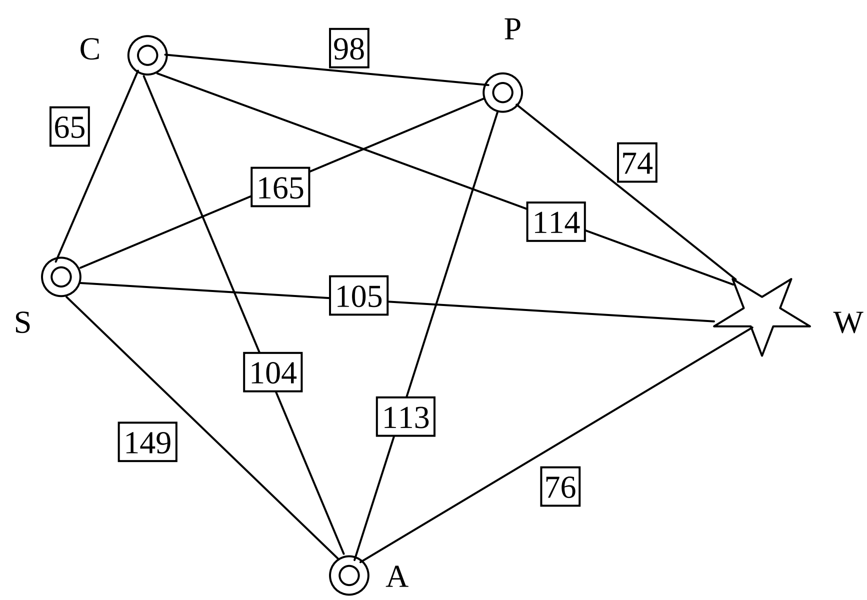
## Extension 1: Repetitive Nearest – Neighbor

One method used to improve upon the basic algorithm is called the "**Repetitive Nearest - Neighbor** " Algorithm.

**Repetitive Nearest - Neighbor Algorithm**

1. Select any node as a starting point. Apply the nearest-neighbor algorithm from that node.
2. Calculate the cost of that circuit.
3. Repeat the process using each of the other nodes as the starting point.
4. Choose the "best" Hamiltonian Circuit.

1. Apply the **Repetitive Nearest – Neighbor Algorithm** to the *original* Outel Semiconductor scenario. (Do *not* include Columbus.)




2. What is your criterion for the "best" circuit?

3. How does this improve upon your **Nearest – Neighbor** result?

4. When would the **Repetitive – Nearest Neighbor** method be useful?

5. Fill in the following chart:

| | **Brute Force** | **Nearest-Neighbor** | **Repetitive N-N** |
|---|---|---|---|
| **Strengths** | | | |
| **Weaknesses** | | | |
| **When to use** | | | |

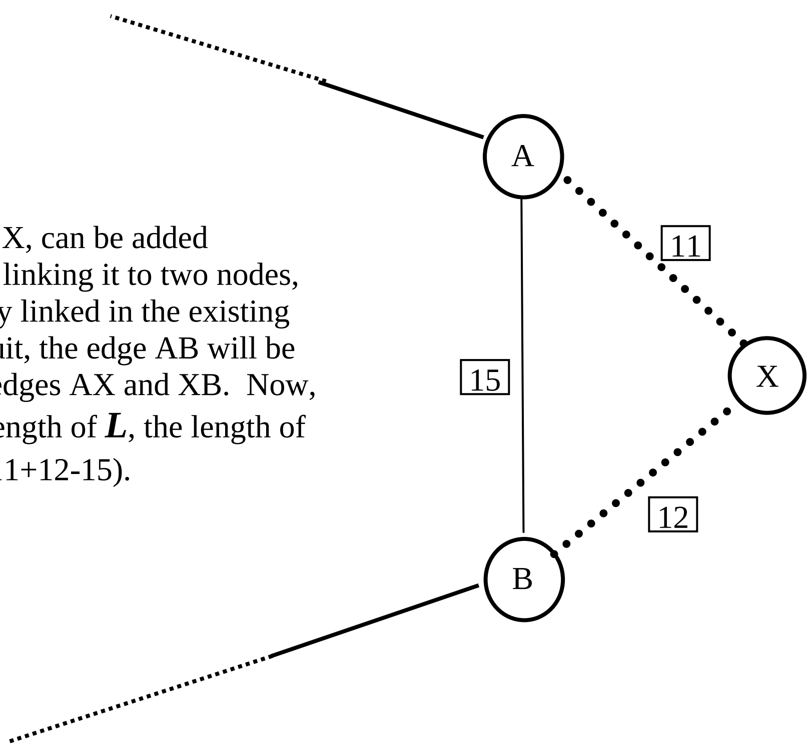
**Extension 2: Triangle Insertion**

A situation may occur when a node is added to the original network and you are forced to re-evaluate your circuit. An algorithm that allows you to efficiently tackle this problem is called the Triangle Insertion Algorithm. To use this algorithm, you begin by linking the new node to two nodes in the original circuit that are directly linked to each other and then eliminating the direct link. The length of the new circuit is found by adding the sum of the lengths of the two new links to the original circuit length and subtracting the length of the direct link that was eliminated. The process is repeated for each possible pair of directly linked nodes in the original circuit, and the shortest resulting circuit is selected.

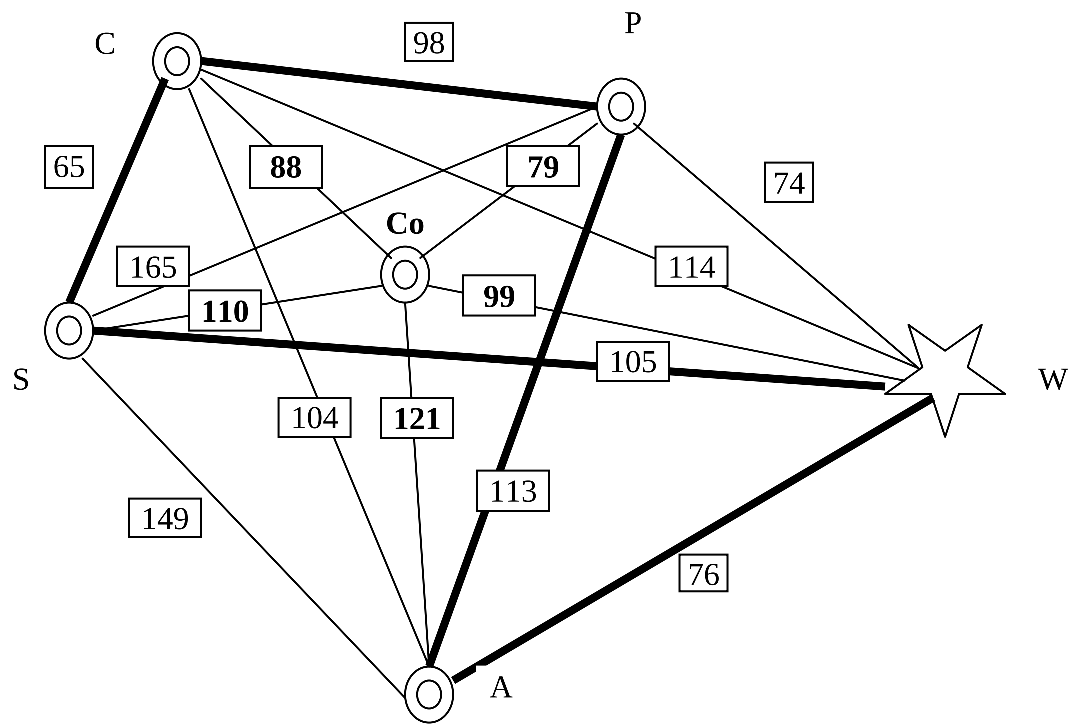
**Triangle Extension Algorithm**:

1. Begin with a Hamiltonian circuit for the original problem.
2. Choose a pair of directly *linked* nodes and link the new node to each of them, while eliminating the old link between them. Calculate the length of the new circuit by adding the sum of the lengths of the two new links, less the length of the old link they replace, to the original circuit length.
3. On the original network, repeat the process by inserting the new node between every pair of nodes that are directly linked in the original Hamiltonian circuit.
4. Choose the new circuit with the shortest length.

Example: A new node, X, can be added to an existing circuit by linking it to two nodes, A and B, that are already linked in the existing circuit. In the new circuit, the edge AB will be replaced by the pair of edges AX and XB. Now, if the old circuit had a length of $\boldsymbol{L}$, the length of the new circuit is $\boldsymbol{L}$ + (11+12-15).



The diagram below shows the addition of Columbus to the Outel recruitment network. Columbus and the associated travel costs are highlighted in boldface in the diagram. The original optimal circuit is also bolded.




Apply the Triangle Insertion Algorithm to the modification of the Outel Semiconductor problem that includes the addition of Columbus to the network.

## Homework Problems

### Problem 1

A traveling salesman is visiting several cities. To save time, the salesman chooses to travel by airplane. His home base is in Burlington, Vermont and he needs to visit each city exactly once. The cities are Burlington, VT, Portland, ME, Boston, MA, Newark, NJ and Albany, NY. The fares are as follows (the fare is the same for a return flight):

Burlington to Portland is $170

Burlington to Boston is $165

Burlington to Newark is $210

Burlington to Albany is $160

Portland to Albany is $150

Portland to Newark is $310

Portland to Boston is $60

Boston to Newark is $240

Boston to Albany is $120

Albany to Newark is $80

**Bu**rlington
**P**ortland
**A**lbany
**Bo**ston
**N**ewark

1. Use a map of the United States to make a visual representation of the route.

2. Find the Hamiltonian circuit with the lowest cost using the **Brute Force** algorithm. Keep track of the time it takes to execute this algorithm. (Computations may be done with a computer or calculator.)

3. Now use the **Nearest - Neighbor** algorithm to find a solution. Keep track of the time it takes to execute this algorithm.

4. Compare the results and effort needed with the two algorithms.

5. Which method would you choose? Why?

6. How would the situation change if you had 10 cities to visit?

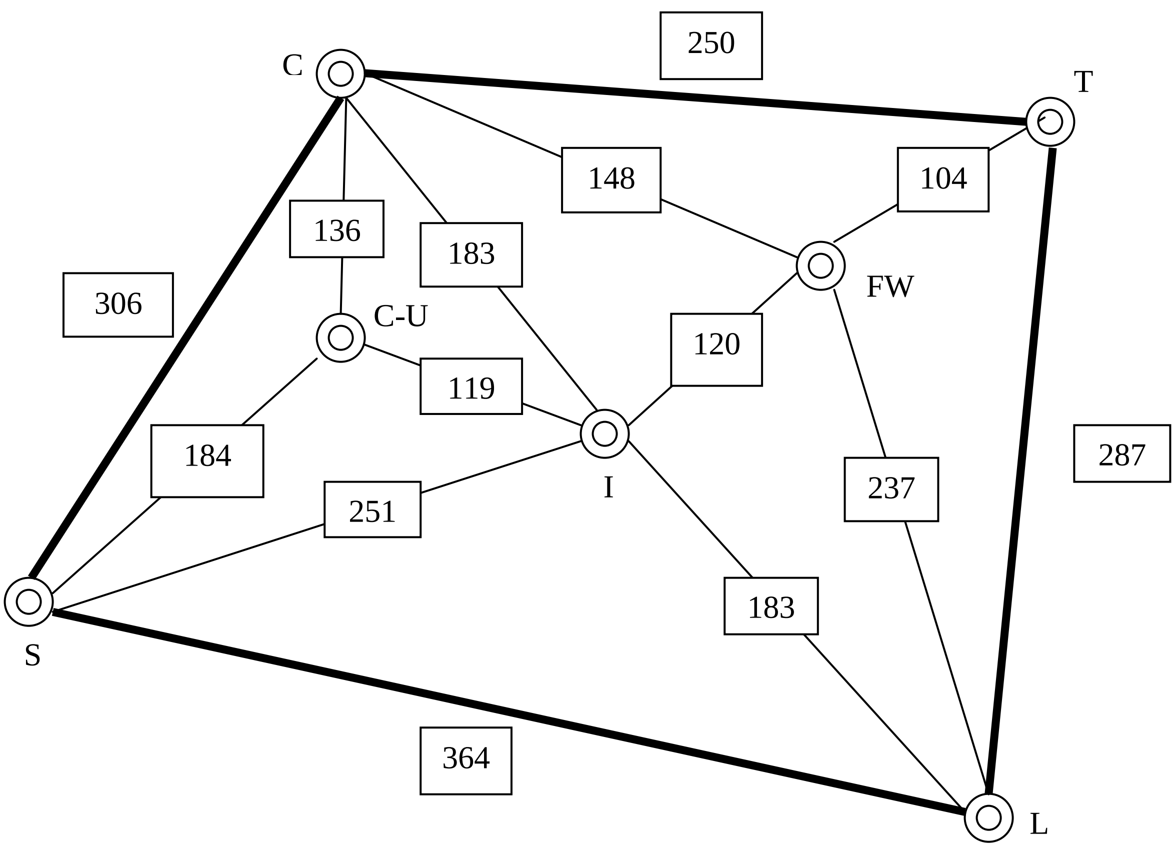
**Problem 2**

Ms I. Care has just been hired by the Global Insurance Agency to manage the Midwest Regional Office in Chicago, IL. At present the region has offices in Chicago, Toledo, OH, Lexington, KY, and St. Louis, MO. Ms. Care must visit each office once each quarter. Beginning in Chicago, she drives to Toledo, then on to Lexington, St. Louis, and back to Chicago, as shown on the map below. The company plans to add representatives in Indianapolis, IN, Champaign—Urbana, IL, and Ft. Wayne, IN. The plan is to add one city in the order listed during each of the next three quarters so the region will be complete by the end of the year.

a) Use the Triangle Extension Algorithm to find the best way Ms. Care can insert Indianapolis into the existing circuit given below.

b) Next, use the Triangle Extension Algorithm to find the best way to insert Champaign—Urbana into the circuit you found in answer to part (a).

c) Finally, use the Triangle Extension Algorithm to find the best way to insert Ft. Wayne into the circuit you found in answer to part (b).



## Solutions Extensions

### Extension 1: Repetitive Nearest – Neighbor

Other methods are used to improve upon our two Algorithms. We can begin by applying what is called the "**Repetitive Nearest - Neighbor** " Algorithm.

> **Repetitive Nearest - Neighbor Algorithm**
>
> 1. Select any node as a starting point. Apply the nearest-neighbor algorithm from that node.
> 2. Calculate the cost of that circuit.
> 3. Repeat the process using each of the other nodes as the starting point.
> 4. Choose the "best" Hamiltonian Circuit.

1. Apply the **Repetitive Nearest – Neighbor Algorithm** to the *original* Outel Semiconductor scenario. (Do *not* include Columbus.)

   **a) Start at "W" WPCSAW=74+98+65+149+76 = 462**
   **b) Start at "P" PWACSP= 74+76+104+65+165 = 484**
   **c) Start at "C" CSWPAC= 65+105+74+113+104 = 461 (best solution)**
   **d) Start at "A" AWPCSA= 76+74+98+65+149 = 462**
   **e) Start at "S" SCPCWAS = 65+98+74+76+149 = 462**

   **Note: The cheapest solution is CSWPAC (which we can rewrite as WPACSW). Notice, the repetitive nearest neighbor algorithm starting at different nodes, produced circuits "a", "d" and "e" which were identical but were not optimal. In addition, if your starting city had been Pittsburgh and you applied just a nearest neighbor algorithm with no repetition, the generated solution would have been significantly worse than the optimal.**

2. What is your criterion for the "best" circuit?

   **We are using "cheapest" as the criterion for "best."**

   **Note to the teacher: In practice, a manager would need to weigh the cost of computer time and employee time used to find an optimal solution against the cost of the trip. It is not cost effective to spend $1000 to determine the "best" solution if the best solution saves only $100.**

3. Does the **Repetitive Nearest – Neighbor Algorithm** yield the best solution?

   **No, but the optimal solution is only $4 cheaper. Remember the optimal circuit was WSCPAW with a cost of $457.**

4. When would the **Repetitive – Nearest Neighbor** method be useful?

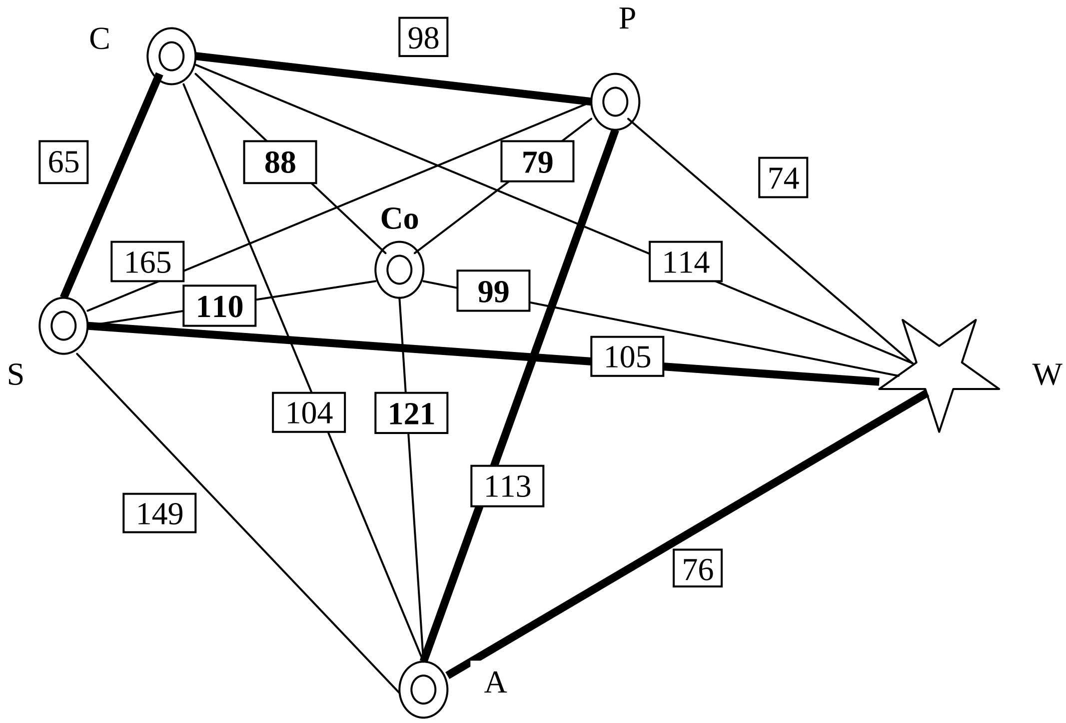
   **When you need a very good solution quickly. The repetitive nearest neighbor algorithm will usually provide a near optimal solution. Notice that if there are "n" nodes in the network, this algorithm requires repeating the nearest neighbor algorithm "n" times. Sometimes, as in this case, the improvement from 462 to 461 may not be worth the extra effort.**

5. Fill in the following chart:

| | **Brute Force** | **Nearest-Neighbor** | **Repetitive Nearest-Neighbor** |
|---|---|---|---|
| **Strengths** | Always produces optimal solution. | Very quick. Can provide quick result, even with many nodes. | Quicker than Brute Force and should provide better solution than Nearest-Neighbor. |
| **Weaknesses** | Time-consuming and may not be practical when there are many nodes. | Solution may be poor relative to optimal and dependent on starting point. | Solution is likely to be near optimal and is not dependent on starting point. |
| **When to use** | When costs are extremely important and saving even a small amount is critical. | When a route is needed immediately. | When you need a very good solution quickly. |

**Extension 2: Triangle Insertion Algorithm**

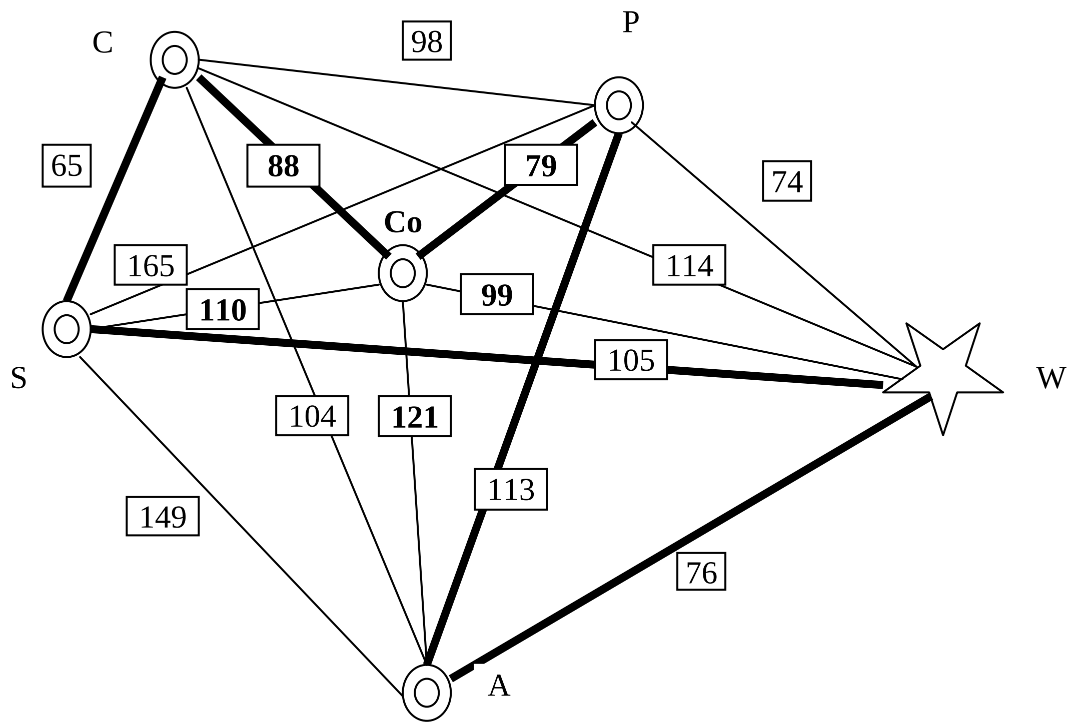
The diagram below shows the addition of Columbus to the Outel recruitment network. Columbus and the associated travel costs are highlighted in boldface in the diagram.




Apply the Triangle Insertion Algorithm to the modification of the Outel Semiconductor problem that includes the addition of Columbus to the network. Begin with the original best route highlighted in bold ➔ WSCPAW = 457

**Insert Co between W and S WCoSCPAW=457+(99+110-105) = 561**
**Insert Co between S and C WSCoCPAW=457+(110+88-65) = 590**
**Insert Co between C and P WSCCoPAW=457+(88+79-98) = 526 (best solution)**
**Insert Co between P and A WSCPCoAW=457+(79+121-113) = 544**
**Insert Co between A and W WSCPACoW=457+(121+99-76) = 601**

**Notice there are significant differences between the alternative insertion points. Even the second best insertion point is $18 or 3% worse than the best solution found by the Triangle Insertion Algorithm.**


C
98
P
65
88
79
74
Co
165
114
110
99
S
105
W
104
121
113
149
76
A

## Homework Answers
## Problem 1

A traveling salesman is visiting several cities. To save time, the salesman chooses to travel by airplane. His home base is in Burlington, Vermont and he needs to visit each city exactly once. The cities are Burlington, VT, Portland, ME, Boston, MA, Newark, NJ and Albany, NY.

The fares are as follows (the fare is the same for a return flight):

Burlington to Portland is $170

Burlington to Boston is $165

Burlington to Newark is $210

Burlington to Albany is $160

Portland to Albany is $150

Portland to Newark is $310

Portland to Boston is $60

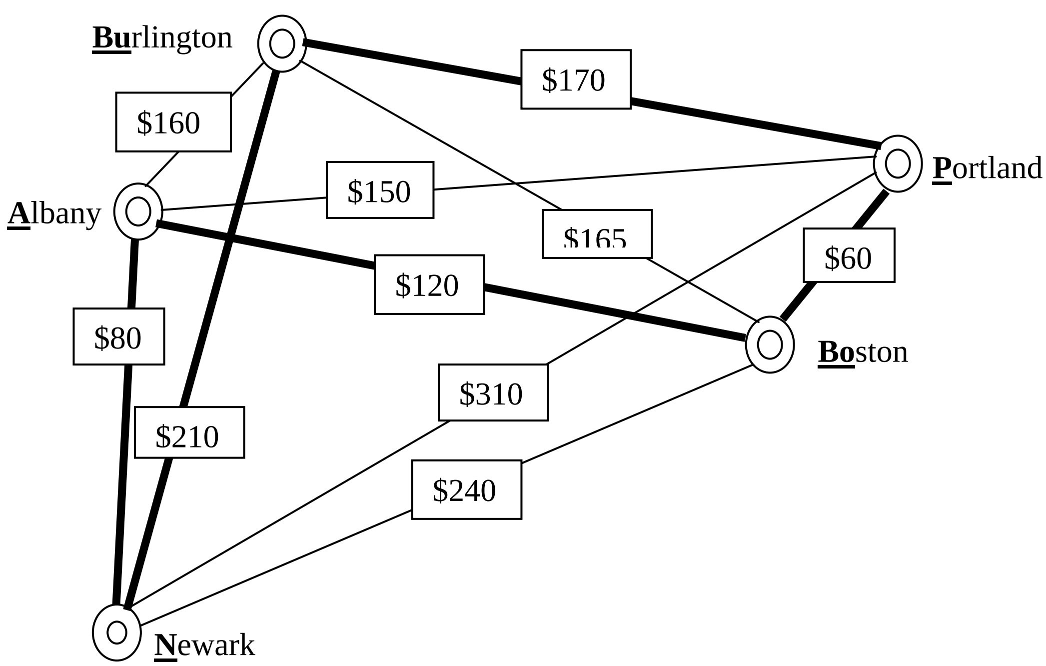
Boston to Newark is $240

Boston to Albany is $120

Albany to Newark is $80

**Bu**rlington
**P**ortland
**A**lbany
**Bo**ston
**N**ewark

1. Use a map of the United States to make a visual representation of the network. (Note: the optimal solution is also shown.)



2. Find the Hamiltonian circuit with the lowest cost using the **Brute Force** algorithm. Keep track of the time it takes to execute this algorithm. (Computations may be done with a computer or calculator.)

   **BuPABoNBu = 170 + 150 + 120 + 240 + 210 = $890**
   **BuPBoANBu = 170 + 60 + 120 + 80 + 210 = $640 (best solution)**
   **BuPANBoBu = 170 + 150 + 80 + 240 + 165 = $805**
   **BuPNABoBu = 170 + 310 + 80 + 120 + 165 = $845**
   **BuPNBoABu = 170 + 310 + 240 + 120 + 160 = $1000**
   **BuPBoNABu = 170 + 60 + 240 + 80 + 160 = $710**
   **BuABoPNBu = 160 + 120 + 60 + 310 + 210 = $860**
   **BuAPBoNBu = 160 + 150 + 60 + 240 + 210 = $820**
   **BuANPBoBu = 160 + 80 + 310 + 60 + 165 = $775**
   **BuNAPBoBu = 210 + 80 + 150 + 60 + 165 = $665**
   **BuNPABoBu = 210 + 310 + 150 + 120 + 165 = $955**
   **BuBoNPABu = 165 + 240 + 310 + 150 + 160 = $1025**

3. Now use the **Nearest - Neighbor** algorithm to find a solution. Keep track of the time it takes to execute this algorithm.
   The nearest neighbor algorithm generated the solution BuANBoPBu at a cost $710.

4. Compare the results and effort needed with the two algorithms.
   The nearest neighbor algorithm found the third best solution which is $70 or 10% worse than the optimal solution. It should have taken the students significantly longer to execute the brute force algorithm as compared to the nearest neighbor algorithm.

5. Which method would you choose? Why?
   Ask the students whether or not the extra effort was worth saving $70.

6. How would the situation change if you had 10 cities to visit?
   With the brute force algorithm, there would be 9!/2 = 181,440 circuits to evaluate. Thus, it would no longer be practical to use Brute Force if the computations are done by hand.

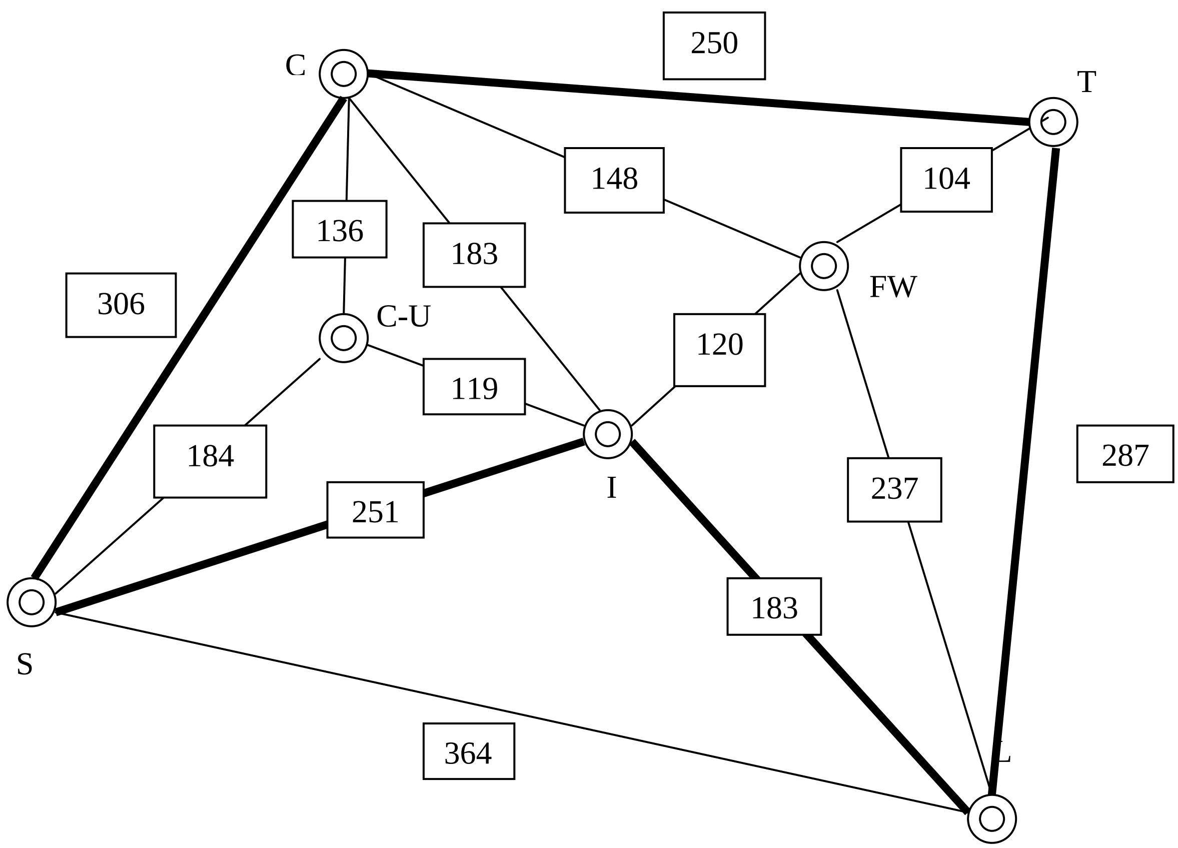
**Problem 2**

Ms I. Care has just been hired by the Global Insurance Agency to manage the Midwest Regional Office in Chicago, IL. At present the region has offices in Chicago, Toledo, OH, Lexington, KY, and St. Louis, MO. Ms. Care must visit each office once each quarter. Beginning in Chicago, she travels to Toledo, then on to Lexington, St. Louis, and back to Chicago, as shown on the map below. The company plans to add representatives in Indianapolis, IN, Champaign—Urbana, IL, and Ft. Wayne, IN. The plan is to add one city in the order listed during each of the next three quarters so the region will be complete by the end of the year.

a) Use the Triangle Extension Algorithm to find the best way Ms. Care can insert **I**ndianapolis into the existing circuit given above.

**CTLSC=250+287+364+306=1207**

C**I**TLSC**=1207+(183+120+104-250) = 1364**
CT**I**LSC**=1207+(120+104+183-287) = 1327**
CTL**I**SC**=1207+(183+251-364) = 1277 (best solution)**
CTLS**I**C**=1207+(251+183-306) = 1335**

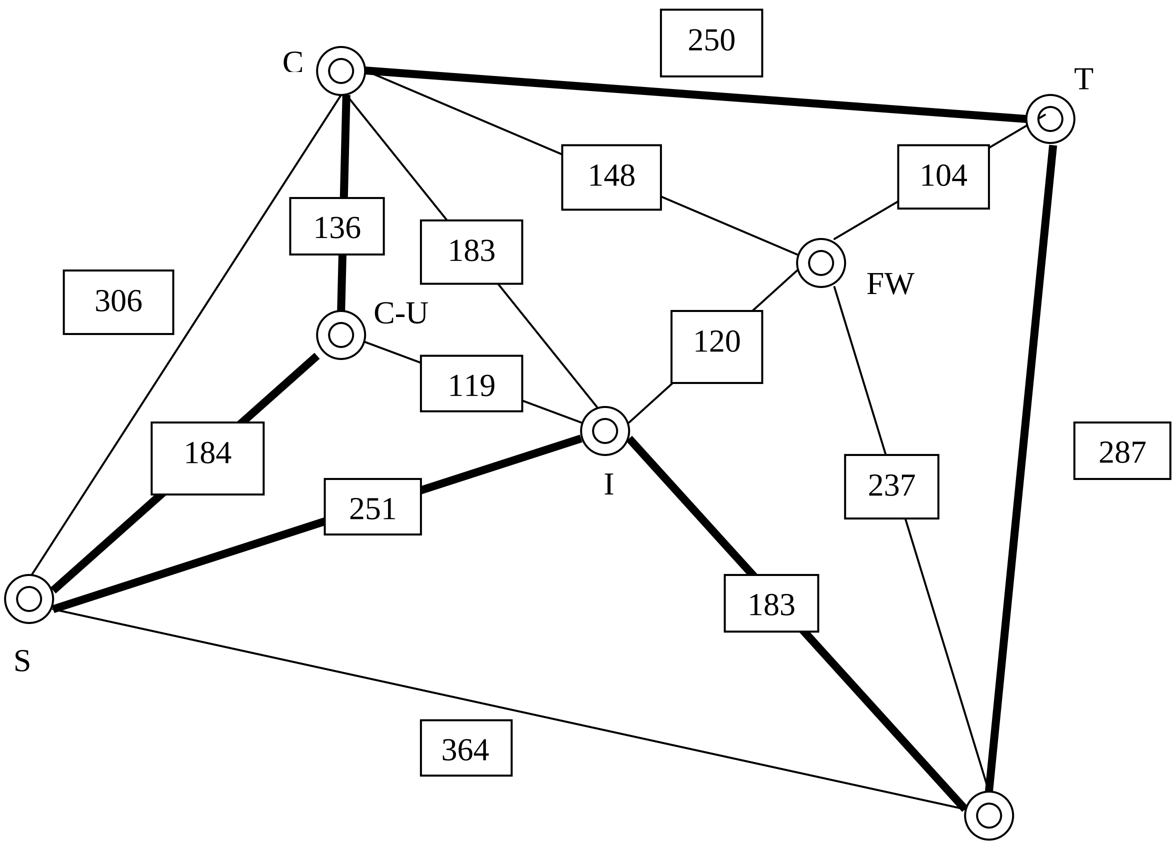


b) Next, use the Triangle Extension Algorithm to find the best way to insert Champaign—Urbana into the circuit you found in answer to part (a).

**CTLISC=1277**

CTLI**C-U**SC**=1277+(119+184-251) = 1329**
CTLIS**C-U**C**=1277+(184+136-306) = 1291 (best solution)**

**Note: Champaign—Urbana only has direct links to Indianapolis, St. Louis, and Chicago. Therefore, it may only be inserted into the previous circuit between Indianapolis and St. Louis or between St. Louis and Chicago.**



c) Finally, use the Triangle Extension Algorithm to find the best way to insert Ft. Wayne into the circuit you found in answer to part (b).

**CTLISC-UC=1291**

C**F**TLISC-UC**=1291+(148+104-250) = 1293 (best solution)**
CT**F**LISC-UC**=1291+(104+237-287) = 1345**
CTL**F**ISC-UC**=1291+(237+120-183) = 1465**

**Note: Due to its limited number of links, Ft. Wayne may only be inserted between Chicago and Toledo, Toledo and Lexington, or Lexington and Indianapolis.**