AI ENDSEM

Kshitij Srivastava (MT18099)

Discussed with : Ashish Sethi (MT18024), Anuparna Deb (MT18001)

Question-1:

A. PEAS :

- **Performance measure**: Route time, number of stops covered, route length, red lights saved.
- **Environment**: Roads, traffic, pedestrians.
- **Actuators**: Steering wheel, brakes, accelerators, traffic signal, horn, driver hands, legs.
- **Sensors**: GPS, driver's eyes, mobile device.

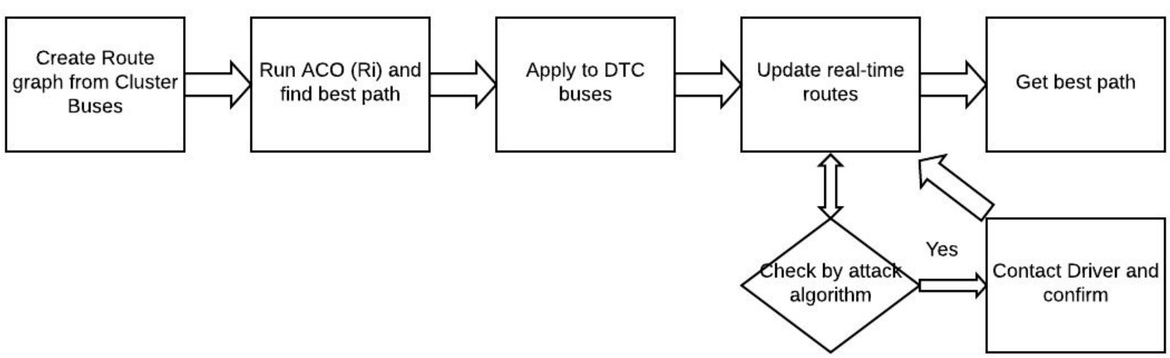
PAGE :

- **Percept**: GPS, driver's eyes, mobile device.
- **Action**: Turn left, turn right, stop, pick up passengers.
- **Goal**: Minimizing route time, number of traffic stops, maximizing the number of customers.
- **Environment**: Roads, traffic, pedestrians.

B. Components to be used in algorithms :

- Routes (difference for up and down)
- Traffic Signals (signals encountered on the route)
- Bus number (Service ID)
- Number of trips to be made on a day
- GPS latitudes and longitudes

C. Flow Diagram :



D. Approach for solution :

- My approach is to solve the problem by using **Ant Colony Optimization**. Quite similar to what we did with the Travelling Salesman problem but with some modifications.
- First, we'll generate the route graph using the given data for Cluster Buses.
- Here we have multiple objectives and constraints that need to be satisfied such as minimum traffic stops should be made, more customers should be picked up, covering the routes quickly.
- Using ACO to create the best path for a route and updating bus routes in real-time using the API.
- I have also used an attack identifying algorithm which will match the current bus route with previous routes that the bus has followed and if there is a significant divergence above a set threshold, it will report an attack.
- At this point, we can manually check with the driver what has happened and identify whether the change is needed or not.

Assumptions :

- Not all nodes have a path between them
- A left turn is preferred by the algorithm.
- Buses follow the same path for up and down.

E. Algorithmic Description :

1. Create a global route graph from Cluster Buses data.
2. Make a list of all bus routes.
3. For all bus routes (R_i):
    a. best_path[R_i] = ACO(R_i)
4. Apply paths thus found for DTC buses.
5. Apply real-time ACO on DTC buses:
    a. Use buses as ants.
    b. Update paths between stops using real-time data if there are any diversions.
    c. Save bus route data to a database for each day
    d. Check the updated path for any attacks
    e. If attack observed :
        i. Contact driver
        ii. Check if the attack is a threat or acceptable
        iii. Update the path

F. Why my approach should work ?

- ACO has been widely used for solving Dynamic Vehicle Routing Problem for quite some time now as it has proven to be the most efficient at solving shortest path problem.
- My algorithm here is just a modified version of ACO which fits the constraints suitable for our problem at hand.
- E.g. : Suppose we need to run a bus route from Govindpuri to India Gate and we are given a list of stops that we need to cover. ACO will find the best path for this route. Further, real-time ACO will take care of the updates that need to be made to the path due to some traffic conditions. However, the algorithm makes sure that the given stops are covered.

G. Attacks :

- GPS coordinates could go wrong.
- There is traffic divergence due to some issues (rallies, riots).

How the algorithms handles these attacks :

- For wrong GPS coordinates, we can directly take help of the driver and give instructions.
- For traffic divergence, the attack identifying algorithm will report an attack. The situation can be solved by contacting the bus driver and making sure in what way the situation can be handled.

REFERENCES :

- https://www.hindawi.com/journals/ddns/2018/1295485/
- https://www.ups.com/us/en/services/knowledge-center/article.page?name=orion-the-algorithm-proving-that-left-isn-t-right&kid=aa3710c2

Question-2 (B):

A. PEAS :

- **Performance measure**: Correct evaluation of sheets, correct grade distribution, student feedback.
- **Environment**: Institution, University, Lecture Hall.
- **Actuators**: OMR checker.
- **Sensors**: Scanner, OCR sensor.

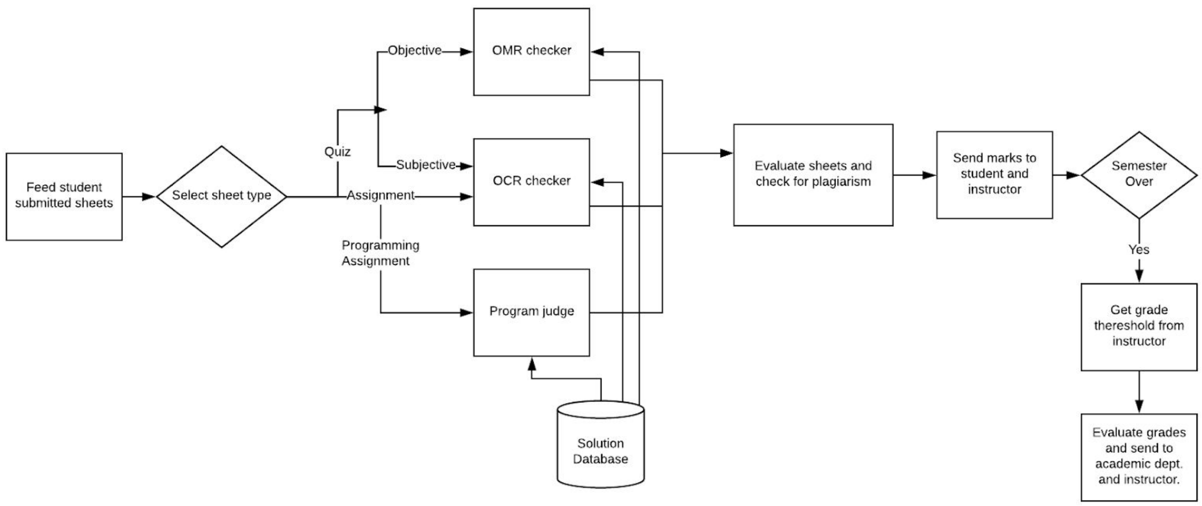
PAGE :

- **Percept**: Scanner, OCR sensor.
- **Action**: Scan sheets, calculate marks, distribute grades.
- **Goal**: Correct evaluation of sheets and grade distribution.
- **Environment**: Institution, University, Lecture Hall.

B. Components to be used in algorithms :

- Students’ sheets (quizzes, assignments, programming homework).
- Students’ feedback regarding the evaluation system.
- OCR algorithm
- OMR checker

C. Flow Diagram :



D. Approach for solution :

- My approach is to design the smart evaluator tool as a combination of some independent tools such as OMR checker, OCR, and program evaluation.
- For objective quizzes, the tool will use OMR checker for checking.
- For subjective quizzes, the smart evaluator will use OCR and check answers against the given solution for maximum similarity.
- If the similarity is above a certain threshold, the student's answer is evaluated as correct otherwise it is incorrect. Special cases can be handled manually by the TAs.
- For assignments, the process is similar to subjective quizzes.
- Programming assignments are checked against a fixed output file containing all different types of outputs (similar to an online judge). There are step marks for solving partial problems.

Assumptions :

- The evaluation machinery works correctly and efficiently.
- Solution database is correct.
- Solution databse has no discrepancy.
- Students' handwriting is recognizable by the system.

E. Algorithmic Description :

1. Feed students' sheets to the smart evaluator tool.
2. Select the type of sheet submitted.
3. For each sheet (S_i) :
    a. Extract data from sheets according to its type.
    b. Retrieve solution from the solution database.
    c. Evaluate answers and check for plagiarism.
    d. Calculate marks.
    e. Send marks to Instructor and students.
    f. Save marks to a database.
4. At semester end:

a. Get grade thresholds from the Instructor.
b. Retrieve marks from the database.
c. Compute the grades of each student.
d. Send the grades to Instructor and academic department.

F. Why my approach should work ?

- The approach should work as this is a simple planning task.
- Students' sheets are given as input to the evaluator and are evaluated against a fixed set of solutions saved in the solution database.
- Besides, there is no human interference in the process and no learning involved.
- The student answers are first converted into digital data and checked for plagiarism.
- E.g. : This tool can be deployed in IIIT Delhi and used for evaluation during the entire semester. It also guarantees fairness.

G. Fairness :

- The algorithm makes sure that there is fairness since all students' sheets are evaluated against a fixed solution database by the evaluator tool.
- No bias is given to any student, no matter how the handwriting is or how good the student has been previously. If a student performs well in the given submission, he will get what marks he deserves.