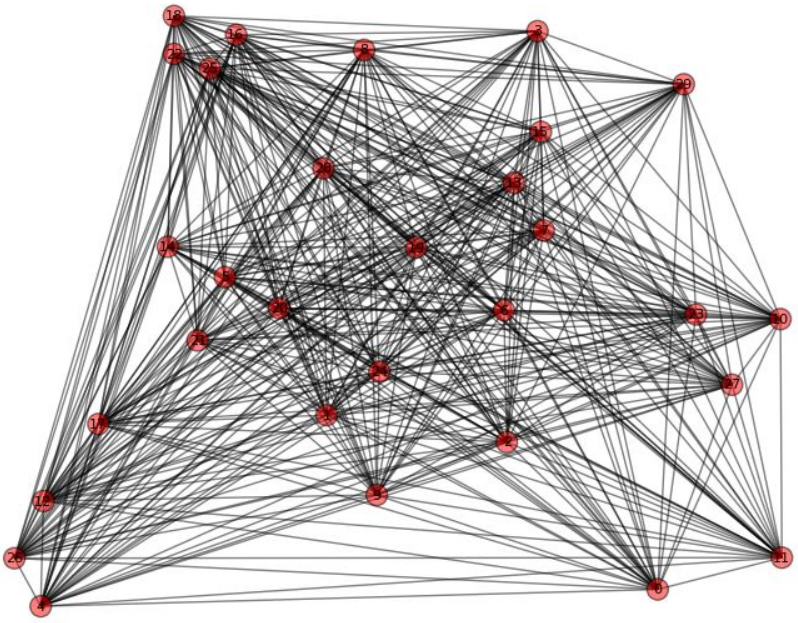
Árvores geradoras mínimas (MST)

Neste exercício foi implementado o algoritmo de MST do Prim, sendo este um dos mais utilizados para se encontrar MST's em dado grafo.
O grafo mapeado em questão é um grafo representando relações entre 30 cidades do mundo, o objetivo é gerar a árvore mínima deste grafo(imagem 1).



A função implementada foi a MST_prim, no qual recebe como parâmetro um grafo G, e retorna sua MST. Inicialmente as cores de todos os vértices pertencentes a G são setados como Brancos 'B', em seguida o atributo correspondente aos seus antecessores são setados como nil, ou vazio.
A função inicializa lambda de todos os vértices, exceto a raíz como infinito, enquanto a raíz é inicializada com 0.

```
nx.set_node_attributes(graph, 'B', 'color')

    # Inicializa o pi(vértice antecessor) dos vértices como vazio
    nx.set_node_attributes(graph, None, 'pi')

    #Inicializa o lambda de todo v e V != Raíz como infinito
```

```
nx.set_node_attributes(graph, float('inf'), 'λ')
node = 1
nx.set_node_attributes(graph, {node: 0}, 'λ')#Seta o lambda do primeiro nó com 0
```

Em seguida se ainda existir vértices com a cor Branca, isso é, ainda não visitados, todos os vértices vizinhos ao node atual da execução são selecionados, entre os vértices selecionados é escolhido o de menos lambda, assim é gerada uma nova distância w(u,v), e comparada se o lambda atual do vértice é menor ou não de w, se w for menor, são atualizados dados como seu lambda e seu antecessor.

```
for node in graph.neighbors(mNode[0]):
        node_info = (node, dict(nodes_info)[node])
        if node_info[1]['color'] == 'B':
            novaDistancia = graph.get_edge_data(mNode[0], node_info[0])['weight']
            # Verifica se o lambda atual é menor que w(u,v), se for nao atualiza.
            if novaDistancia< node_info[1]['λ']:
                nx.set_node_attributes(graph, {node_info[0]: novaDistancia}, 'λ')
                nx.set_node_attributes(graph, {node_info[0]: mNode[0]}, 'pi')

    #Marca o Node como visitado
    nx.set_node_attributes(graph, {mNode[0]: 'P'}, 'color')
    nodes_info = graph.nodes(data=True)
```

Quando todos os vértices são marcados como visitados a função retorna a MST resultante, no exemplo citado anteriomente, ao carregarmos a matriz de adjacência:

A = np.loadtxt('ha30_dist.txt').

E criarmos um grafo G a partir dela, chamando a função mst_prim(G) passando o grafo G como parâmetro, a função nos retorna a seguinte árvore mínima:

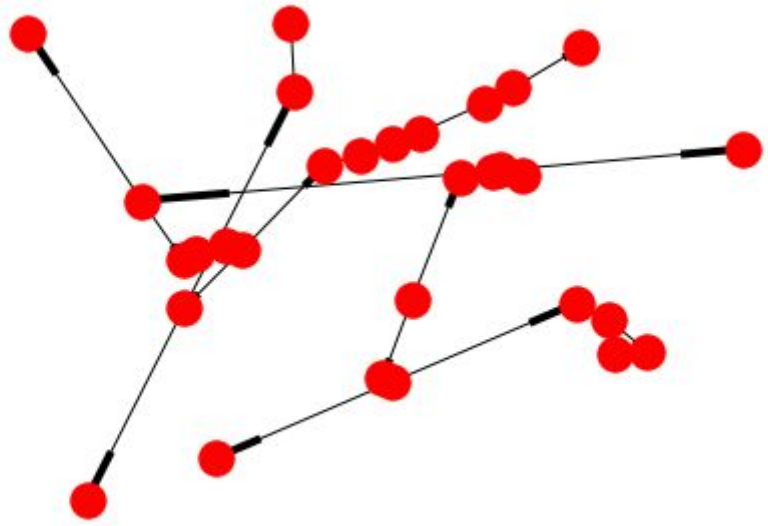


Imagem: MST gerada a partir de um grafo G.