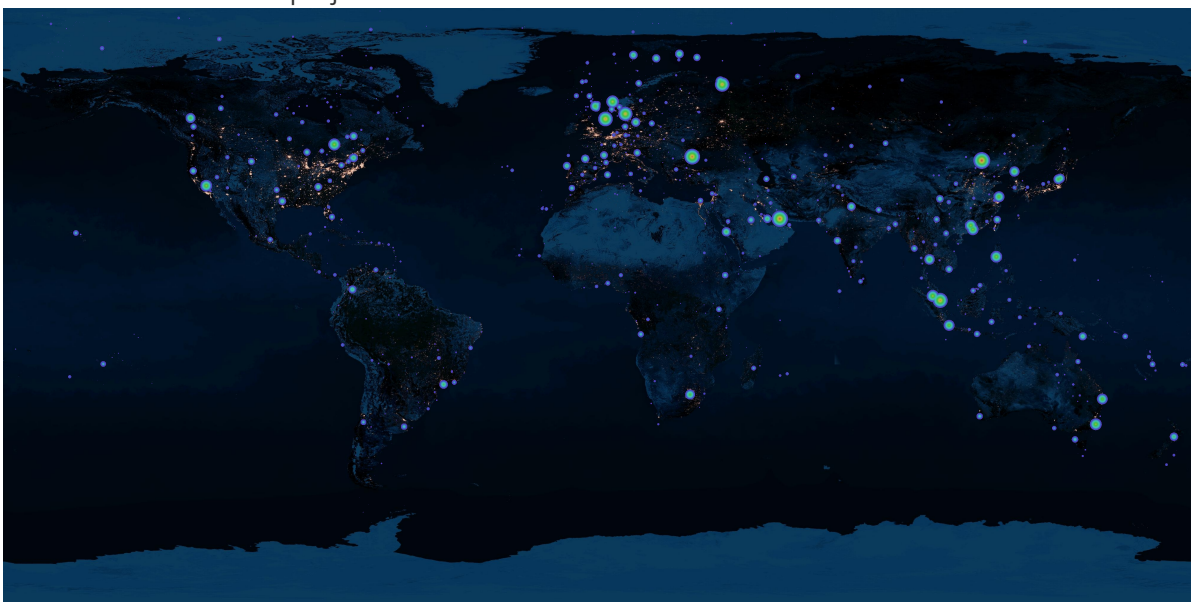
RESULTS

Outcome

Our team's project is called "**Voyager**." Voyager is a project which tries to find worldwide airports which have important effect on the transition. Heatmap is the visualization method we choose to show the results of our project.



Implementation

By using the algorithm **betweenness centrality**, our teammate finds the importance and centrality of every airport in the world. The unique ID for each airport is used as a key, and the value is all the neighbor airports around it. Centrality of all the airports is stored in the double array for "DrawGraph" function to use later.

First part is to find **betweenness centrality**, which is the number of shortest path of each pair of nodes that passes through each node. This task can be divided into two steps, first compute the length and number of shortest path between all pairs of nodes, and then sum all pair-dependencies. Using adjacency matrix to implement second step will take at least $O(n^3)$ time complexity, we choose to use Brande's algorithm to reduce run time to $O(m \cdot n)$. Brande's algorithm primarily replies on the following theorem.

1. A vertex $v \in V$ lies on a shortest path between vertices $s, t \in V$, if and only if $d_G(s,t) = d_G(s,v) + d_G(v,t)$.
2. Combinatorial shortest-path counting. $For s \neq v \in V, \sigma_{sv} = \sum_{u \in P_s(v)} \sigma_{su}$.
3. Accumulation of Pair-Dependencies. If there is exactly one shortest path from $s \in V$ to each $t \in V$, the dependency of s on any $v \in V$ obeys $\delta s \bullet (v) = \sum_{w:v \in P_s(w)} (1 + \delta s \bullet (w))$.
4. The dependency of $s \in V$ on any $v \in V$ obeys $\delta s \bullet (v) = \sum_{w:v \in P_s(w)} \frac{\sigma_{sv}}{\sigma_{sw}} \cdot (1 + \delta s \bullet (w))$.

Combining these theorem, we have Brande's algorithm implemented by

```
for each node v the graph:
    using queue to do BFS
    for each node q in queue:
        add to stack
        update minimum steps from v to each node and keep track of
predecessor
    using stack to do DFS
        update centrality by pair-depencency accumulation.

```

The second part is to present centrality onto world map as **heatmap**. This can be done in two steps. First, project the position of each airport on to the map based on the data we retrieve from dataset by converting the longitude and latitude of every airport into (X, Y) coordinate. Then we use the **BFS** to change the target pixels' hue. The range of hue is from 0 to 270. Redder (0) the pixel, then centrality of airport is higher. On the contrary, more purple (270) the pixel be, the less of centrality of this airport is.

We also write **test case** to ensure the accuracy of every function. We seprately test airport database size, airport read accuracy, adjacency matrix, centrality and coordinate conversion.

Discovery

Upon the heatmap we generate, we find that Asia and Europe have the reddest points, which means that airports have very important roles on the transition. In addition, comparing to the airports in the Asia, Europe, North America, South America, Africa, and Australia, almost all the important airports along the coastline.