**KEYWORDS** A

* Management of delivery routes - The process of planning and controlling the most efficient paths and schedules for vehicles to transport goods between collection points and delivery locations.
* Optimization problem - A mathematical problem in which the goal is to find the best solution from a set of feasible solutions, typically by minimizing or maximizing an objective function (such as travel time or cost).
* Shortest paths - The problem of finding the minimum distance or minimum time required to travel between two points in a network or graph, often used in route planning.
* Decision problem - A problem that has a simple yes/no answer.
* Complexity class (asymptotic) - A classification of computational problems based on how the resources (time, space) required to solve them scale with the size of the input.
* Traveling Salesman Problem – An optimization problem where the objective is to find the shortest possible route that visits a set of cities exactly once and returns to the starting city.
* Heuristic - A problem-solving technique that produces approximate solutions when an exact solution is computationally infeasible.
* Exponential complexity - A type of computational complexity where the time or space required to solve a problem grows exponentially with the input size.
* Metric version - A variation of a problem where the distances or costs between points satisfy the triangle inequality, which states that the direct path between two points is always shorter than any indirect path.
* NP-Complete - A class of problems that are both in NP (nondeterministic polynomial time) and as hard as any other problem in NP.
* NP - A class of decision problems for which a proposed solution can be checked quickly (in polynomial time) by a deterministic Turing Machine, though finding the solution may not necessarily be quick.
* Polynomial-time reduction - A technique used to show that one problem is at least as hard as another by transforming one problem into another in polynomial time.
* Turing Machine - A mathematical model of computation used to define algorithmic processes. It’s used to explore the limits of what can be computed and to classify problems into complexity classes.
* Certificate algorithm - An algorithm that verifies whether a given solution to a decision problem is correct.
* Decision problem modelling - The process of framing an optimization problem as a decision problem, which simplifies the task of determining its computational complexity and aids in selecting appropriate solution methods.
* Modern computing system - Refers to contemporary computer systems, which have significantly more processing power and memory than older systems, making them capable of handling larger datasets and more complex algorithms, especially in problems involving space complexity concerns.

**CONTEXT** A

You and Agathe are tackling the challenge of optimizing delivery routes, but before jumping into algorithm design, Agathe emphasizes the need to properly model the problem, understand its complexity, and determine whether it's NP-Complete, guiding your approach toward feasible solutions.

**PROBLEM STATEMENT** A

How can understanding the complexity class of a decision problem help in formulating strategies to effectively optimize delivery routes, considering the computational challenges and constraints involved?

**CONSTRAINTS** A

* Prove NP
* Should go through every city
* Time optimization

**SOLUTION APPROACH** A

* Prove NP
* Should go through every city
* Time optimization

**DELIVERABLE** A

* Which complexity class it falls to prove it
* Algorithm (not high priority but check)

**ACTION PLAN** A

* Study:
  + TSP (verify its relevance for the prosit)
  + NP / NP types / classes
  + Heuristic algorithm
  + Decision and Optimization problem (what/why/how)
* Compare decision & optimization problem
* Understand the complexity class it falls under
* Write optimized algorithm that respects constraints

**NOTES** A

Some questions to answer:

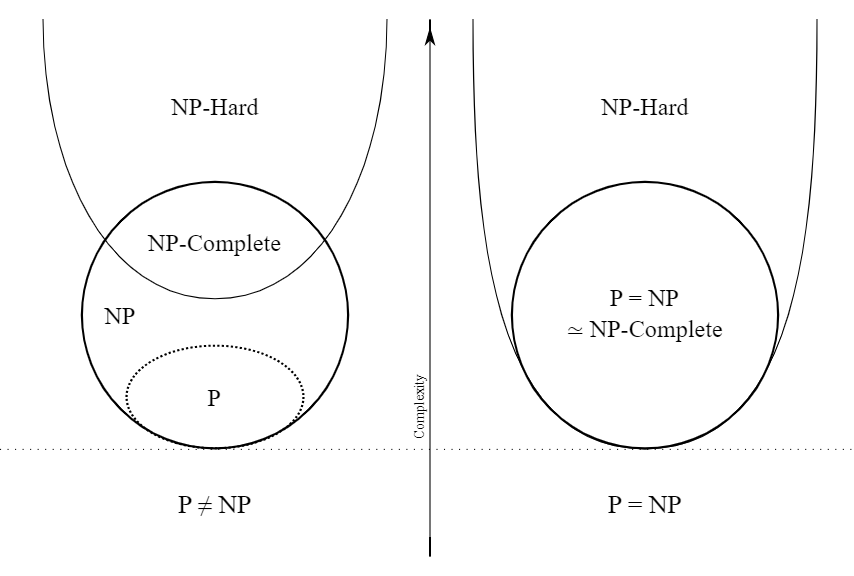
1. How or where does the complexity of our algorithm fall using deterministic Turing Machine?
2. What is the diff b/w decision and optimization problem?
3. Our algo falls under which class complexity?

**TRAVELLING SALESMAN PROBLEM [TSP]**

* The TSP is an optimization problem where a salesman must visit a set of cities, each exactly once, and return to the starting city while minimizing the total travel distance or cost.
* Types of TSP:
  + Symmetric TSP: The distance between two cities is the same in both directions (graph edges are undirected).
  + Asymmetric TSP: The distance between two cities can vary depending on the direction (graph edges are directed).
  + Metric TSP: A special case where distances satisfy the triangle inequality, meaning traveling directly between two points is always the shortest path (e.g., road networks where this property holds).
* Relevance to Our Project:
  + Pros: TSP aligns well with our problem if our objective is to minimize travel time across multiple delivery points. The constraint of visiting each location only once is similar to our situation.
  + Cons: In our case, the problem involves time-dependent travel times due to traffic, which adds a dynamic element not typically modelled by the TSP. The TSP assumes static costs between nodes, whereas you have variable travel times.
* Application in Delivery Route Optimization:
  + Metric TSP may be relevant as it accounts for real-world conditions (like obeying the triangle inequality), but the dynamic nature of your problem may require further adaptations. You might need to explore variants such as the Time-Dependent TSP (TD-TSP).
* TSP with triangle inequality: The cost function c satisfies the triangle inequality if for all vertices u,v,w ∈ V:
  + c(u,w) ≤ c(u,v) + c(v,w).
  + TSP with triangle inequality is NP-Complete: also known as metric TSP or constrained TSP.

**ASYMPTOTIC COMPLEXITY**

* NP (Nondeterministic Polynomial time):
  + A class of decision problems for which a given solution can be verified in polynomial time by a deterministic Turing machine.
  + Example: In the TSP, if a proposed solution (route) is given, checking whether it satisfies the conditions (visits each city once and has a total cost below a threshold) can be done in polynomial time.
* NP-Complete:
  + Problems that are both in NP and as hard as any problem in NP, meaning they can be reduced to one another in polynomial time. If you can solve an NP-Complete problem in polynomial time, you can solve all NP problems in polynomial time.
  + Example: TSP in its general form is NP-Complete, meaning that finding the optimal route in polynomial time is not feasible unless P = NP.
* NP-Hard:
  + Problems that are at least as hard as NP-Complete problems but may not be in NP (i.e., their solutions may not be verifiable in polynomial time).
  + Example: Some variations of the TSP (e.g., Time-Dependent TSP with constraints) may be NP-Hard.



* Practical Implications:
  + If our problem is NP-Complete, it is computationally difficult to find an exact solution for large instances. Heuristic algorithms may be necessary to obtain near-optimal solutions within a reasonable timeframe.

**HEURISTIC ALGORITHM**

* A heuristic is a strategy or method that guides problem-solving towards a satisfactory solution when an exact solution is not feasible due to time or computational constraints. Heuristics are typically used in optimization problems to find good (but not necessarily optimal) solutions quickly.
* Common Heuristics in Route Optimization:
  + Nearest Neighbor: Start at a node and repeatedly visit the nearest unvisited node. While fast, it often leads to suboptimal solutions.
  + Greedy Algorithm: Make the locally optimal choice at each step with the hope of finding a global optimum.
  + Simulated Annealing: A probabilistic technique that explores different solutions by occasionally accepting worse solutions to escape local minima and potentially find a global minimum.
  + Genetic Algorithm: Uses evolutionary techniques such as selection, crossover, and mutation to evolve a population of solutions toward the best one.
* Application in Our Case:
  + Heuristics are especially useful for large-scale delivery route optimization, where solving the problem exactly might take too long. You can start with simple methods like Nearest Neighbor or explore more advanced techniques like Genetic Algorithms that consider time-dependent factors like traffic.

**DECISION PROBLEM VS OPTIMIZATION PROBLEM**

Decision Problem:

* What: A problem with a yes/no answer. For example, "Is there a delivery route such that the total travel time is less than a given threshold?"
* Why: Decision problems are important because their complexity can often be analyzed more easily. Understanding the complexity of the decision problem helps determine the feasibility of solving the related optimization problem.
* How: To solve a decision problem, we frame our optimization question (e.g., minimizing travel time) into a binary question. Then we can analyze if this decision problem is in NP (can a solution be verified quickly).

Optimization Problem:

* What: A problem where the goal is to find the best solution from a set of possible solutions. For example, "What is the delivery route with the shortest travel time?"
* Why: The optimization problem is our ultimate goal, as we want to minimize travel times. However, it is usually harder to solve directly compared to its decision counterpart.
* How: To solve an optimization problem, we often start by solving a related decision problem (e.g., "Is there a route with time < X?") and then use algorithms or heuristics to explore possible solutions and optimize them.

Importance of Decision Problem Modeling:

Decision problems can help understand the problem's complexity (whether it’s in NP, NP-Complete, etc.). Solving decision problems can guide toward solving or approximating the optimization problem more effectively.