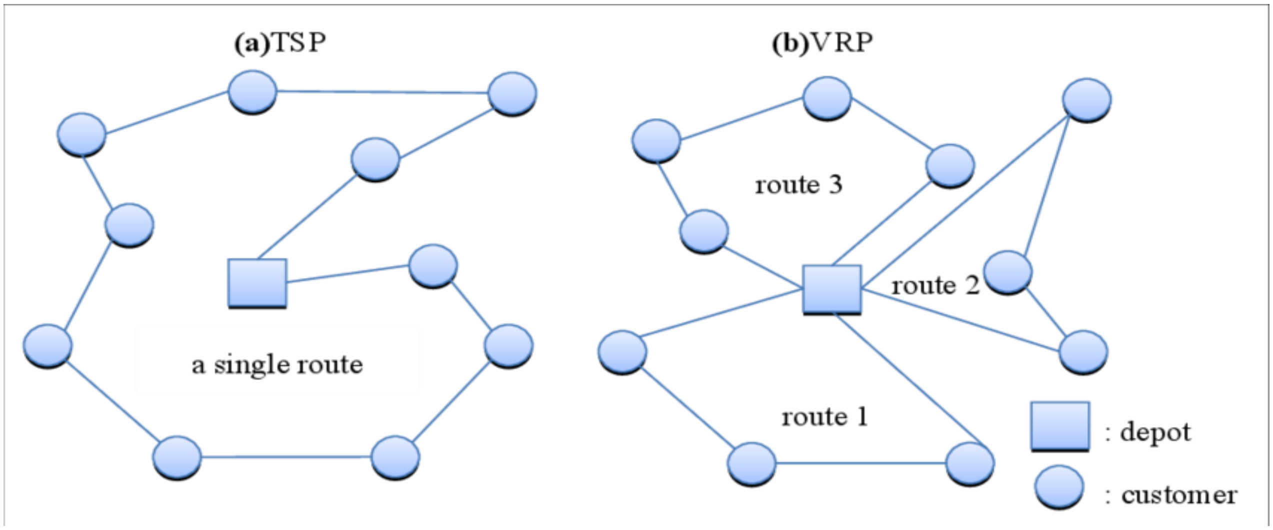
[2023]

Traveling Salesman Problem.

By:
ENG: ESRAA MOHAMED HAMAD

Traveling Salesman Problem.

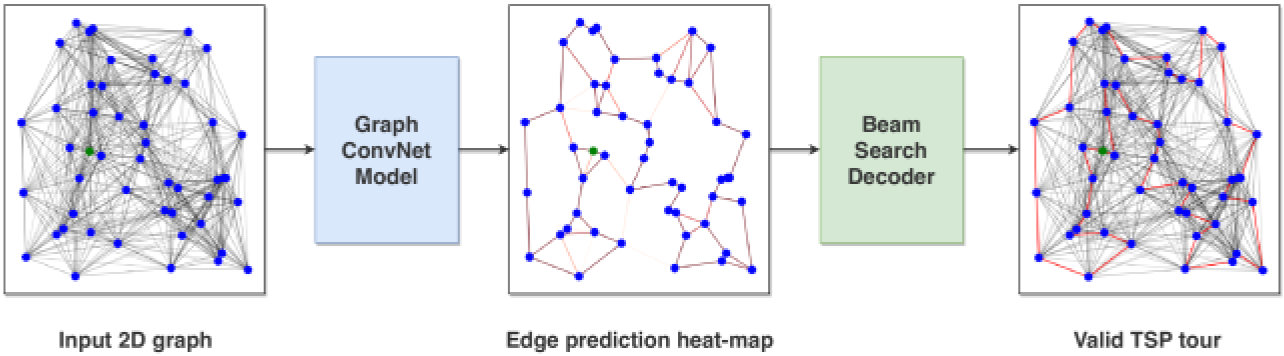


Summary:

The traveling salesman problem, TSP for short, has model character in many branches of mathematics, computer science, and operations research. Heuristics, linear programming, and branch and bound, which are still the main components of today's most successful approaches to hard combinatorial optimization problems, were first formulated for the TSP and used to solve practical problem instances. When the theory of NP-completeness developed, the TSP was one of the first problems to be proven NP-hard by Karp in 1972. New algorithmic techniques have first been developed for or at least have been applied to the TSP to show their effectiveness. Examples are branch and bound, Lagrangean relaxation, Lin-Kernighan type methods, simulated annealing, and the field of polyhedral combinatorics for hard combinatorial optimization problems. The chapter presents a self-contained introduction into algorithmic and computational aspects of the traveling salesman problem along with their theoretical prerequisites as seen from the point of view of an operations researcher who wants to solve practical instances.

Solutions Of Traveling Salesman Problem:

- naive bayes algorithm.
- dynamic programming algorithm.
- approximation algorithms.



A Summary Of Our Project Idea:

- We will design a program containing a graph consisting of a several different methods and spaced points.
- We will enter one input **(start point)** .
- Our program process this input and try to fetch the shortest route.
- Output is the shortest route between start point and end point **(is the same start point)** ,You must return the starting point without going through the station twice.

Agent(PEAS):

- **TSP agent(PEAS):**
  1. **P** ➡ Speed , Find shortest route.
  2. **E** ➡ Cost of routes.
  3. **A** ➡ Process the route of the station.
  4. **S** ➡ Calc the lowest cost of every route.

Environment(ODESA):

- **TSP (ODESA):**
  1. **O** ➡ Fully Observable.
  2. **D** ➡ Deterministic.
  3. **E** ➡ Sequential.
  4. **S** ➡ Static.
  5. **A** ➡ Single-agent.

Problem Formulation

Initial State

Any edge on the graph could be initial state.

Actions

Determine each edge it pass through and not pass through it again until reach the goal test and cost calculation.

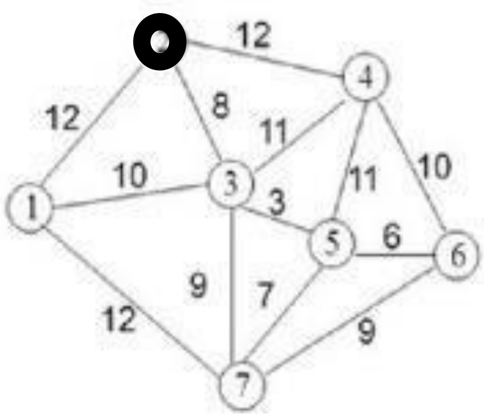
Goal Test

It is the goal that you want to reach after carrying out some actions.

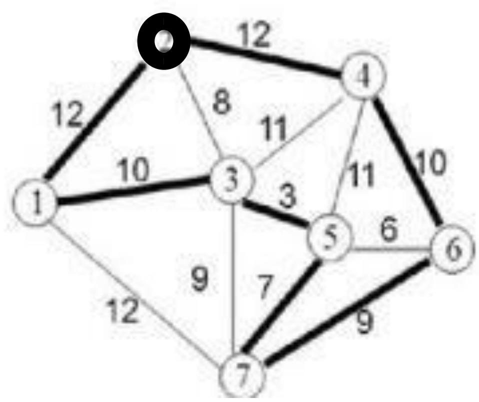
Path Cost

It is the cost of the path between the initial state (start edge) and goal test (the same start edge).

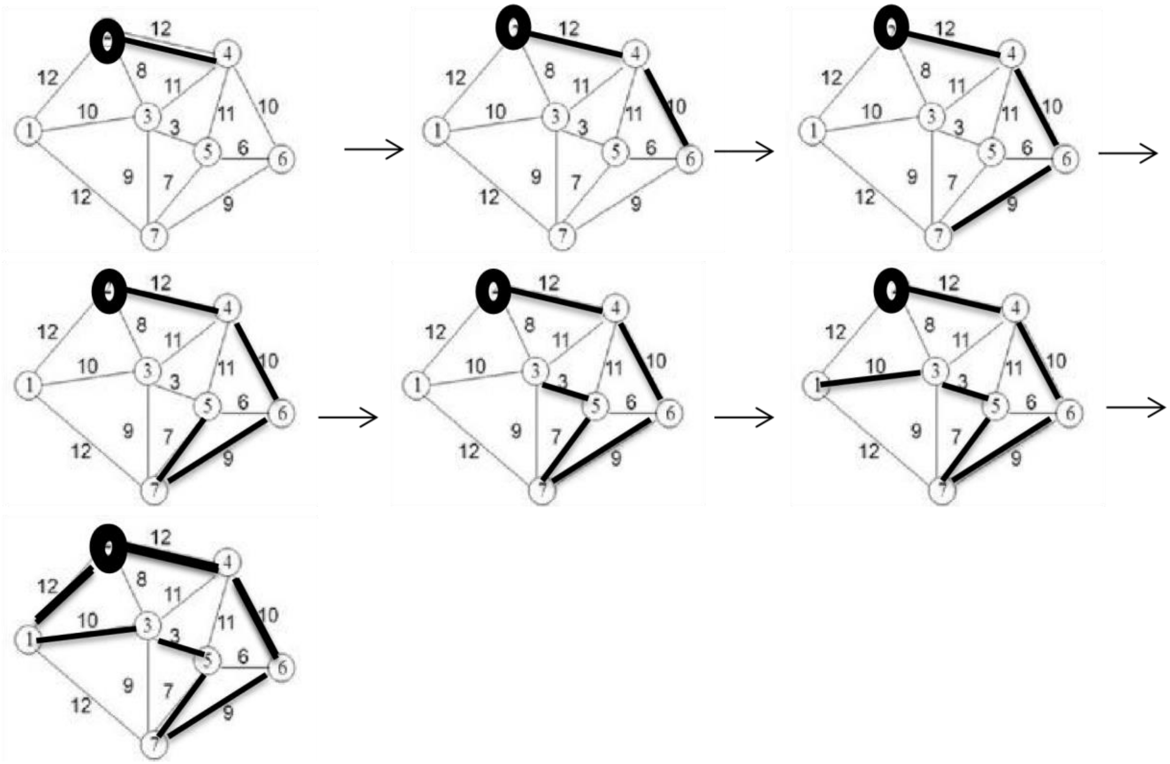
Transition Model



Initial state



Goal test


12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7
12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7
12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7
12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7
12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7
12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7
12 12 8 11 4 10 1 3 3 11 5 6 6 9 7 12 9 7