Report

Here are some photos of me explaining my thought process to chatgpt:

Task 2: Algorithm Design (Coding exercise)
Problem:
Imagine the FMS has sent the paths to robots without considering any scheduling constraints. The
robots are moving along the path towards the conflicting region. Paths of robots are expressed as
an array of node ids that it has to go through (Nodes are blue dots in Fig. 1).
1. Write a pseudocode algorithm that:
• Detects conflicts in the paths of robots dynamically as they approach the conflicting
regions.
• Resolves the conflict by stopping (output: robot name, the time to it should be stopped) one
of the robots based on any assumed priorities (e.g., task urgency, remaining battery life
can be hardcoded in the beginning).
2. Provide a small explanation of how your algorithm ensures reliability and safety.

{so what i think is we need to create a class and consider robots as the objects or something like that and then we will include properties in the object like task urgency, battery remaining , cascading impact, proximity, estimated time to clear etc and we will also build an algo based on the previous task solution which gives priority rating as an output and decides which robot is gonna use the path first and also we need to Detect conflicts in the paths of robots dynamically as they approach the conflicting regions and we will use different test cases to test if our algo is working fine}

{how is my idea , i am open to discussions and any suggestions, think of your self as a developer so that you can help me out with this task}

considerg A as task urgency rating and B as proximity rating and C as cascading impact rating and D as estimated time to clear and E as Battery remaining
and also below weights:
Priority ratings:

Task urgency(3)

proximity(2)

total positive priority rating: 5

Cascading impact(3)

Estimated time to complete the task/Estimated Time to Clear (Conflict Resolution Speed) (1.5)

Low battery(0.5 as it is a rare case)

total negative priority rating: 5

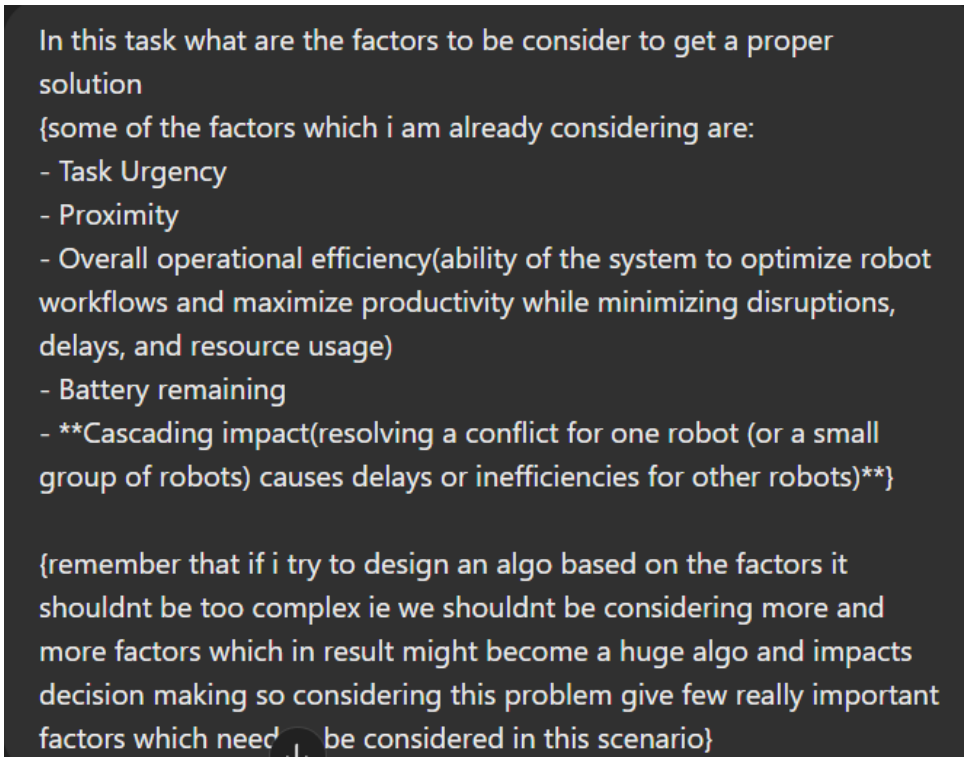
now design a formula which gives overall priority (P) on a scale of 0 to 40

Reasons and Justifications for the formula:

Max Priority a task can have is 40 (+50 and -10)

Min Priority a task can have is -40(10 and -40)

To make sure priority rating is always positive we will add +40 everytime so that rating will always be in the range of $0 \leq P \leq 80$



I've decided to attach entire code with clearly typed explanation than explaining through code snippets

The major problem i faced was during a hackathon where my team need to build a web3 based website and no one among us know how to use solidity to write smart contracts as it was a 24 hr hackathon while 4 of my friends were building the website i started learning solidity then and there and i learnt some good amount within 4-5 hours and wrote some its and bits of code and took help of chatgpt and claude extensively to get a solution i faced a similar situation here i know nothing about the problems robots face as the keep moving in a warehouse so i took the similar approach and built up the required knowledge to start working on the solution

u can see my chat in the below link

https://chatgpt.com/share/677cd4d8-7624-8013-9f85-304ad0509496

my thought process was to build an algo using my knowledge in object oriented programming (c++) and algorithms and it actually reflected in my solution submitted i ve thought of using map to store date so that i get the priority ratings based on the inputs given

the only external tool i used is chatgpt which helped me in understanding the scenario more clearly and it also helped me identifying the factors to be involved

Suggest how your solution would adapt if the number of robots increased significantly?

if number of robots increased significantly i would use machine learning to analyze the priorities of various factors by testing with different cases including huge number of robots and i will get exact weights by using the ml model and will create a better mathematical formula to get better and accurate priority ratings based on the situation in which the robot is.

so by using highly accurate ml models we can handle robots movements in a much better way.

Reliability and safety of the algorithm:

Reliability: The system employs discrepancy resolution in a manner where reliability is maintained when a weighting metric is utilized to determine which robot should have priority in calculating task parameters by estimating aspects such as The urgency of the task, proximity to the area of conflict; operational effectiveness; estimated time of which the robot can clear the area and the battery left on the robot. This bew weighting metric employs predetermined rating maps to determine if there is any need for a particular robot to dominate the other forces in case of a conflict. This is a structured approach that considers reducing the number of subjective decisions depressing into a constant rule that accurately resolves conflicts in real-time while minimizing errors or missed intersections during a reaction and decision-making process.

Safety: Employing safety protocols to stop the robot with less priority also tackles the risk of overlapping direction paths, safely preventing the robots from colliding.

With the algorithm, the critical conditions for these robots were appropriate, such as projected closeness to the battery and level remaining, enabling safe robot operation within these conditions. It also increases the movement and decision-making of the robots as they shift conditions in real-time up faster and prioritize expansive and more urgent tasks or the robots closest to the area of conflict. This flexibility in the robot's decision making, paired with well-laid-out criteria for making decisions, enables a guarantee of safety to be met without any lost time in efficiency.