12. Distance Analysis

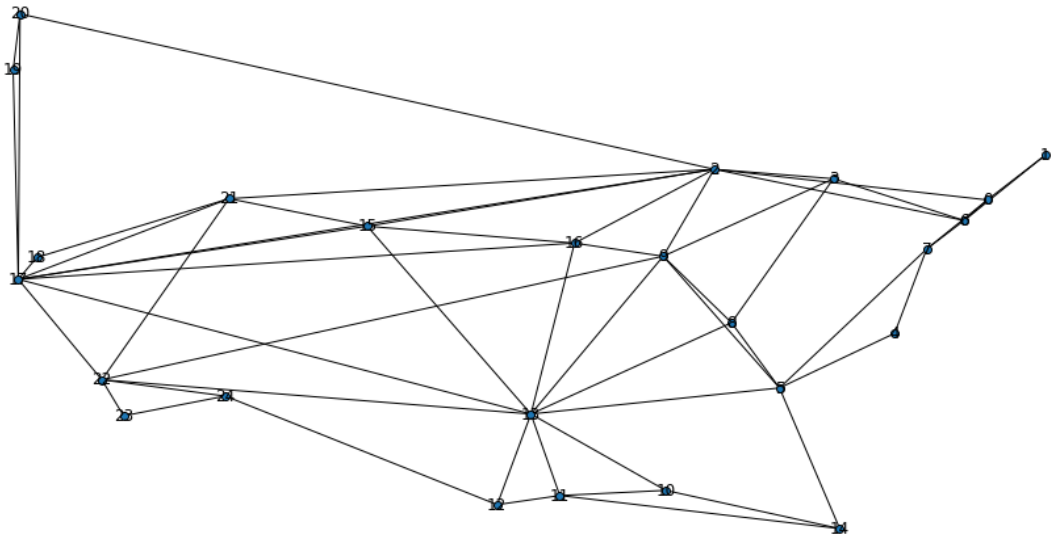
In [1]:

```
%matplotlib inline
import networkx as nx
import matplotlib.pylab as plt
import numpy as np
import netlab as nl
import pprint
pprint=pprint.pprint
#    from mpl_toolkits.basemap import Basemap
```

In [2]:

```
G = nx.read_gml('data/attmpls.gml', label='id')
plt.figure(1,figsize=(12,6))
layout = nl.absolute_layout(G)
nl.draw_atlas(G, pos=layout)
```

```
/opt/anaconda3/lib/python3.7/site-packages/networkx/drawing/nx_pylab.py:57
9: MatplotlibDeprecationWarning:
The iterable function was deprecated in Matplotlib 3.1 and will be removed
in 3.3. Use np.iterable instead.
  if not cb.iterable(width):
```



In [3]:

```
nx.eccentricity(G)
```

Out[3]:

```
{0: 4,
 1: 5,
 2: 3,
 3: 3,
 4: 4,
 5: 3,
 6: 4,
 7: 4,
 8: 3,
 9: 3,
 10: 5,
 11: 5,
 12: 5,
 13: 4,
 14: 4,
 15: 3,
 16: 3,
 17: 3,
 18: 4,
 19: 4,
 20: 4,
 21: 4,
 22: 4,
 23: 5,
 24: 5}
```

In [4]:

```
nx.radius(G)
```

Out[4]:

```
3
```

In [5]:

```
nx.diameter(G)
```

Out[5]:

```
5
```

All-pair average shortest path length

$$\bar{d} = \sum_{s,t \in V} \frac{d(s,t)}{n(n-1)}$$

In [6]:

```
d_bar = nx.average_shortest_path_length(G)
print ('d_bar =', d_bar)
```

```
d_bar = 2.3833333333333333
```

Average shortest path length for a node

$$\bar{d}(u) = \sum_{x \in V} \frac{d(u,x)}{n-1}$$

In [7]:

```
u = 8
plen = nx.shortest_path_length(G, source=u)
print ('the shortest distances from node %s:\n%s' % (u, plen))

pmean = np.sum (list(plen.values()))/float(G.number_of_nodes()-1)
print ('the mean shortest distance from node %s = %s' % (u, pmean))
```

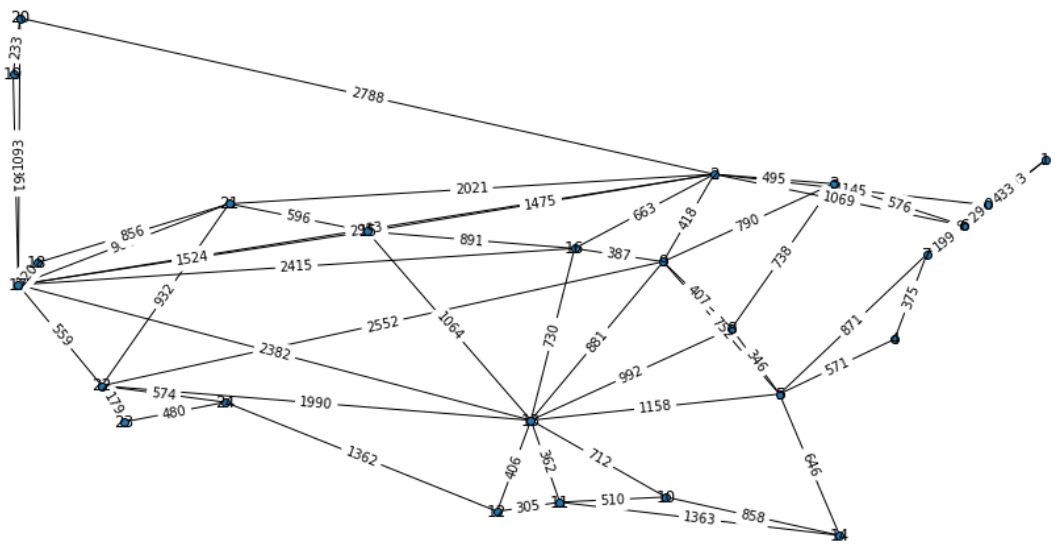
```
the shortest distances from node 8:
{8: 0, 3: 1, 5: 1, 9: 1, 13: 1, 2: 2, 6: 2, 4: 2, 7: 2, 14: 2, 16: 2, 22:
2, 10: 2, 11: 2, 12: 2, 15: 2, 17: 2, 0: 3, 20: 3, 21: 3, 1: 3, 23: 3, 24:
3, 18: 3, 19: 3}
the mean shortest distance from node 8 = 2.1666666666666665
```

In [8]:

```
px = list(nx.get_node_attributes(G, 'longitude').values())
py = list(nx.get_node_attributes(G, 'latitude').values())
for u,v in G.edges():
    G[u][v]['distance'] = int(nl.haversine((px[u],py[u]), (px[v],py[v])))

labels = nx.get_edge_attributes(G, 'distance')

plt.figure(1,figsize=(12,6))
nl.draw_atlas(G, pos=layout, edge_labels=labels)
```



In [9]:

```
# weighted graph
p1 = nx.average_shortest_path_length(G,weight='distance')
print (p1)
```

```
2203.4133333333334
```

In [22]:

```
p2 = nx.all_pairs_dijkstra_path_length(G, weight='distance')
e = nx.eccentricity(G, sp=dict(p2))
e
```

Out[22]:

```
{0: 4226,
 1: 4529,
 2: 3132,
 3: 3521,
 4: 4664,
 5: 4191,
 6: 4097,
 7: 4289,
 8: 3846,
 9: 3439,
 10: 4187,
 11: 3837,
 12: 3588,
 13: 3475,
 14: 4813,
 15: 2923,
 16: 3451,
 17: 4431,
 18: 4325,
 19: 4813,
 20: 4604,
 21: 3469,
 22: 4350,
 23: 4529,
 24: 4428}
```

In [23]:

```
nx.radius(G, e)
```

Out[23]:

```
2923
```

In [24]:

```
nx.diameter(G, e)
```

Out[24]:

```
4813
```

In [ ]: