Maddy Lok

MSDS 460

2 February 2025

Network Models – Project Management

Methods

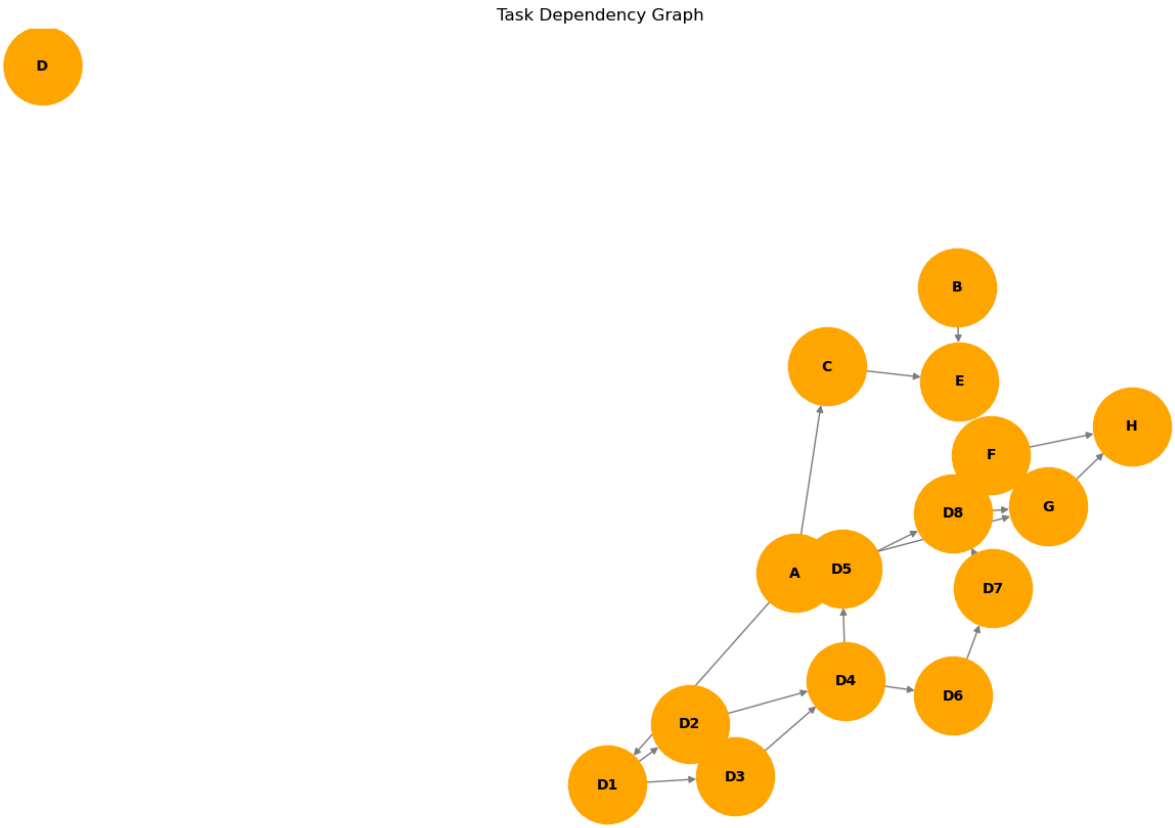
I utilized Linear Programming to create an optimal schedule for the development of a consumer-focused recommendation system of restaurants in Marlborough, Massachusetts. Utilizing critical path analysis, I was able to determine the minimum project duration. No resource constraints were implemented, as the critical path analysis assumes an infinite number of available resources. An LP problem was created to identify the fastest completion time of the project in three different scenarios. I used Python and PuLP to solve this problem in the best-case, worst-case, and expected-case scenarios in order to provide a recommendation of estimated budgets and workforce counts to the clients. Utilizing Gantt charts to visualize task schedules, I was able to analyze how workforce expansion could impact the delivery time.

Part 1: Problem Setup

The project plan was developed by outlining the sixteen tasks in the development of this system. The tasks were arranged based on their predecessors and dependencies. I assigned standard industry time estimates for each task, and estimated the average worst-case, expected-case, and best-case times. A directed graph diagram was then created to see which tasks are dependent on the completion of other tasks. I then was able to identify tasks that can be completed in parallel by categorizing tasks into Independent tasks and Single Dependency Tasks. Independent Tasks are tasks that are capable of beginning immediately and do not require a prior task to be completed. These included tasks A (Describe product), B (Develop marketing strategy), and D (Develop product prototype). I also estimated the average hourly salary of a contractor in each specialization per the average in the state of Massachusetts. *See Excel sheet in Github repository.*

As we are utilizing critical path analysis, this linear problem will not be constrained by limited resources. There are several factors that contribute to areas of uncertainty in this project. For example, the availability of a sufficient amount of data can be one constraint. As the Yelp API is going to be utilized as the dataset for this project, there may be limitations of the API. This also creates the risk of dependency, as we are relying on a third-party service for certain tasks which may cause delays. There are also cost variations that come with the utilization of contract workers. If the project is delayed, this can increase the labor costs that may not have been anticipated into the budget.

Directed Graph Diagram - Task Dependency Graph



Part 2: Model Specification

The linear programming model was formulated with the objective of minimizing the duration of the project development. The decision variables were appointed as the start and end

times of each task. The constraints were defined by the dependencies of each task as well as the total project duration.

Objective function:

$$minT_{end}$$

Decision variables:

T_i = start time of task

D_i = duration of task

Constraints:

$$T_j \geq T_i + D_i \rightarrow \text{dependency constraint}$$

$$T_{end} \geq T_i + D_i \rightarrow \text{total project time constraint}$$

Part 3: Programming

The optimal start times of each task was calculated in Python:

Task	Start Time	Task	Start Time
A	0 hours	D5	182 hours
B	0 hours	D6	164 hours
C	24 hours	D7	196 hours
D	0 hours	D8	220 hours
D1	32 hours	E	48 hours
D2	82 hours	F	252 hours
D3	82 hours	G	232 hours
D4	132 hours	H	276 hours

The critical path was identified in order to identify all the tasks that contributed to T_{end}.

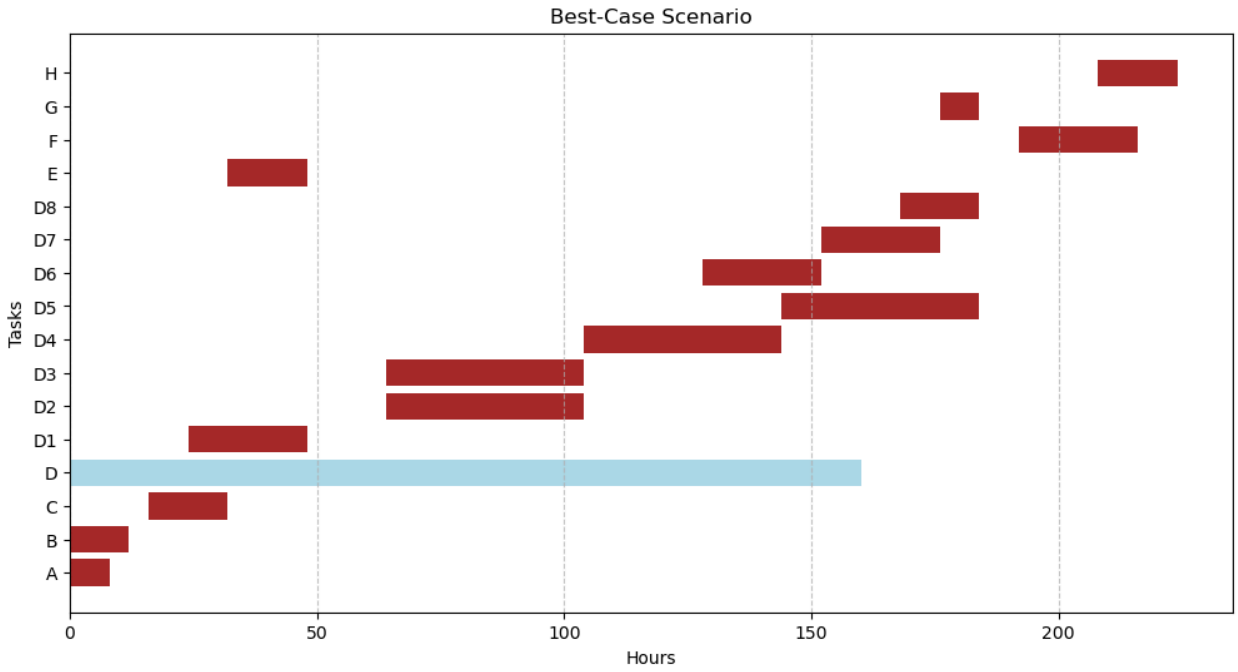
$$B \rightarrow A \rightarrow C \rightarrow D1 \rightarrow E \rightarrow D2 \rightarrow D3 \rightarrow D4 \rightarrow D6 \rightarrow D5 \rightarrow D7 \rightarrow D8 \rightarrow G \rightarrow F \rightarrow H$$

To find the critical path, I was running into some bugs in my code, so I utilized ChatGPT to help with debugging. I have noted these blocks of codes that I utilized from ChatGPT in the Python file.

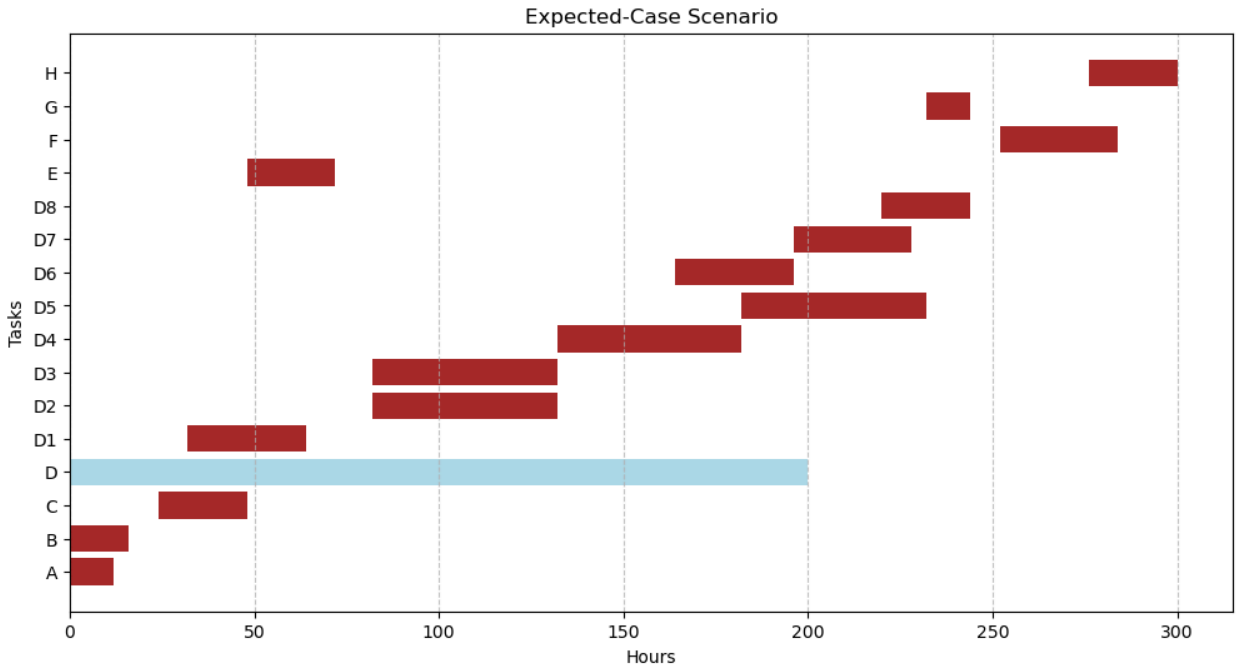
Part 4: Solution

The model was implemented in Python utilizing the PuLP package in order to solve the minimized project development time for the best-case, expected-case, and worst-case scenarios.

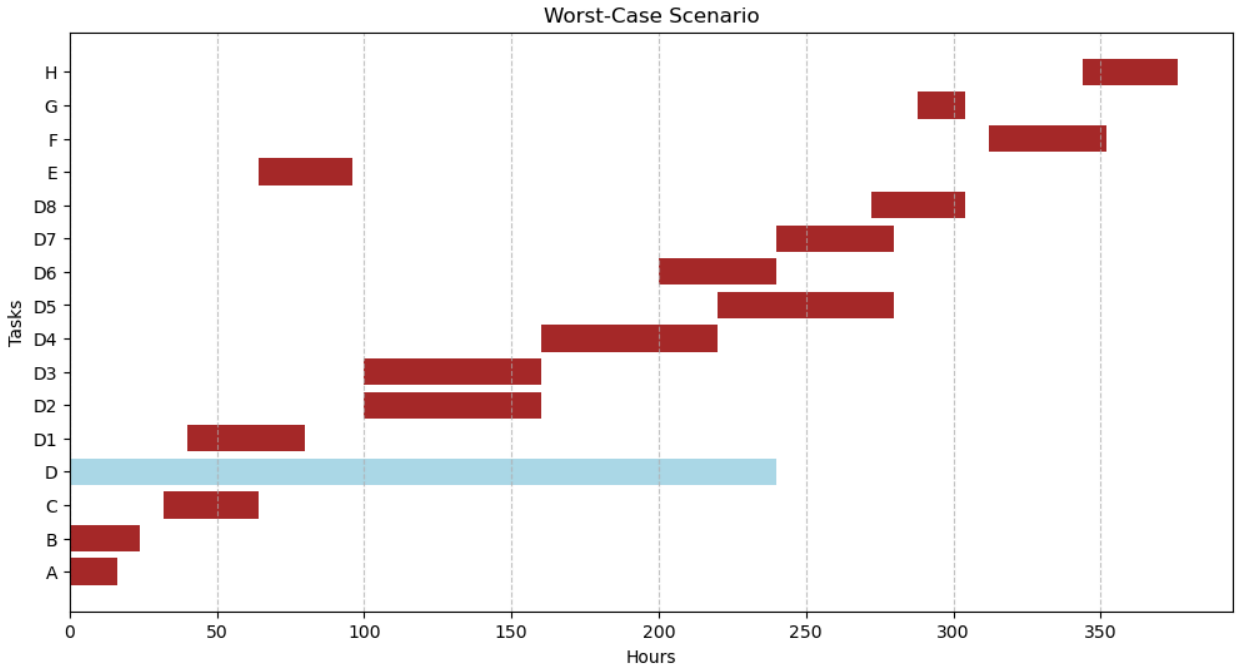
- Best case scenario: 224 hours



- Expected case scenario: 300 hours



- Worst case scenario: 376 hours



The critical path remained the same across all three scenarios, which implies that increases in the workforce affects tasks that are completed in parallel rather than on the sequential dependencies.

Gantt charts were created for each scenario to display tasks that were able to be completed in parallel, task sequencing, and overall timeline differences.

Part 5: Overview

For this project regarding the development of a consumer-focused recommendation system for restaurants of Marlborough, Massachusetts, we have developed a plan to ensure the most optimal outcome. We have come up with several different scenarios to estimate the projected delivery timeline of this project.

- Best case scenario: 224 hours/5.5 weeks
- Expected scenario: 300 hours/7.5 weeks
- Worst case scenario: 376 hours/9.4 weeks

The expected scenario is the most likely estimate, in which we anticipate a fully functional recommendation system within about 7 and a half weeks from the launch of the project. This timeline considers all steps of development, from software development to model training and deployment.

We also conducted a product cost estimate, in which we calculated the total labor costs of each scenario. Please note that this does not include server or software licensing costs.

- Best case scenario: $99,000
- Expected scenario: $129,990
- Worst case scenario: $161,220

The hourly rates for each professional were calculated based on the average contractor rate for each specialized worker in the state of Massachusetts. The total cost per scenario was calculated utilizing these rates along with the estimated work hours for each task. It is recommended to budget for the worst case scenario to cover all potential scenarios.

If a faster delivery timeline is necessary, the workforce can be increased in order to satisfy this requirement. This leads to higher labor costs, but will drastically reduce the duration of the

project. With an increase of 50% of the available workforce, we can expect the duration of the project to be reduced by about 16-18%. These percentages account for diminishing returns and dependencies. Below are the reduced estimates with a 50% increased workforce.

- Best case scenario: 184 hours/4.6 weeks
- Expected scenario: 246 hours/6.2 weeks
- Worst case scenario: 316 hours/ 8 weeks

To mitigate risks, we can employ certain methods to ensure the success of the project. Stochastic programming can be utilized to account for variability in task durations. There are several developmental delays or API issues that can occur during development which need to be accounted for. Additionally, the Monte Carlo simulation can be utilized, as these methods are able to simulate multiple project timelines under various risk conditions. This approach can help decision-makers account for any delays that may occur. Agile development can also be implemented in order to ensure that a working prototype is always available even if unexpected delays occur, as this method utilizes bi-weekly sprints and incorporates incremental development.

Conclusion

Based on our LP model, the development time of the recommendation system should only take around 7.5 weeks as long as no unexpected delays occur. Increasing the workforce resources by 50% can decrease this duration to around 6.2 weeks, while increasing the minimized total labor costs of the project. With the standard recommended workforce count, the total labor costs of the project is estimated around $129,990, but the clients should ensure to budget for unexpected delays which would increase labor costs. To mitigate the risks of bottlenecks in the development process, the critical path should be followed.