ADVANCED AERIAL REFUELING

OPTIMIZATION FOR THE MILE HIGH FUEL CLUB

Evan Marrone

Lincoln Laboratory Team: Allison Chang, Brian Bassham, Jesse Pietz
MIT Advisor: Jordan Levine

Problem Statement

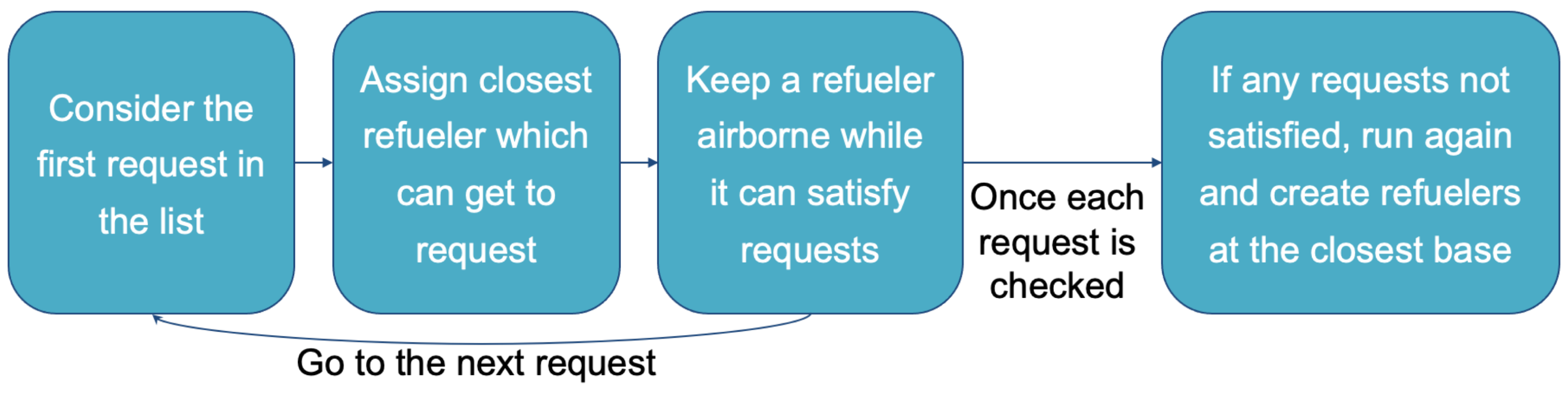
Motivation

Aerial Refueling: Air Force missions require mid-air refueling due to limited fuel capacity in our aircraft

Room for Improvement: Air Force's current refueler scheduler leads to suboptimal scheduling—could leverage better approach

Expensive: Air Force spends hundreds of millions of dollars on each individual refueler and billions of dollars on fuel every year

Baseline Scheduler



Consider the first request in the list → Assign closest refueler which can get to request → Keep a refueler airborne while it can satisfy requests → Once each request is checked → If any requests not satisfied, run again and create refuelers at the closest base

Go to the next request

Objective

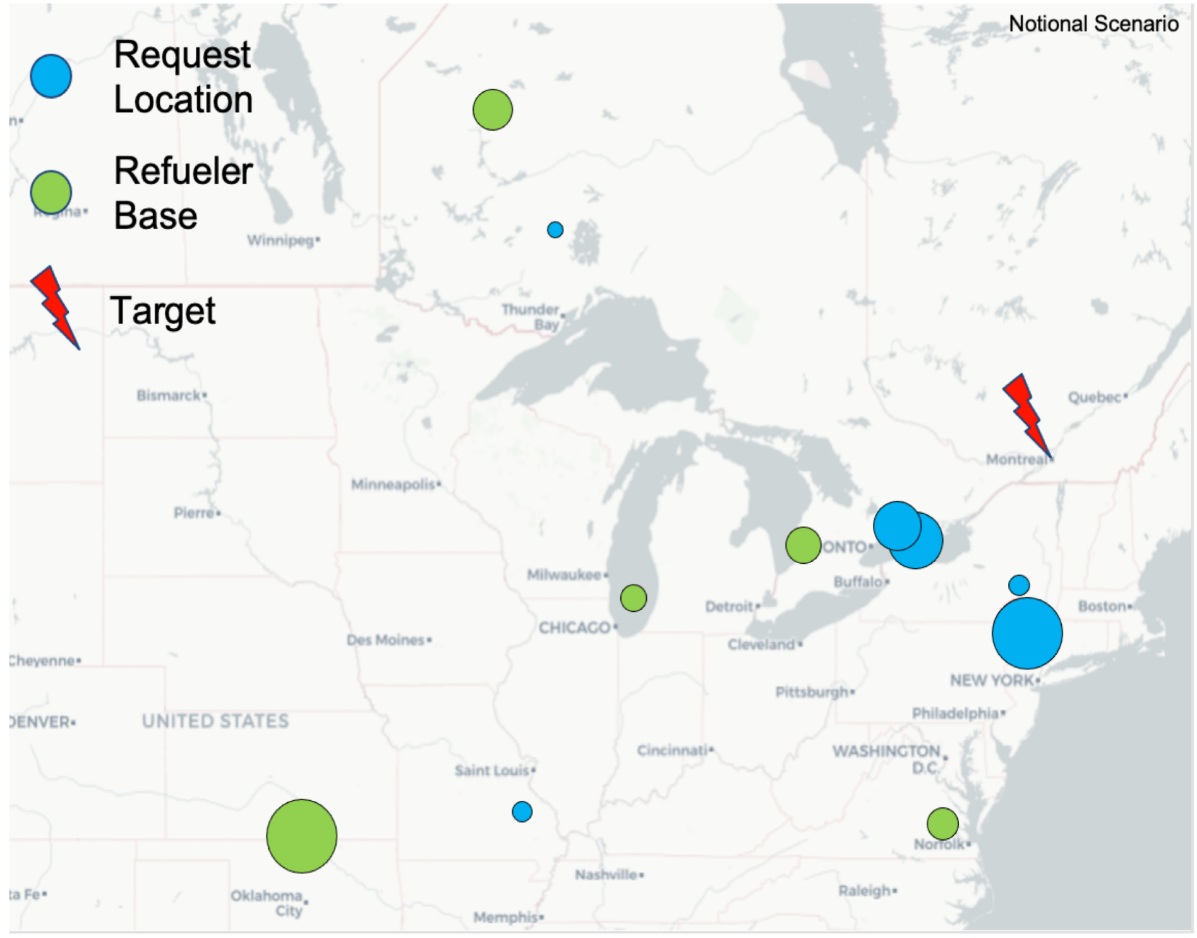
Redesign the current refueler scheduler, improving the initial 'naïve' algorithm to a more holistic approach, **quickly minimizing the number of refuelers and fuel burned while meeting all requests**

Scope

The **target** is where aircraft (e.g., fighters, bombers) are focusing their efforts.

They request refueling at one of several predefined **request locations.**

The scheduler assigns refuelers from the various **refueler bases** to the requests.

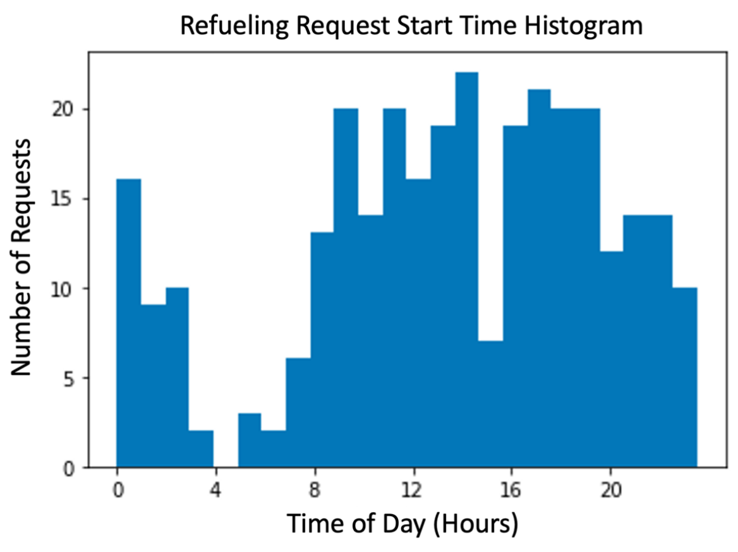


Analysis

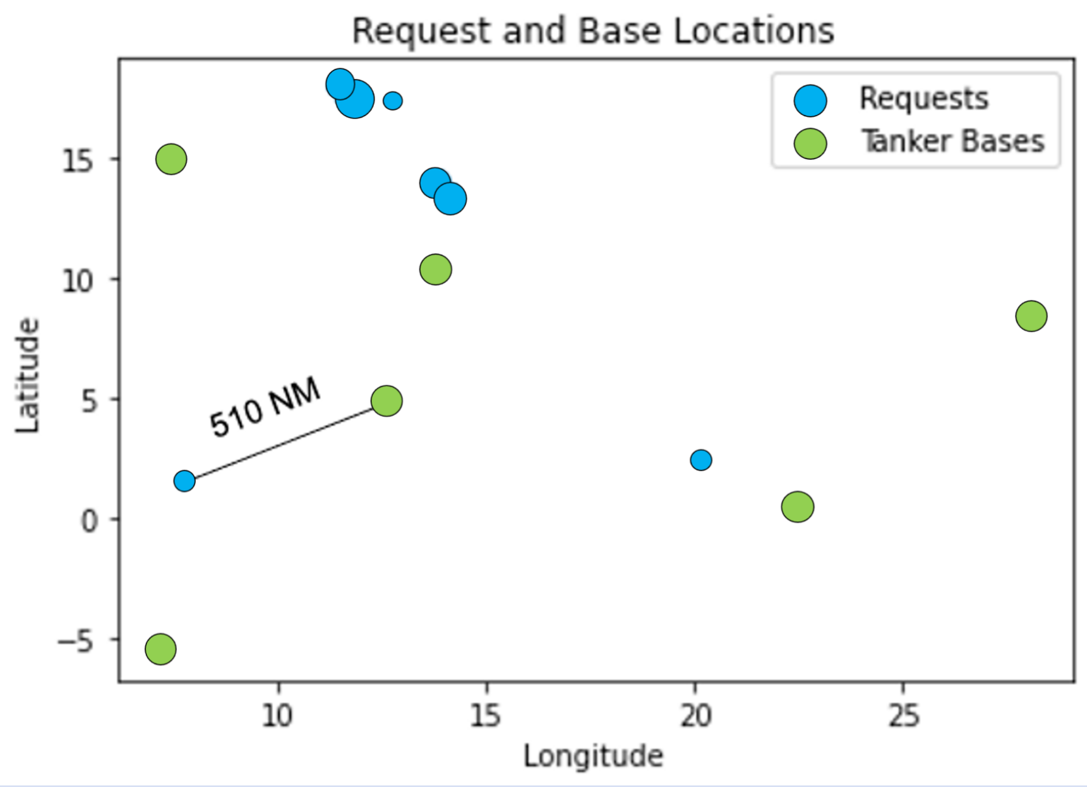
Thorough exploratory data analysis (EDA) and a variety of different approaches eventually led to an **innovative and successful algorithm**

EDA

When looking at the request start times of a multi-day scenario, we see requests happen at all times of the day with a small lull between 0300 and 0800 hours.



Looking at the request sizes (based on number of requests at a location) and distances to bases we notice there is an uneven distribution of requests as well as some well-positioned bases.



Approaches

Developed and implemented several approaches to improve upon the baseline… with little success

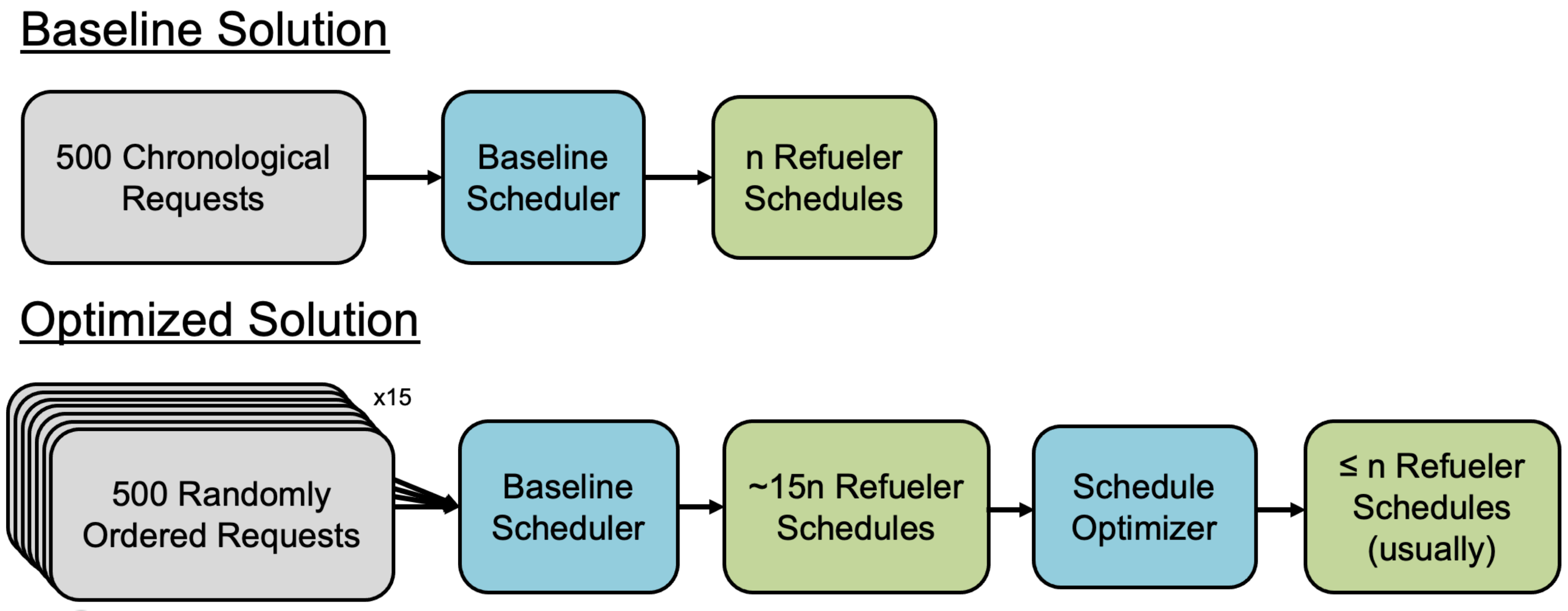
Refueler-Centric
Let refuelers maximize their requests

Trip Comparison
Compare each base's refueler's trips

Combined Trip Savings
Combine trips that maximize savings

Geospatial Clustering
Vary request order by location

These failures led to a final approach: **Run the baseline scheduler multiple times with randomized requests orderings, then merge the resulting schedules to minimize refuelers and fuel burned**



Baseline Solution

500 Chronological Requests → Baseline Scheduler → n Refueler Schedules

Optimized Solution

500 Randomly Ordered Requests (x15) → Baseline Scheduler → ~15n Refueler Schedules → Schedule Optimizer → ≤ n Refueler Schedules (usually)

The Schedule Optimizer first filters through the best resulting scenario schedules (based on number of refuelers) and, of those, employs a mixed integer program which treats individual refueler schedules as decision variables, and then minimizes fuel subject to every request being met

Results and Impact

Results

Tested across 200 unique scenarios, the optimized scheduler results in…

5.0%
Fewer Refuelers

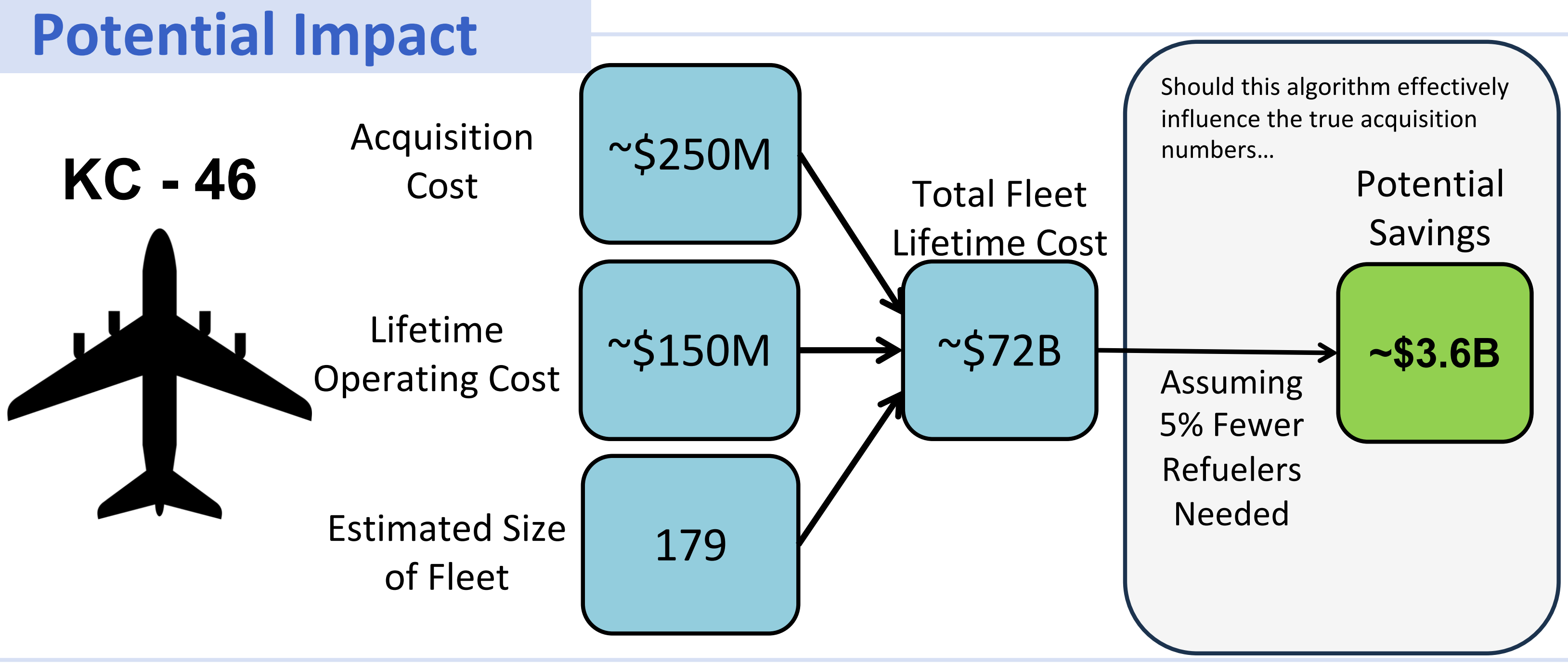
0.7%
Less Fuel Burned

97.5%
of cases had ≤ baseline number of refuelers

$O(n^2)$
computational complexity (excluding optimizer)

All done with a slight, yet innovative, modification to the baseline scheduler (i.e. easily implementable for host company)

Potential Impact


KC - 46
Acquisition Cost
~$250M
Lifetime Operating Cost
~$150M
Estimated Size of Fleet
179
Total Fleet Lifetime Cost
~$72B
Should this algorithm effectively influence the true acquisition numbers…
Potential Savings
Assuming 5% Fewer Refuelers Needed
~$3.6B

Future Work

Find programmatic way to order requests to produce an optimal schedule

Add heuristics to the proposed scheduler to more quickly minimize both fuel and number of refuelers