Project description

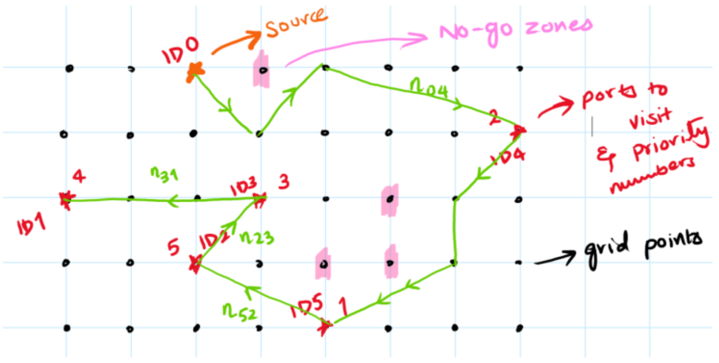
Team name: Group 4

Problem What is the problem you would like to solve?

Ship route optimization is an important aspect of maritime trade planning. Before voyaging, freight shipping companies utilize complex algorithms to plan shipping routes between ports based on several factors including available routes, weather conditions, congestion, trade levies, fuel costs, and geopolitical and safety considerations (shallow waters, hostile waters etc.). Additionally, the ship condition, size, weight, fuel capacity, and emissions also play a role.

This is a highly complex problem, and we will only focus on a subset of this problem: **given a port of origin and multiple ports that a freight ship must call on, how do we optimize the sequence of ports that a ship visits and the route that it adopts. We can also explore how different objectives and constraints can affect the outcome of the optimization problem.**

Optimization model



1. What are the decision variables?

This would be a directed graph problem, with the edge flow being Boolean decision variables. If the ship passes between two grid points (x, y coordinates) then the edge flow has a value of 1, otherwise 0.

2. What are the constraints?

There could be a combination of constraints, for example:

- Each port can only be visited once
- The ship can only travel a certain maximum distance between a call
- Different routes may have different max speeds depending on factors such as congestion, weather conditions etc. These could be edge costs
- Some routes are to be avoided because of hostile waters, weather conditions etc.

3. What are the parameters?

Parameters could be:

- Edge costs such as the speed a ship can have for a given edge or the time it takes for the ship to traverse that edge
- Maximum distance a ship can travel before it must stop at a port: this could be a fuel capacity constraint
- The constraint on each node (grid point) could be a parameter, e.g. some nodes are not allowed to be in the ship's planned route
- Different ports could be allocated different priority numbers and the cost/objective could change based on these parameter values

Analysis What kind of analysis do you want to do? (Sensitivity to constraints/parameters, difference between solving a nonlinear vs linear model, etc.)

Given the complexity and the number of factors involved in ship route optimization, we choose to perform sensitivity analysis and scenario analysis majorly.

1. Sensitivity Analysis:
    a. Sensitivity to Edge Costs (Travel Time/Speed) - Vary the edge cost parameters and observe the changes in the optimal port sequence and route.
    b. Sensitivity to Maximum Travel Distance (Fuel Capacity) - Change the maximum allowed distance and see how it restricts the possible routes and the optimal port order.
    c. Sensitivity to Port Priorities – Adjust the priority parameters assigned to different ports and observe the resulting changes in the optimal solution.
2. Scenario Analysis:
    a. Different Weather Conditions – Define discrete weather scenarios and adjust the corresponding edge costs or constraints for each scenario, then solve the optimization problem for each.
    b. Congestion Levels – Create scenarios with varying congestion levels, reflected in increased edge costs for affected routes.

Data requirements Where can you find the data you need?

For this project, we will generate synthetic data to simulate the maritime environment. Our model will operate on a 2D uniform grid where each point represents a navigable waypoint. The port of origin, required destination ports, and no-go zones (e.g., hostile or shallow waters) will be randomly generated using user-defined parameters.

We will model:

- **Nodes** as grid coordinates (e.g., (x, y))
- **Edges** as possible travel paths between neighboring nodes
- **Edge Costs** as attributes like travel time or fuel usage, sampled from predefined distributions (e.g., uniform or normal distributions) to simulate real-world variability such as sea state, weather, and congestion.

Python will be used for data generation due to its simplicity and powerful libraries. A typical generation process would involve:

- **Grid Creation**: Generate a meshgrid using numpy.meshgrid.
- **Port Placement**: Randomly select grid points as ports using numpy.random.choice.
- **No-go Zones**: Use a probability threshold to flag some grid points as restricted.
- **Edge Costs**: Assign edge weights using numpy.random.normal to reflect real-world travel time or risk.

Attributes of each edge can include Base travel time, Fuel cost, Weather delay factor and Congestion multiplier.

Later, we plan to introduce **jet streams or ocean currents,** modeled as vector fields over the grid that accelerate travel in specific directions. These will be incorporated by modifying edge costs dynamically: if the direction of travel aligns with a jet stream, the effective travel cost will be reduced (e.g., cost = base_cost * (1 - boost_factor)).

This approach provides control, reproducibility, and flexibility, allowing us to explore how various constraints and parameters affect optimal routing. As the model matures, we may replace synthetic data with real-world datasets.

Next steps What are next steps (e.g., formulate optimization problem, gather data, start implementing the model)? Who is responsible for doing the next steps?

Make a list of some next steps and how and by when you want to achieve them.

Team Responsibilities

- Dataset creation and grid-based scenario generation
- Exploring different objective functions and constraints in Julia

- Implementing the base optimization model and testing solvers in Julia

Timeline Overview

1) April 9–10: Prepare for and present our project pitch to another group. Gather feedback and make adjustments to the problem scope if necessary

2) April 11–17:
    - Generate the 2D grid, ports, no-go zones, and edge cost data (using Python).
    - Begin setting up the optimization model in Julia.
    - Define candidate objective functions and possible constraint sets.

    Hold a team meeting to align on structure and integration.

3) April 18–24:
    - Integrate the dataset into the Julia model.
    - Implement multiple constraints and test different objective functions
    - Try different solvers and track performance

    In-class group meeting on April 24 to show progress and get feedback

4) April 25–30:
    - Conduct sensitivity and scenario analyses.
    - Visualize routing outcomes under different conditions
    - Begin drafting the final report and prepare any figures

5) May 1–2: Final integration of all components. Review and polish code, documentation, and report Submit the complete project by the deadline

We will meet as a team once a week to discuss progress, debug issues, and integrate our work.