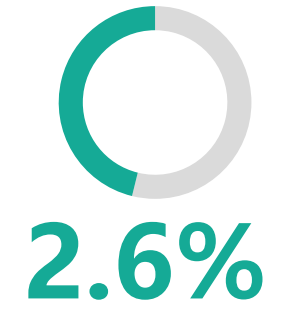
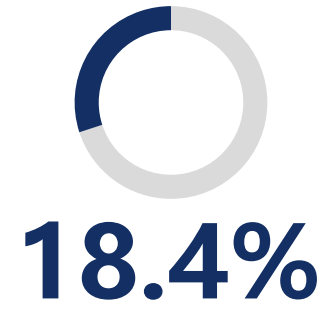
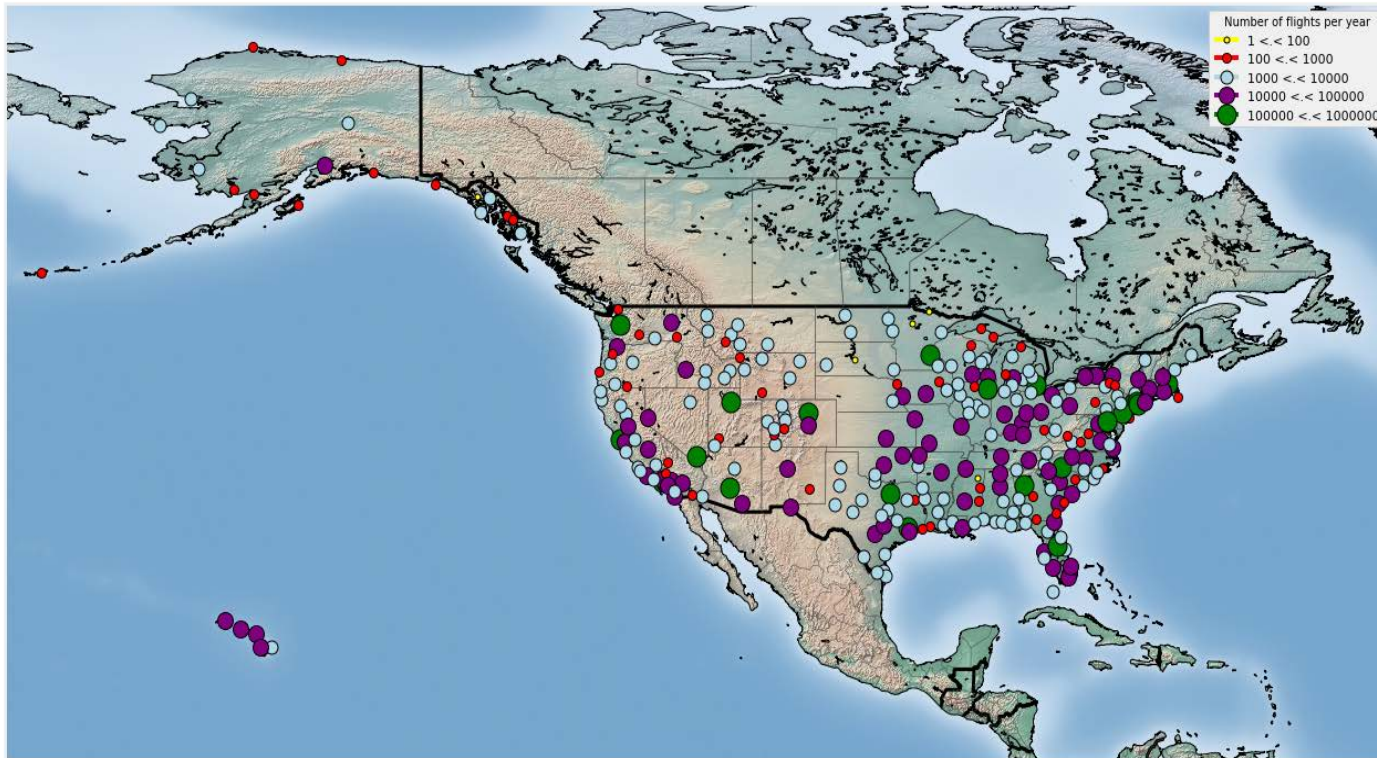


How to Plan Your Trip?

--an Analysis of Flights Data (1989-2008)

Group: Hide on Bush

Member: Yaling Xu; Siyuan Peng;
Zeyu Xu; Dongcheng Yang



Catalog

Introduction 01 | 02 Descriptive Analysis

Search Engine 03 | 04 Cross Point Heatmap

Introduction

PART



Data Description



Flight Delays and Cancellation Dataset from Bureau of Transportation Statistics



726,919,200 lines of records



Over 450 airports in U.S. domestic market from 1989 to 2008.

Main Variables

CRSDepTime/CRSArrTime:
scheduled departure/arrival time
(local, hhmm)



ArrDelay:
arrival delay, in minutes:

UniqueCarrier:
unique carrier code



Origin/Dest:
Origin/destination IATA airport code

FlightNum:
flight number



Date

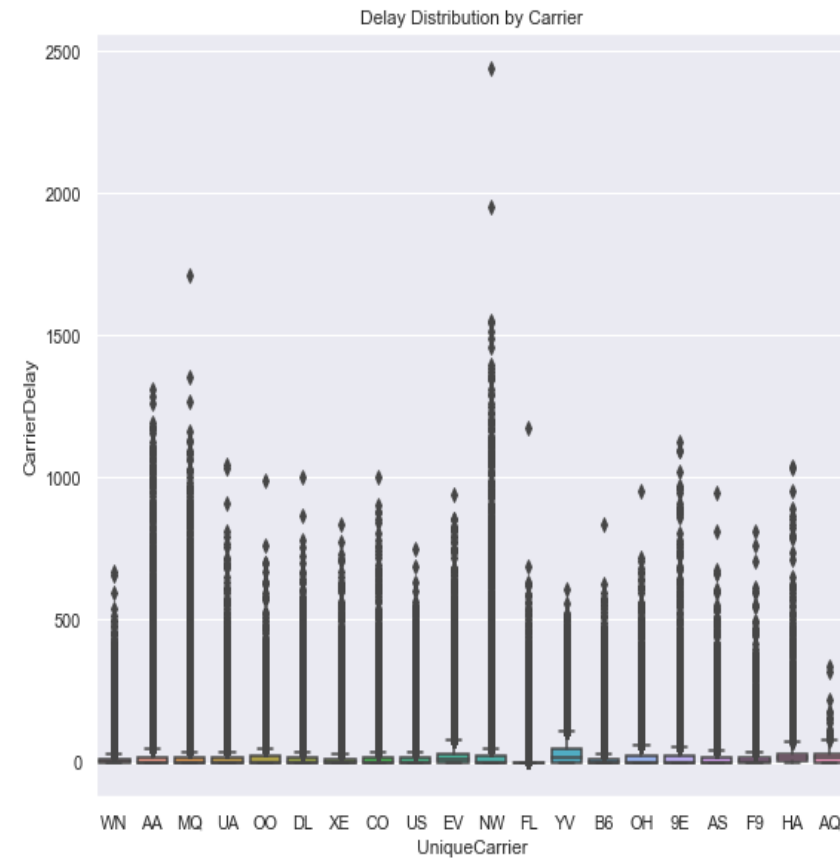
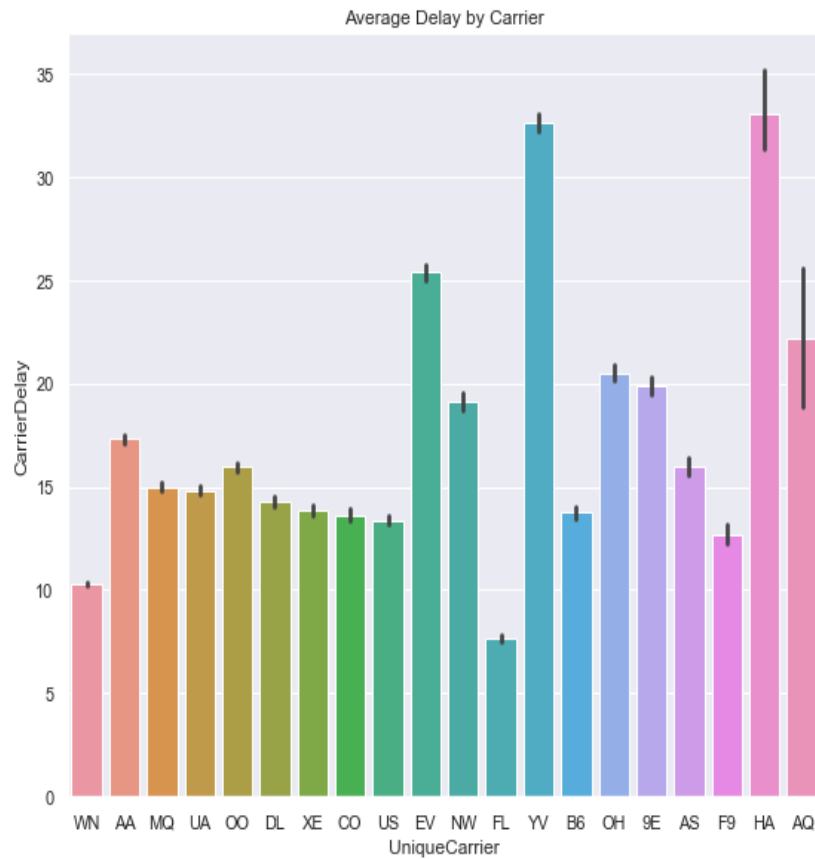
Descriptive Analysis

Delay Status

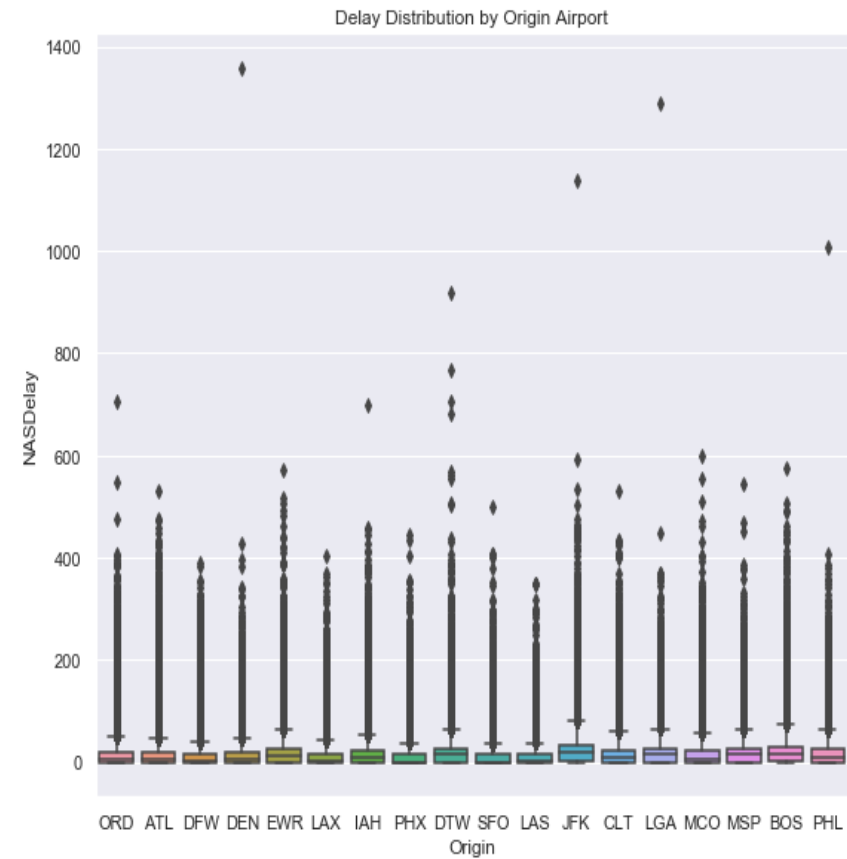
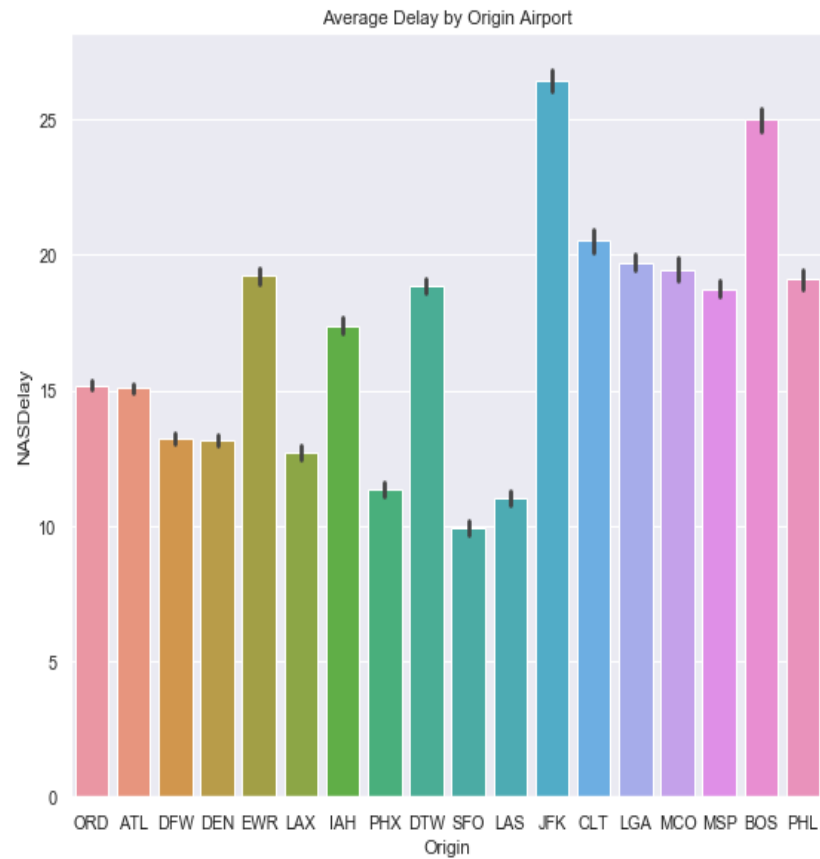
A woman with long brown hair, wearing a white long-sleeved shirt and dark pants, is sitting on a grassy hill. She is holding a smartphone up in her right hand, taking a photo of a mountain landscape. A red suitcase is next to her. The background shows a vast mountain range under a cloudy sky. The image is overlaid with a large white number '2' and the text 'PART 2' in large white letters.

PART 2

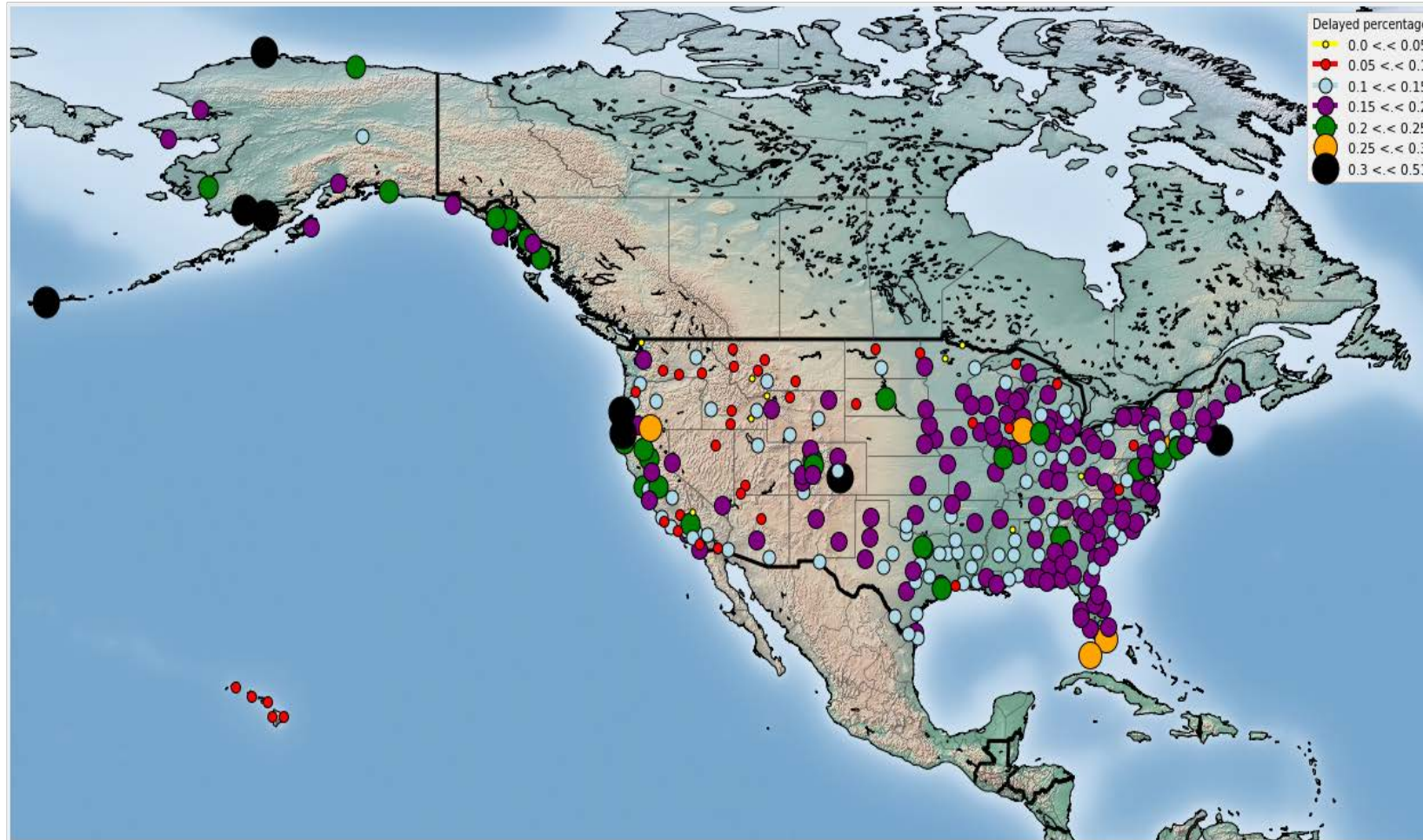
Delay by Carrier



Delay by Airport



Delay by Location



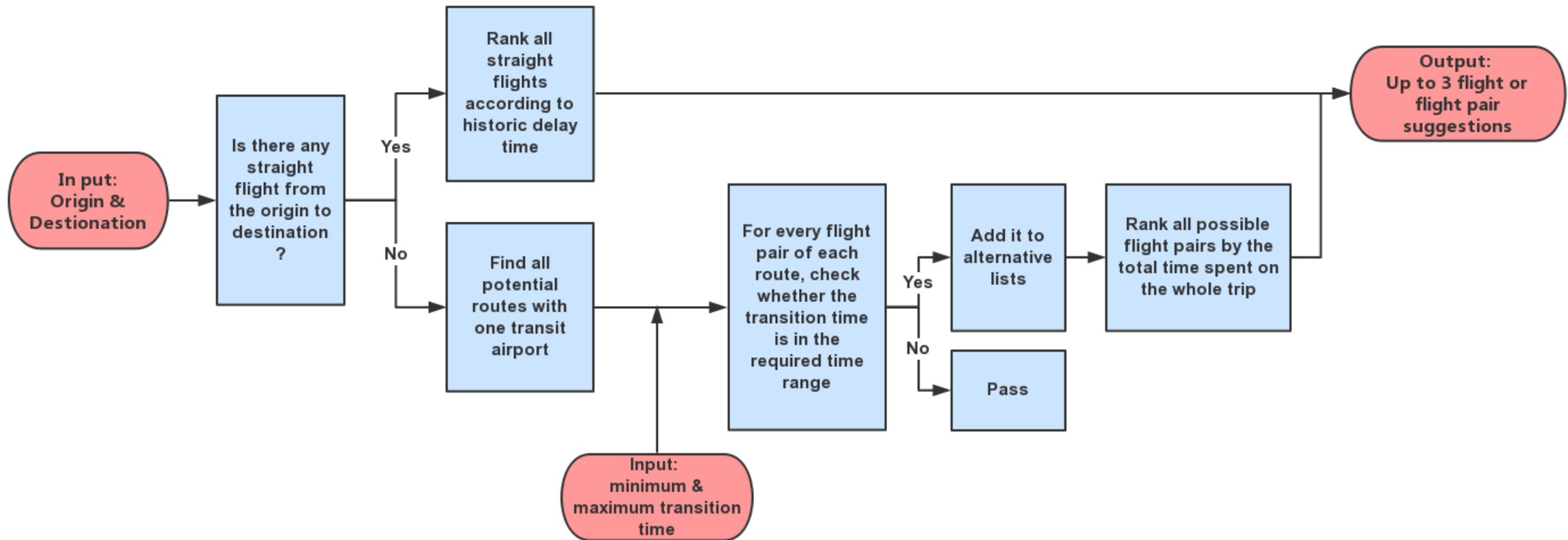
Search Engine

An App that
recommends
flights

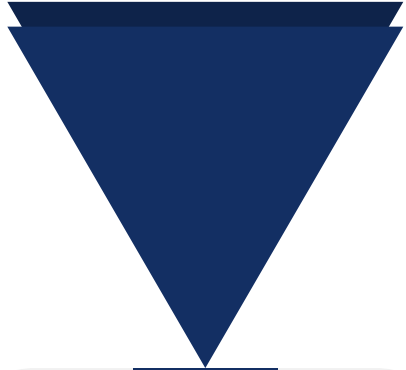


PART 3

Search Engine: Algorithm

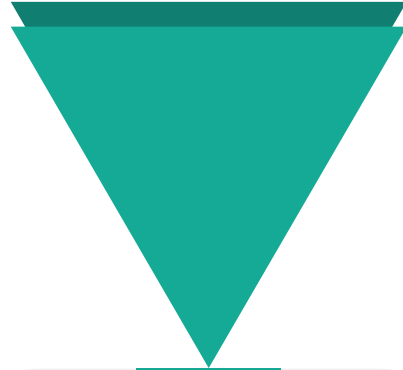


Search Engine: Information Collection



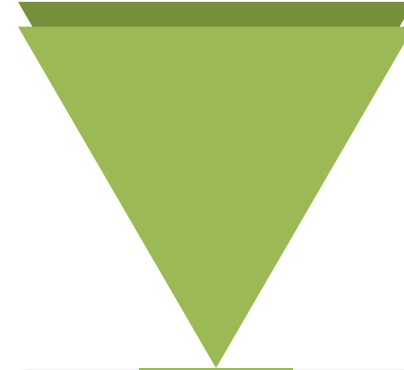
airportname.py

Use the airport IATA code as key, thus we can get those unique IATA codes. Later we will generate possible route with those airports.



timelist.py

Use a string including IATA codes of origin and destination as key, and a tuple consists of a string including carrier and flight number, departure time and arrival time as value. Thus we get information of all straight flights.



connecting.py

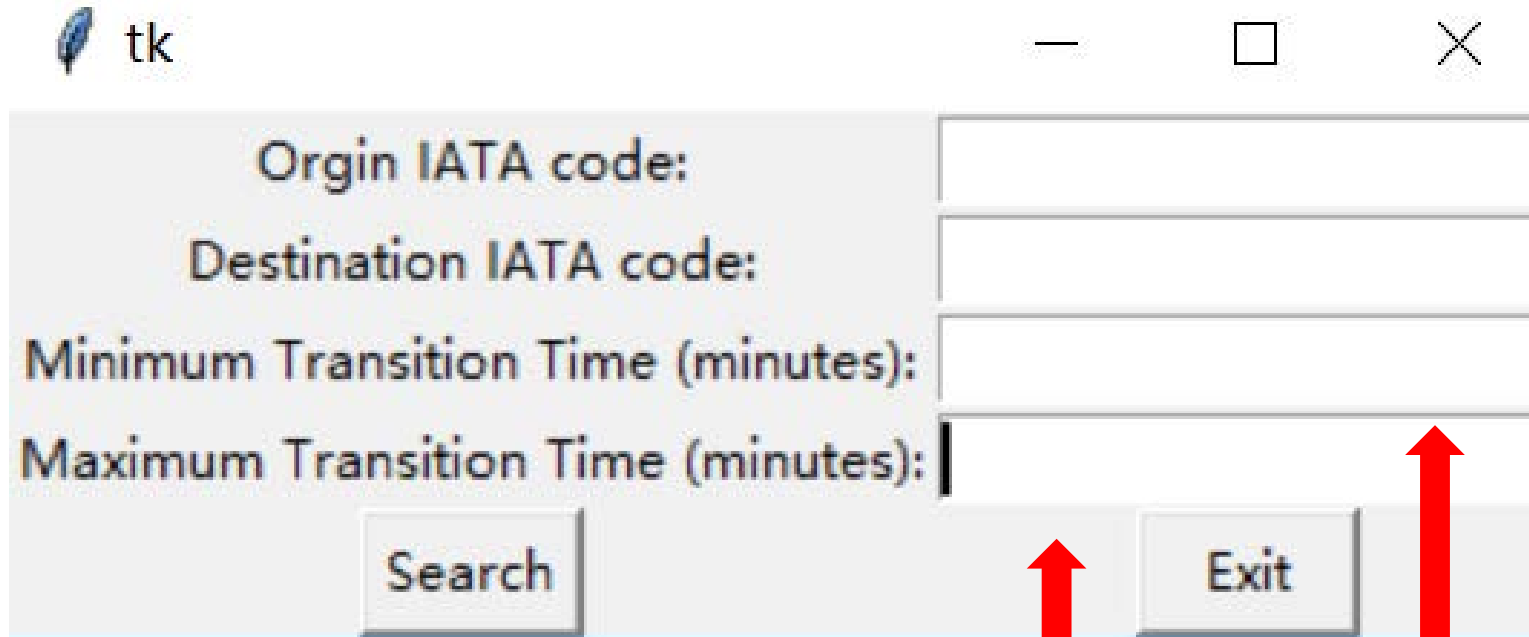
Use a string including information of every flight (IATA codes of origin and destination, carrier and flight number) as key, and a tuple consists of departure and arrival delay time as value. Then take average for every flight.



pair_generator.py

Generate two lists, including pairs of origin and destination with and without straight flight between them respectively. Generate a dictionary for pairs without straight flight between them and find possible transition airports for them.

Search Engine: UI interface

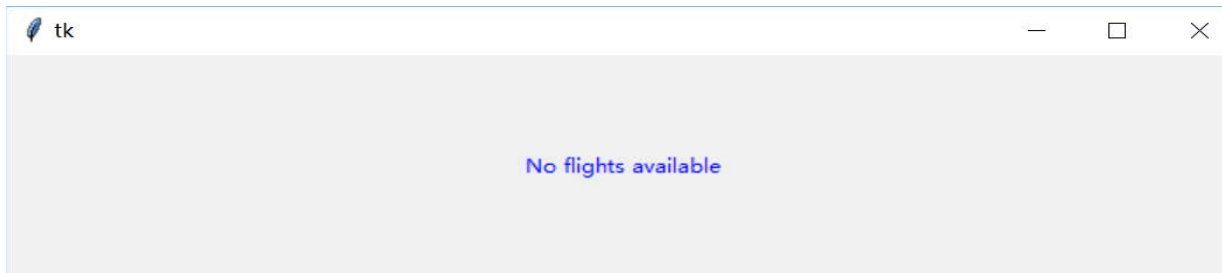
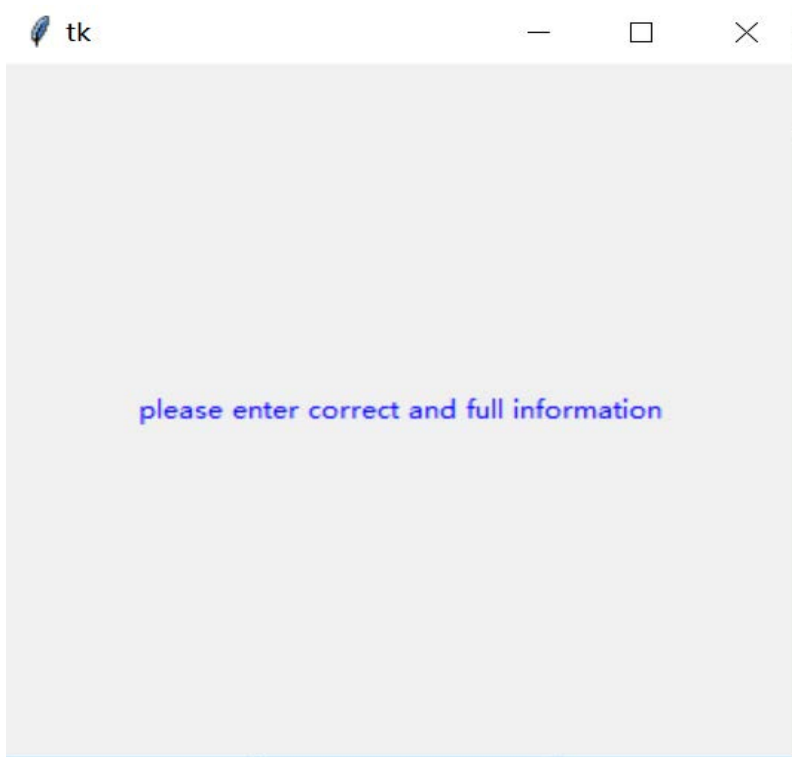


A screenshot of a Tkinter window titled 'tk'. The window contains four text input fields and two buttons. The labels for the fields are: 'Orgin IATA code:', 'Destination IATA code:', 'Minimum Transition Time (minutes):', and 'Maximum Transition Time (minutes):'. The 'Search' button is located below the first two fields, and the 'Exit' button is below the last two fields. Red arrows point from the input fields to the explanatory text boxes on the right.

Enter IATA code (3 letters, both capital and small letter are acceptable)

Enter integar, then the time range is determined

Search Engine: Results – Error Warning



Two cases:

- The user enters wrong information or leaves some box blank
- The user enters correct and full information but our database doesn't have related data to be analyzed

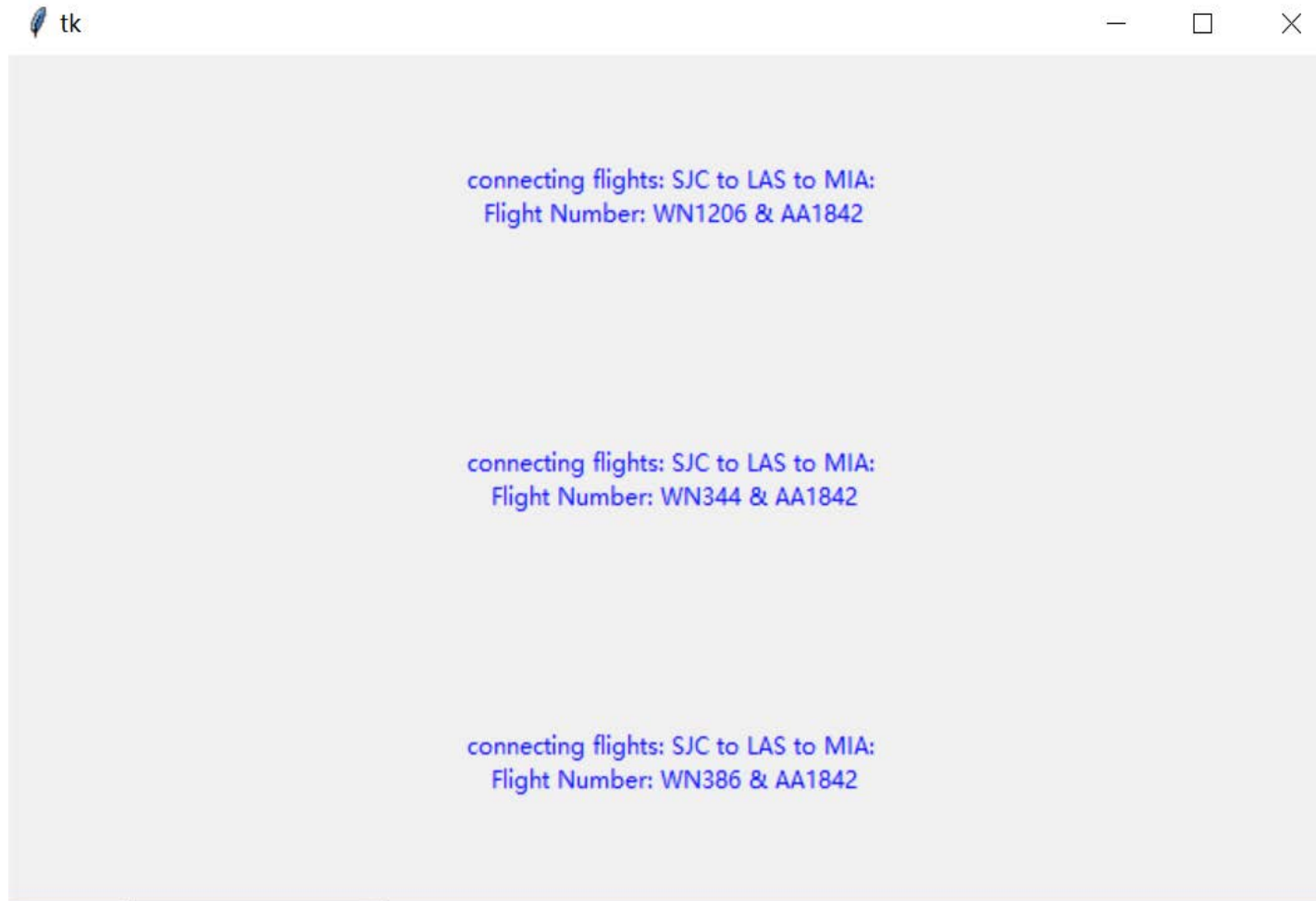
Search Engine: Results – Straight Flight



Information provided:

- Specific Flight Number (Short Name of Carrier and Flight Number)
- Planned Departure Time (24 hours)
- Planned Arrival Time (24 hours)

Search Engine: Results – Connecting Flights



Information provided:

- The whole route (IATA codes of Origin, Transition Airport, Destination)
- Specific Flight Number (Short Name of Carrier and Flight Number)

Weather Inference

Heatmap of
Flight Delay



PART 4

Weather Inference: Algorithm

Step 1: Pair all delayed flights in one day and calculate the crisscross points

Assumption: Crisscross points could be calculated based on plane coordinate system

Step 2: Round the latitude-longitude coordinates up to the nearest integers

Step 3: Divide the values of each grid by number of years

Step 4: Plot average number of delayed flights in the grid map

Weather Inference: Method

Dask

- primary parallel processing library in python

Dask.delayed() decorator

- stages operations that are to be parallelized
- any function touched becomes lazy and run later

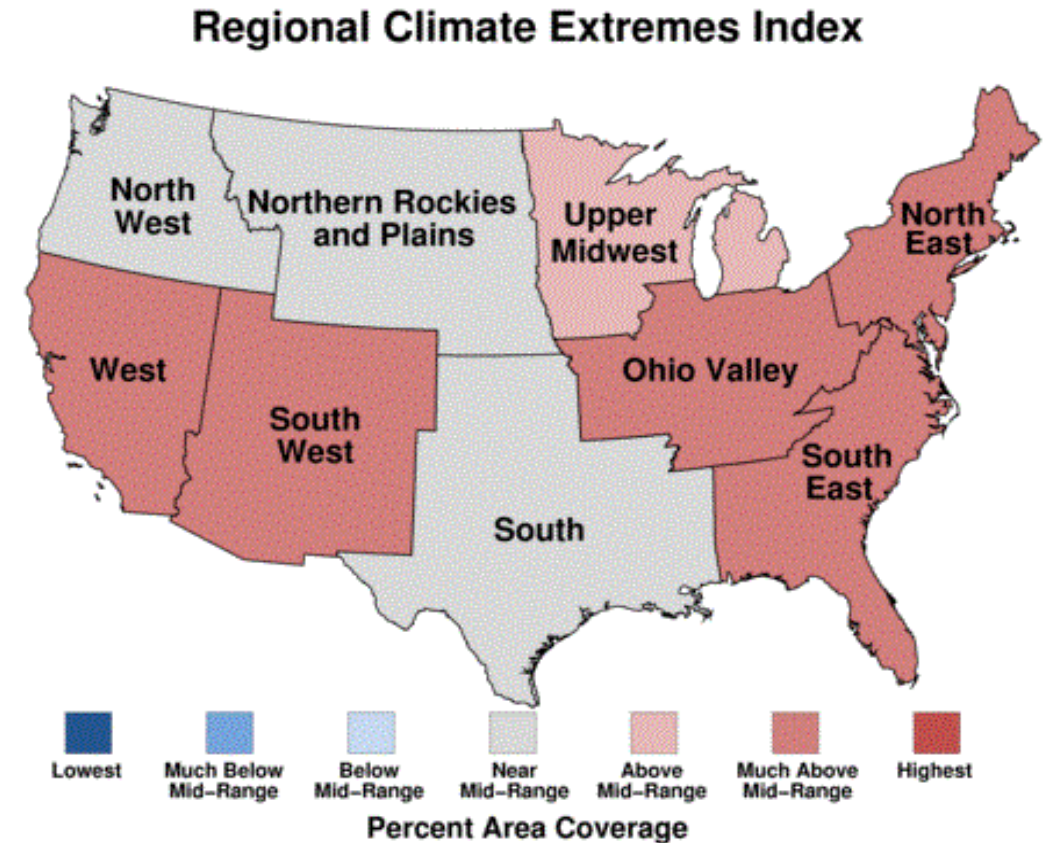
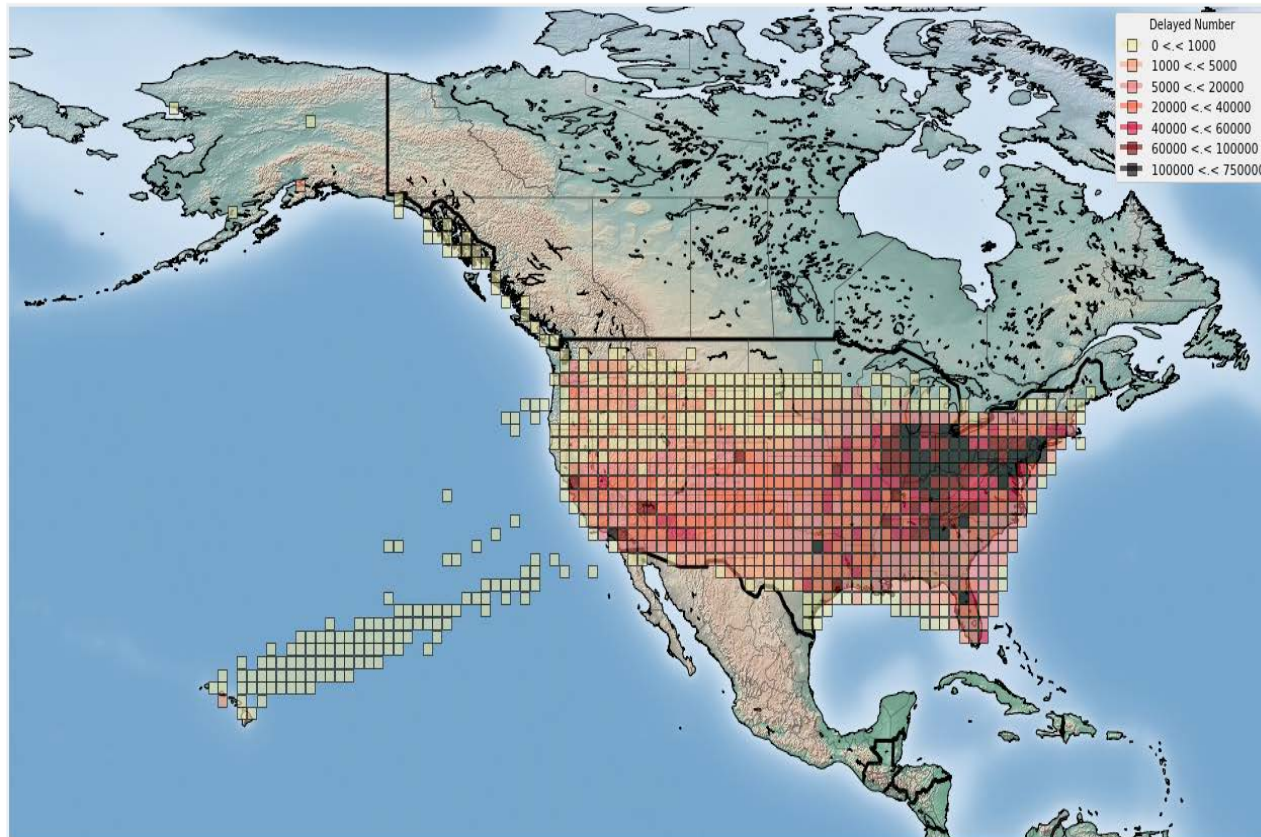
Dask.compute()

- runs delayed objects parallel

Time Reduction

- from 20 hours to 6 hours

Weather Inference: Results



Data Source: National Climate Data Center

THANK YOU

The image features a horizontal banner. The left portion of the banner is a solid teal color, while the right portion is filled with a complex geometric pattern of overlapping triangles in various shades of yellow, orange, and light green. The text 'THANK YOU' is written in a bold, white, sans-serif font across the teal section.