depending on well Ogemaw County can make the case that this would be an efficient use of such federal funds. But what is the most cost-effective number of landfills to build? Clearly, building more landfills requires more capital; but building more landfills should lower overall waste transportation costs the transportation logistics are efficiently managed.

While the EPA funds might be "free" to Ogemaw County (if they receive them at all), nevertheless the possibility of receiving these funds hinges on the making the case that the funds will be used efficiently, which includes designing and operating the most cost-effective waste operations. KSA has lots of experience in estimating capital costs of landfills and in the logistics operations of waste management. While the capital outlay for a landfill is huge, the outlay is typically financed with a tax-free bond. For Ogemaw County and at current bond rates, KSA estimates that the weekly cost of servicing a bond on one newly-constructed landfill is $8,750 per week.

In this question, we ask you to modify the model presented in question 1(b) to solve for the most cost-efficient overall solution that optimizes the number of landfills and includes both transportation costs and the costs of servicing the bonds for constructing landfills.

To make things more efficient for you, in the following code we have defined the decision variables and the constraints for you. You will notice that we *do not* have a constraint on how many landfills we can build. You will also notice that we have not (yet) defined the objective function.

In [17]:
```
# calculate big-M value
bigM = max(center_info["waste tonnage"])

# Create the model object
model_question1c = gp.Model("question1c_model")

# Add the decision variables
xcl = model_question1c.addVars(centers, landfills, name="xcl")
z = model_question1c.addVars(landfills, vtype=GRB.BINARY, name="z")

# Waste collection constraints:
# We must transport all waste away from each center
collect_waste_constraint = model_question1c.addConstrs(
    (sum(xcl[i, j] for j in landfills) == center_info.loc[i, 'waste
    name="collect_waste")

# We can only transport waste to landfill locations that have been 
bibig_M_constraint = model_question1c.addConstrs(
    (xcl[i, j] <= bigM * z[j] for i in centers for j in landfills),
    name="big_M")
```

In the cell below, modify the objective function by adding the weekly cost of servicing the bonds.

> Recall that when you see the phrase "## YOUR CODE HERE ##" anywhere in the notebook, **YOU WILL NEED TO REPLACE THIS PHRASE WITH YOUR CODE** and then execute the cell.

In [23]:
```
# Construct the modified objective function

objective_function = model_question1c.setObjective(
    0.97 * sum(sl_distance_centers_landfills.loc[i, j] * xcl[i, j]f
    + 8750 * sum(z[j] for j in landfills),
    GRB.MINIMIZE)
```

Solve the new optimization model.

In [24]:
```
# Solve the new model
model_question1c.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (mac64[rosetta2])

CPU model: Apple M2
Thread count: 8 physical cores, 8 logical processors, using up to
8 threads

Optimize a model with 800 rows, 765 columns and 2250 nonzeros
Model fingerprint: 0x0817de22
Variable types: 750 continuous, 15 integer (15 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+03]
  Objective range  [1e+00, 9e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [2e+02, 1e+03]

MIP start from previous solve produced solution with objective 428
684 (0.01s)
Loaded MIP start from previous solve with objective 428684

Presolve time: 0.00s
Presolved: 800 rows, 765 columns, 2250 nonzeros
Variable types: 750 continuous, 15 integer (15 binary)

Root relaxation: cutoff, 26 iterations, 0.00 seconds (0.00 work un
its)

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

     0     0     cutoff    0      428684.390 428684.390  0.00%
-    0s

Explored 1 nodes (26 simplex iterations) in 0.02 seconds (0.00 wor
k units)
Thread count was 8 (of 8 available processors)

Solution count 1: 428684

Optimal solution found (tolerance 1.00e-04)
Best objective 4.286843896151e+05, best bound 4.286843896151e+05,
gap 0.0000%
```

Which landfills are built in the new model solution?

In [25]:
```
built_landfills = []
for j in landfills:
    if z[j].x >= 0.9999:
        print("Build landfill at", j)
        built_landfills.append(j)
```

```
Build landfill at Hurt
Build landfill at Lake Lorraine
Build landfill at Muhlenberg Park
Build landfill at Ottumwa
Build landfill at Weiser
Build landfill at Anoka
Build landfill at Renfrow
```

What is the weekly *total* cost of the new solution?

Show your answer by running a line of code, replacing `YOUR CODE HERE`, running that cell and showing the output.

Hint: Click here to see how we did this for question 1(a)

In [26]:
```
model_question1c.ObjVal
```

Out[26]:
```
428684.3896150577
```

Why is the optimal objective function value higher than it was in the model in part 1(b)?

The optimal objective function value is higher than it was in the previous model, since the weekly cost of serving landfills is included into our consideration.

What is the weekly *transportation* cost of the new solution?

Note that the transportation cost does not include the cost of building the landfills, just the cost of transporting the waste.

In [27]:
```
total_transportation_cost = 0.97 * sum(sl_distance_centers_landfill
print("Total weekly cost of transportation is: $ %f" % (total_trans
```

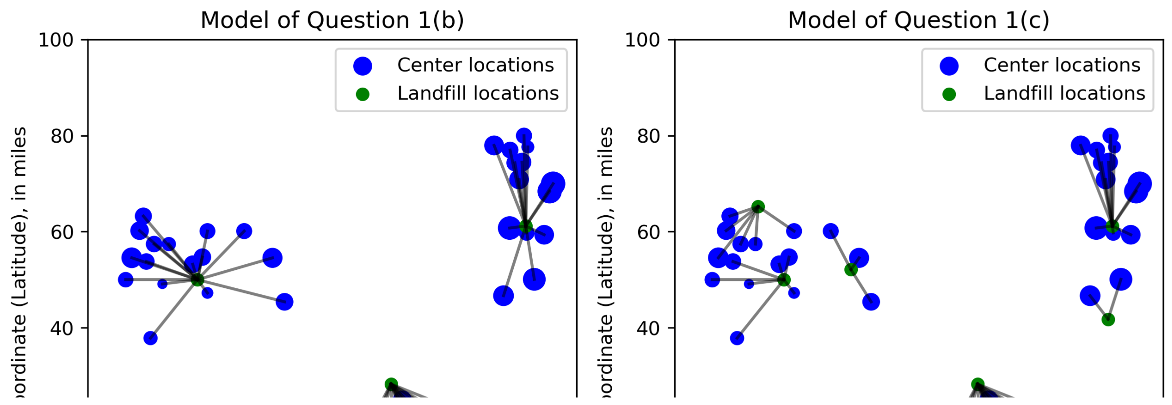
```
Total weekly cost of transportation is: $ 367434.389615
```

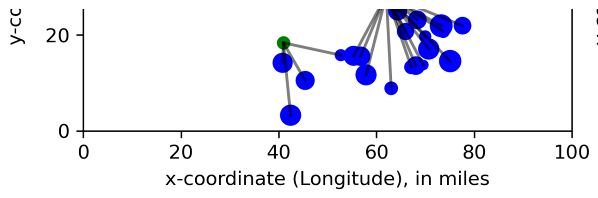
Now let us plot the locations of the landfills chosen and transportation flows from the centers to the landfills.

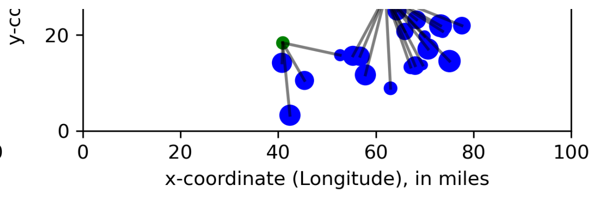
You do not need to understand the following code--it is just for plotting purposes.

In [28]:

```
built_landfills_for_plot2 = built_landfills.copy()
all_vars = model_question1c.getVars()
values = model_question1c.getAttr("X", all_vars)
names = model_question1c.getAttr("VarName", all_vars)
optimal_solution_for_plot2 = dict(zip(names, values))

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10, 4.5),

for i in built_landfills_for_plot1:
    for j in centers:
        index_name = "xcl["+ j + ',' + i + ']'
        if optimal_solution_for_plot1[index_name] > 1e-6:
            x_coordinates = [landfill_locations.loc[i,'x'], center_
            y_coordinates = [landfill_locations.loc[i,'y'], center_
            ax1.plot(x_coordinates, y_coordinates, color='black',li
ax1.scatter(center_locations['x'], center_locations['y'], color='bl
ax1.scatter(landfill_locations.loc[built_landfills_for_plot1,'x'],
ax1.legend()
ax1.set_xlim(0, 100);
ax1.set_ylim(0, 100);
ax1.set_xlabel('x-coordinate (Longitude), in miles')
ax1.set_ylabel('y-coordinate (Latitude), in miles')
ax1.set_title('Model of Question 1(b)')


for i in built_landfills_for_plot2:
    for j in centers:
        index_name = "xcl["+ j + ',' + i + ']'
        if optimal_solution_for_plot2[index_name] > 1e-6:
            x_coordinates = [landfill_locations.loc[i,'x'], center_
            y_coordinates = [landfill_locations.loc[i,'y'], center_
            ax2.plot(x_coordinates, y_coordinates, color='black',li
ax2.scatter(center_locations['x'], center_locations['y'], color='bl
ax2.scatter(landfill_locations.loc[built_landfills_for_plot2,'x'],
ax2.legend()
ax2.set_xlim(0, 100);
ax2.set_ylim(0, 100);
ax2.set_xlabel('x-coordinate (Longitude), in miles')
ax2.set_ylabel('y-coordinate (Latitude), in miles')
ax2.set_title('Model of Question 1(c)')
plt.show()
```







Now examine the sites of the landfills chosen in the optimal solution and the transportation flows from the centers to the chosen landfills. What are the key differences between your new optimal solution and the previous optimal solution in question 1(b)? Why is the solution different from that in question 1(b)? What might you want to recommend to your client based on your evaluation?

In the new optimal solution, two more landfills are selected, compared to the previous optimal colution, and the waste transportation routes of five town centers are reallocated to the addtional two landfills. When considering the weekly costs of landfill maintenance, the optimal function will compare the increased costs of additional landfills with the decreased costs of transportation, and generalized the optimal solution. In summary, I would recommend my client to implement the updated optimal solution, taking into account the trade-off between landfill maintenance costs and transportation costs for the most cost-effective and efficient waste management strategy.

Question 1(d): Modifying the model using more realistic road distances

The two models constructed so far in questions 1(b) and 1(c) use the computed distances between town/city centers and landfills using the straight-line distance $[\mathrm{DCL}]_{ij}$ between a center i and a landfill site j given by the well-known formula:

$$[\mathrm{DCL}]_{ij} := \sqrt{|\text{x-coord. of center } i \ - \ \text{x-coord. of landfill } j|^2 + |\text{y-coord. of center } i$$

However, as a practical matter, waste transportation trucks do not travel straight-line distances, as they must travel on government-approved roads and highways. In many parts of the world, and in particular in areas that are more recently developed, most of the roads are oriented either north-south or east-west. For example, here is a road-map for a portion of eastern Michigan in the United States.

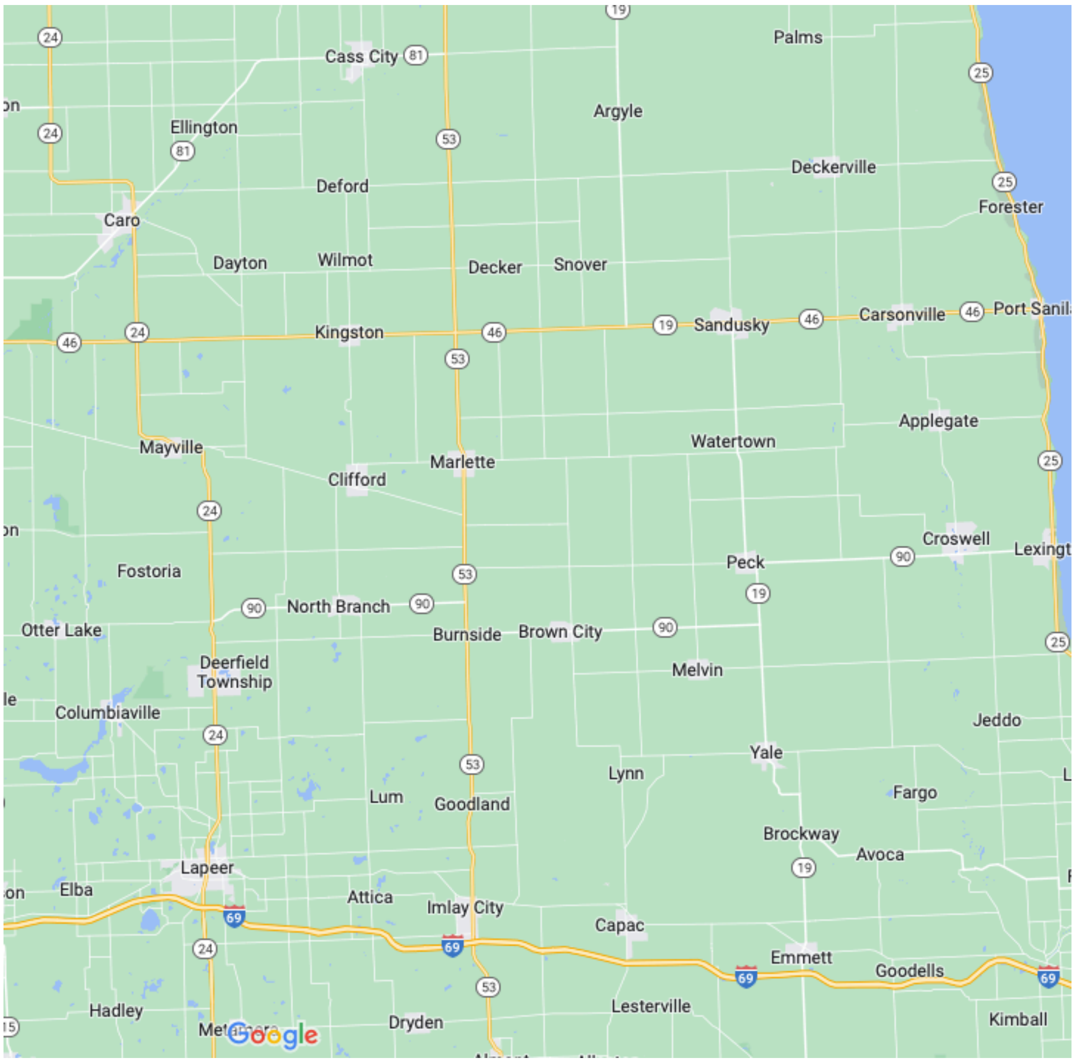


Figure 3: Road map of a portion of eastern Michigan, where most roads are oriented either north-south or east-west.

For such regions, it is more accurate to estimate travel distances instead using the formula:

$$[\text{DCL}]_{ij} := |\text{x-axis of center } i \ - \ \text{x-axis of landfill } j| + |\text{y-axis of center } i \ - \ \text{y-axis}$$

Notice that this formula for computing distance is the sum of the horizontal and vertical distances between the points, much like in Manhattan or in any region whose roads are mostly oriented either north-south or east-west. In fact, this way of computing distance is usually referred to as the "*Manhattan distance*".

Let us therefore change the way we compute distances between centers and landfills. Let us compute all distances between centers and landfills using the Manhattan distance instead of the straight-line distance.

In [29]:
```
# compute the ``Manhattan'' distances between all centers and all l
manhattan_distance_centers_landfills = pd.DataFrame(index = centers
for i in centers:
    for j in landfills:
        center_x = center_locations.loc[i,"x"]
        center_y = center_locations.loc[i,"y"]
        landfill_x = landfill_locations.loc[j,"x"]
        landfill_y = landfill_locations.loc[j,"y"]
        distance = np.abs(center_x - landfill_x) + np.abs(center_y 
        manhattan_distance_centers_landfills.loc[i,j] = distance
```

As usual, let us take a quick look to check that things look right.

In [30]:
```
manhattan_distance_centers_landfills.head().astype(float).round(2)
```

Out[30]:

	Avondale Estates	Hurt	Lake Lorraine	Muhlenberg Park	Arcadia	Ottumwa	Weiser	Anoka	Boyki
Sioux Falls	23.16	16.49	45.36	4.81	99.57	69.70	56.48	25.33	48.0
Apison	21.58	19.65	58.22	12.17	112.43	82.56	69.34	12.47	60.9
Grandin	29.13	25.82	64.39	14.22	118.60	88.73	75.51	16.60	67.1
Southern Gateway	26.78	24.85	63.42	13.25	117.63	87.76	74.54	8.37	66.1
Seattle	30.52	23.85	56.44	8.05	110.65	80.78	67.56	17.99	59.1

Are you curious how different the Manhattan distances are, compared to the straight-line distances? Let us take a look!

In [31]:
```
sl_distance_centers_landfills.head().astype(float).round(2)
```

Out[31]:

	Avondale Estates	Hurt	Lake Lorraine	Muhlenberg Park	Arcadia	Ottumwa	Weiser	Anoka	Boyki
Sioux Falls	17.50	12.61	33.25	3.44	72.92	64.47	42.05	19.42	44.3
Apison	16.91	14.13	44.88	10.32	80.11	66.80	49.20	8.96	47.2
Grandin	28.45	24.20	45.76	11.10	87.01	77.67	56.15	12.55	57.5
Southern Gateway	24.02	20.25	46.03	9.43	84.97	73.81	53.96	7.84	53.9
Seattle	25.72	21.07	40.07	7.22	81.90	73.82	51.18	16.23	53.6

We can see some significant differences. For example, the distance from Sioux Falls to Bazile Mills has a straight-line distance of 71 miles but the Manhattan distance is 40% longer!

Now we need to modify the optimization model from question 1(c), to account for the different distances between centers and landfill sites using the Manhattan distances. You will need to replace the straight-line distances with Manhattan distances in the objective function of the model.

As we did for you in Question 1(c), we have defined the decision variables and the constraints for you.

In [32]:
```
# calculate big-M value
bigM = max(center_info["waste tonnage"])

# Create the model object
model_question1d = gp.Model("question1d_model")

# Add the decision variables
xcl = model_question1d.addVars(centers, landfills, name="xcl")
z = model_question1d.addVars(landfills, vtype=GRB.BINARY, name="z")

# Waste collection constraints:
# We must transport all waste away from each center
collect_waste_constraint = model_question1d.addConstrs(
    (sum(xcl[i, j] for j in landfills) == center_info.loc[i, 'waste

# We can only transport waste to landfill locations that have been
bibig_M_constraint = model_question1d.addConstrs(
    (xcl[i, j] <= bigM * z[j] for i in centers for j in landfills),
```

In the cell below, modify the objective function (`YOUR CODE HERE`) by replacing the straight-line distances with Manhattan distances.

In [33]:
```
# Construct the modified objective function

objective_function = model_question1d.setObjective(
    0.97 * sum(manhattan_distance_centers_landfills.loc[i,j]*xcl[i,
    + 8750 * sum(z[j] for j in landfills),
    GRB.MINIMIZE)
```

Solve the new optimization model

In [34]:
```
# Solve the new model
model_question1d.optimize()
```

```
Gurobi Optimizer version 10.0.1 build v10.0.1rc0 (mac64[rosetta2])

CPU model: Apple M2
Thread count: 8 physical cores, 8 logical processors, using up to
8 threads

Optimize a model with 800 rows, 765 columns and 2250 nonzeros
Model fingerprint: 0xf2342a66
Variable types: 750 continuous, 15 integer (15 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+03]
  Objective range  [2e+00, 9e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [2e+02, 1e+03]
Found heuristic solution: objective 2387738.3519
Presolve time: 0.01s
Presolved: 800 rows, 765 columns, 2250 nonzeros
Variable types: 750 continuous, 15 integer (15 binary)

Root relaxation: objective 5.228181e+05, 21 iterations, 0.00 secon
ds (0.00 work units)

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

*    0     0               0    522818.05780 522818.058  0.00%
-    0s

Explored 1 nodes (21 simplex iterations) in 0.03 seconds (0.00 wor
k units)
Thread count was 8 (of 8 available processors)

Solution count 2: 522818 2.38774e+06

Optimal solution found (tolerance 1.00e-04)
Best objective 5.228180578000e+05, best bound 5.228180578000e+05,
gap 0.0000%
```

Which landfills are built in the new model solution?

In [35]:
```
built_landfills = []
for j in landfills:
    if z[j].x >= 0.9999:
        print("Build landfill at", j)
        built_landfills.append(j)
```

```
Build landfill at Avondale Estates
Build landfill at Lake Lorraine
Build landfill at Muhlenberg Park
Build landfill at Arcadia
Build landfill at Ottumwa
Build landfill at Weiser
Build landfill at Anoka
Build landfill at Renfrow
```

What is the weekly *total* cost of the new solution?

In [36]:
```
model_question1d.ObjVal
```

Out[36]:
```
522818.0578
```

Why do you think the optimal objective function value is higher than it was in the model in part 1(c)?

The optimal objective function value is higher since the transportation costs increase with the incrase in total distance.

What is the weekly *transportation* cost of the new solution?

In [37]:
```
total_transportation_cost = 0.97 * sum(manhattan_distance_centers_la
print("Total weekly cost of transportation is: $ %f" % (total_trans
```

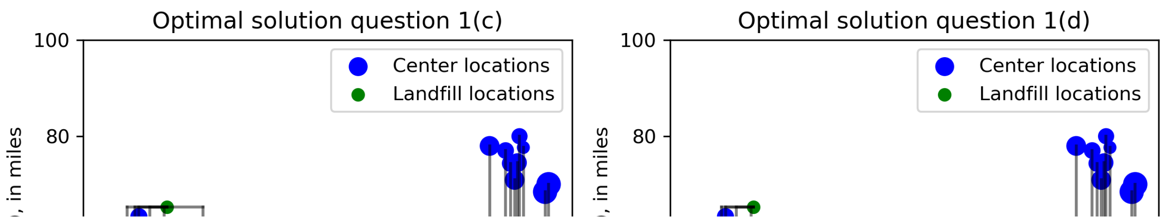
```
Total weekly cost of transportation is: $ 452818.057800
```

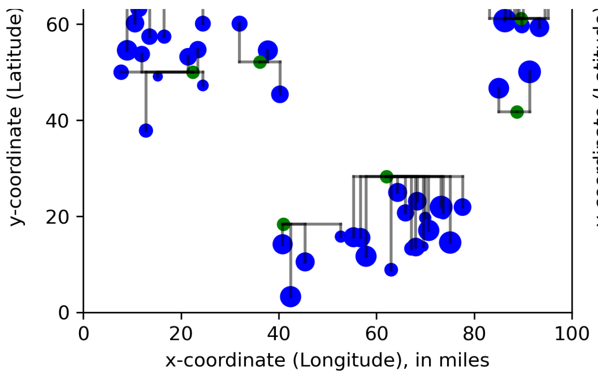
Now let us plot the locations of the landfills chosen and the transportation flows from the centers to the landfills.

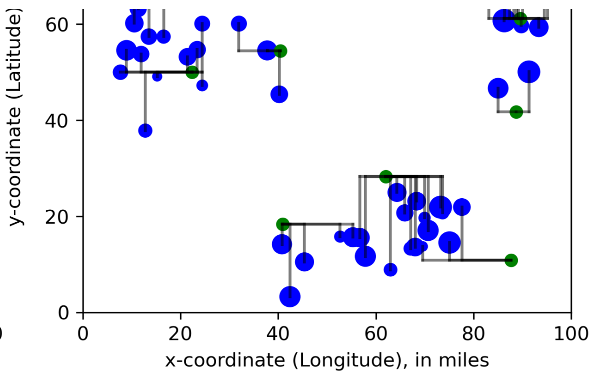
You do not need to understand the following code--it is just for making the plot.

In [38]:

```
built_landfills_for_plot3 = built_landfills.copy()
all_vars = model_question1d.getVars()
values = model_question1d.getAttr("X", all_vars)
names = model_question1d.getAttr("VarName", all_vars)
optimal_solution_for_plot3 = dict(zip(names, values))

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10, 4.5),

for i in built_landfills_for_plot2:
    for j in centers:
        index_name = "xcl["+ j + ',' + i + ']'
        if optimal_solution_for_plot2[index_name] > 1e-6:
            x_coordinates = [landfill_locations.loc[i,'x'], center_
            y_coordinates = [landfill_locations.loc[i,'y'], landfil
            ax1.plot(x_coordinates, y_coordinates, color='black',li
            x_coordinates = [center_locations.loc[j,'x'], center_lo
            y_coordinates = [landfill_locations.loc[i,'y'], center_
            ax1.plot(x_coordinates, y_coordinates, color='black',li
ax1.scatter(center_locations['x'], center_locations['y'], color='bl
ax1.scatter(landfill_locations.loc[built_landfills_for_plot2,'x'],
ax1.legend()
ax1.set_xlim(0, 100);
ax1.set_ylim(0, 100);
ax1.set_xlabel('x-coordinate (Longitude), in miles')
ax1.set_ylabel('y-coordinate (Latitude), in miles')
ax1.set_title('Optimal solution question 1(c)')

for i in built_landfills_for_plot3:
    for j in centers:
        index_name = "xcl["+ j + ',' + i + ']'
        if optimal_solution_for_plot3[index_name] > 1e-8:
            x_coordinates = [landfill_locations.loc[i,'x'], center_
            y_coordinates = [landfill_locations.loc[i,'y'], landfil
            ax2.plot(x_coordinates, y_coordinates, color='black',li
            x_coordinates = [center_locations.loc[j,'x'], center_lo
            y_coordinates = [landfill_locations.loc[i,'y'], center_
            ax2.plot(x_coordinates, y_coordinates, color='black',li
ax2.scatter(center_locations['x'], center_locations['y'], color='bl
ax2.scatter(landfill_locations.loc[built_landfills_for_plot3,'x'],
ax2.legend()
ax2.set_xlim(0, 100);
ax2.set_ylim(0, 100);
ax2.set_xlabel('x-coordinate (Longitude), in miles')
ax2.set_ylabel('y-coordinate (Latitude), in miles')
ax2.set_title('Optimal solution question 1(d)')
plt.show()
```







Now examine the sites of the landfills chosen in the optimal solution and transportation flows from the centers to the chosen landfills. What are the key differences between your new optimal solution and the previous optimal solution in question 1(c)? Why is the solution different from that in question 1(c)? What might you want to recommend to your client based on your evaluation?

Compared to the previous optimal solution in question 1(c), one additional landifill is chosen in the new optimal solution, and altogether 8 landfills are selected to minimize transportation costs and maintenance costs. These two solution are different in the measurement of distance between centers and landfills. The new model uses the sum of vertical and horizontal distance as the transportation distance, while the previous model uses the straight line distance. I would recommend my client to use the new optimal solution (i.e., selecting 8 landfills), since the calculation of distances in the new model is more realistic, which produces a more accurate prediction to the actual situation.