13. Clustering

In [1]:

```
%matplotlib inline
import networkx as nx
import matplotlib.pylab as plt
import matplotlib.colors as colors
import matplotlib.cm as cm
import numpy as np
#import haversine as hsin
import netlab as nl
```

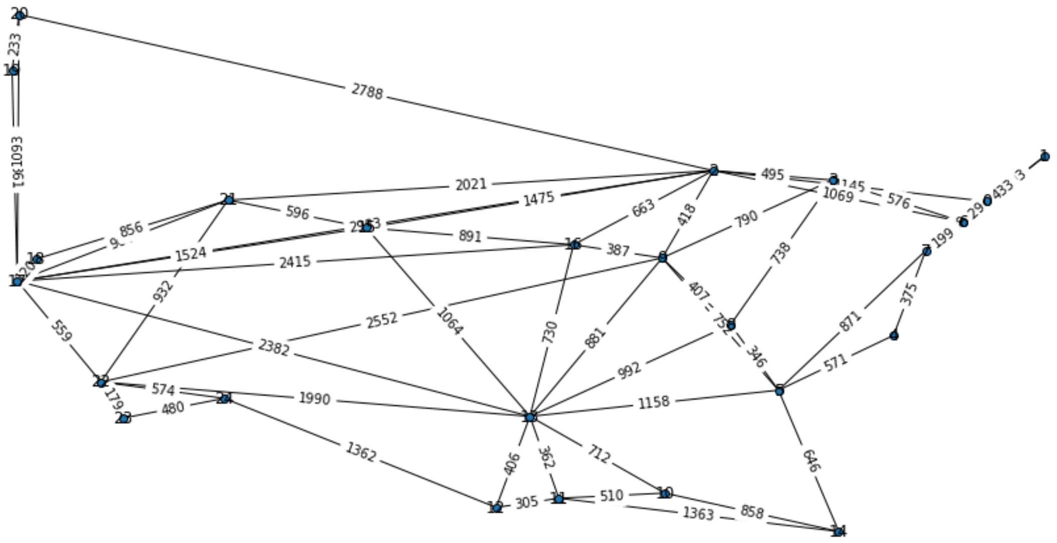
In [2]:

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

In [3]:

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

```
/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 [4]:

```
nx.clustering(G)
```

Out[4]:

```
{0: 0.5,
 1: 1.0,
 2: 0.2777777777777778,
 3: 0.5,
 4: 1.0,
 5: 0.26666666666666666,
 6: 0.4,
 7: 0.3333333333333333,
 8: 0.6666666666666666,
 9: 0.38095238095238093,
 10: 0.6666666666666666,
 11: 0.5,
 12: 0.3333333333333333,
 13: 0.24444444444444444,
 14: 0.3333333333333333,
 15: 0.7,
 16: 0.7,
 17: 0.3333333333333333,
 18: 1.0,
 19: 1.0,
 20: 0.6666666666666666,
 21: 0.5,
 22: 0.26666666666666666,
 23: 1.0,
 24: 0.3333333333333333}
```

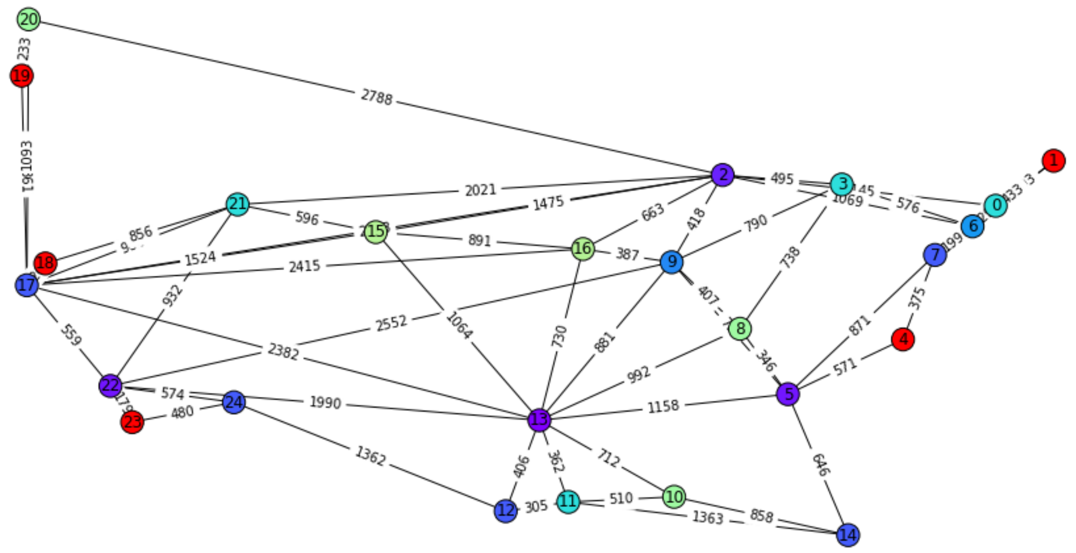
In [5]:

```
# for unweighted graphs
cc = list(nx.clustering(G).values())
print (cc)
```

```
[0.5, 1.0, 0.2777777777777778, 0.5, 1.0, 0.26666666666666666, 0.4, 0.33333
33333333333, 0.6666666666666666, 0.38095238095238093, 0.6666666666666666,
0.5, 0.3333333333333333, 0.24444444444444444, 0.3333333333333333, 0.7, 0.
7, 0.3333333333333333, 1.0, 1.0, 0.6666666666666666, 0.5, 0.26666666666666
666, 1.0, 0.3333333333333333]
```

In [6]:

```
# draw it with nodal colormap of clustering coefficient
plt.figure(1,figsize=(12,6))
#nl.draw_atlas(G, pos=layout, edge_labels=labels, node_cmap=cc.values())
nl.draw_atlas(G, pos=layout, edge_labels=labels, node_cmap=cc)
```



In [7]:

```
nx.average_clustering(G)
```

Out[7]:

```
0.5561269841269841
```

In [8]:

```
nx.transitivity(G)
```

Out[8]:

```
0.38372093023255816
```

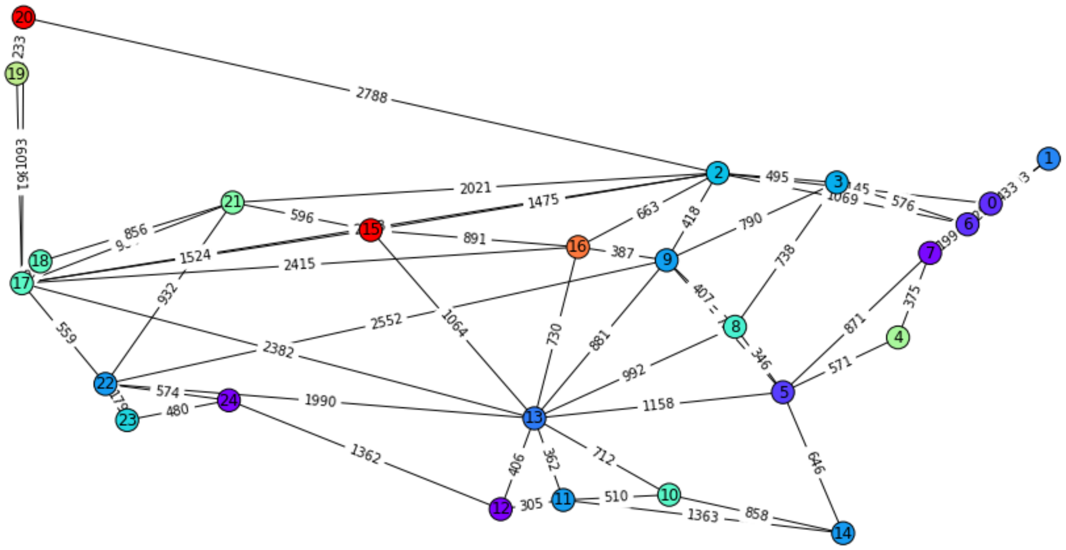
In [8]:

```
# for weighted graphs
cc = list(nx.clustering(G,weight='distance').values())
print (cc)
```

```
[0.05591033107255402, 0.08607033467786519, 0.1104065215351166, 0.102740249
48341072, 0.1915322557857827, 0.060233442280757554, 0.05610668939519705,
0.043288741981861445, 0.14169111357983996, 0.09601869009558837, 0.15087808
763587163, 0.0952865971409726, 0.039695106646073564, 0.08157634910569242,
0.09406107369318963, 0.299963391211655, 0.2590314648228229, 0.152006974936
61108, 0.15509248746825932, 0.20214972651224486, 0.30059344866731325, 0.17
510113841569974, 0.09371285444381662, 0.12293686486621636, 0.0409789549554
05454]
```

In [9]:

```
# draw it with nodal colormap of clustering coefficient
plt.figure(1,figsize=(12,6))
nl.draw_atlas(G, pos=layout, edge_labels=labels, node_cmap=cc)
```



In [10]:

```
nx.average_clustering(G,weight='distance')
```

Out[10]:

```
0.12828251561639273
```

no transivity is defined for weighted graph in networkx library

work-to-do