13. Path Planning

金力 Li Jin
li.jin@sjtu.edu.cn
上海交通大学密西根学院
Shanghai Jiao Tong University UM Joint Institute

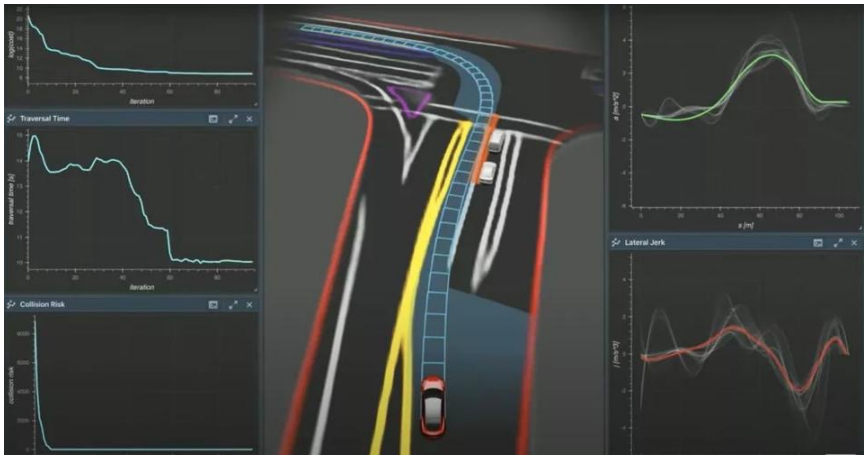
上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Outline

- Path planning in Euclidean spaces
  - One-dimension
  - Two-dimension
- Path planning on networks
  - Minimum spanning tree problem
  - Traveling salesman problem
  - Chinese postman problem

Objective

- We have a vehicle that needs to travel from an origin to a destination.
- For 1D, the orbit is known, but the spatio-temporal trajectory is to be determined.
- For 2D, the orbit is unknown, and the geometry of the orbit as well as the waypoint times are to be determined.



1D: Problem formulation

- Data
  - Vehicle parameters
  - Locations of the origin and the destination
- Decision variables
  - Spatio-temporal trajectory
  - Denoted by $\{x(t); t = 0,1,2, \dots\}$
- Constraints
  - Vehicle dynamics
- Objective
  - Option 1: reach the destination as soon as possible
  - Option 2: consume as little energy as possible
  - Option 3: a weighted sum of the above

1D: Problem formulation

- Data
  - Location of origin: $x = 0$ [m]
  - Location of destination: $x = D$ [m]
  - Maximal speed $\bar{v}$ [m/s]
  - Maximal acceleration (speed increment) $\bar{a}$ [m/s^2]
  - Fuel rate f [L/s]: $f = \alpha v^2$.
  - Rationale: air drag is proportional to v^2. (Let's use this simple model for now.)
  - Assume unit time step $\Delta t = 1$ [s]
- Decision variables
  - $v(t)$ for $t = 0,1,2, \ldots$
  - Speed will determine acceleration and position.

1D: Problem formulation

- We need to first resolve a technical issue: what is the dimension of v? (We cannot deal with an infinite set of decision variables $\{v(t); t = 0,1,2 \dots\}$ in practice.)
- We can address this issue by computing an apparent upper bound on the traverse time.
- Traverse time = how long it takes for the vehicle to move from the origin to the destination.
- This upper bound is typically determined by prior knowledge.
- Let T be the upper bound.
- So, the decision variables are $v(1), v(2), \dots, v(T)$

1D: Problem formulation

- Constraints are as follows.
  - Origin & destination

$$x(0) = 0$$
$$x(T) = D$$

  - Kinematic equation

$$x(t+1) = x(t) + v(t)$$
$$a(t) = v(t) - v(t-1)$$

  - Technological constraint (saturation)

$$-\bar{a} \le a(t) \le \bar{a}$$
$$0 \le v(t) \le \bar{v}$$

- Objective function
  - One-step fuel cost: $\alpha v(t)^2$
  - Cumulative cost: $\sum_{t=0}^{T} \alpha v(t)^2$

1D: Problem formulation

A standard optimization formulation:

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\begin{aligned}
\text{s.t.} \quad & x(0) = 0, x(T) = D, \\
& x(t+1) = x(t) + v(t), t = 0,1,\dots,T-1, \\
& -\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2,\dots,T, \\
& 0 \le v(t) \le \bar{v}, t = 0,1,\dots,T.
\end{aligned}$$

Data? Decision variables? Constraints? Objective function?

1D: Optimization

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\begin{aligned}
\text{s.t.} \quad & x(0) = 0, x(T) = D, \\
& x(t+1) = x(t) + v(t), t = 0,1, \dots, T-1, \\
& -\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2, \dots, T, \\
& 0 \le v(t) \le \bar{v}, t = 0,1, \dots, T.
\end{aligned}$$

- Mathematical classification
- Feasibility
- Existence of optimal solution(s)
- Computation of optimal solution(s)

1D: Optimization: Mathematical classification

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\begin{aligned}
\text{s.t.} \quad & x(0) = 0, x(T) = D, \\
& x(t+1) = x(t) + v(t), t = 0,1,\dots,T-1, \\
& -\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2,\dots,T, \\
& 0 \le v(t) \le \bar{v}, t = 0,1,\dots,T.
\end{aligned}$$

- Linear constraints
- Quadratic objective function
- Classification: quadratic programming with linear constraints（线性约束二次规划）.

1D: Optimization: Mathematical classification

- Consider a **real-valued** optimization problem

$$\min f(x)$$
$$s.t.\ g(x) \leq 0$$
$$x \in \mathbb{R}$$

- If f and g are both linear, we have a linear programming（线性规划）.
- If f or g is quadratic **or linear** (but not both linear), we have a quadratic programming（二次规划）.
- If f or g is nonlinear, we have a nonlinear programming（非线性规划）.
- Linear programming is easy to solve, and nonlinear programming is in general hard (or not that easy).

1D: Optimization: Mathematical classification

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\begin{aligned}
\text{s.t.} \quad & x(0) = 0, x(T) = D, \\
& x(t+1) = x(t) + v(t), t = 0,1, \dots, T-1, \\
& -\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2, \dots, T, \\
& 0 \le v(t) \le \bar{v}, t = 0,1, \dots, T.
\end{aligned}$$

- So, the 1D trajectory planning problem is a linearly-constrained quadratic programming.

1D: Optimization: Mathematical classification

- Consider a generic optimization problem

$$\min f(x)$$
$$s.t.\ g(x) \leq 0$$
$$x \in \mathbb{X}$$

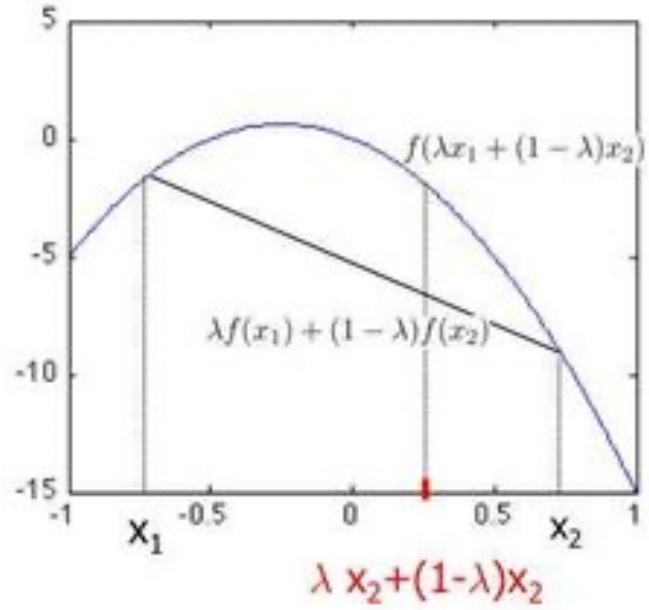
- An optimization problem may have not constraints -> unconstrained optimization.
- If $\mathbb{X}$ (the set of x) is the set of integers -> integer programming（整数规划）.
- If $\mathbb{X} = \mathbb{R}$ and if both f and g are convex in x -> convex optimization（凸优化）.

1D: Optimization: Mathematical classification

- Convex 凸 & concave 凹

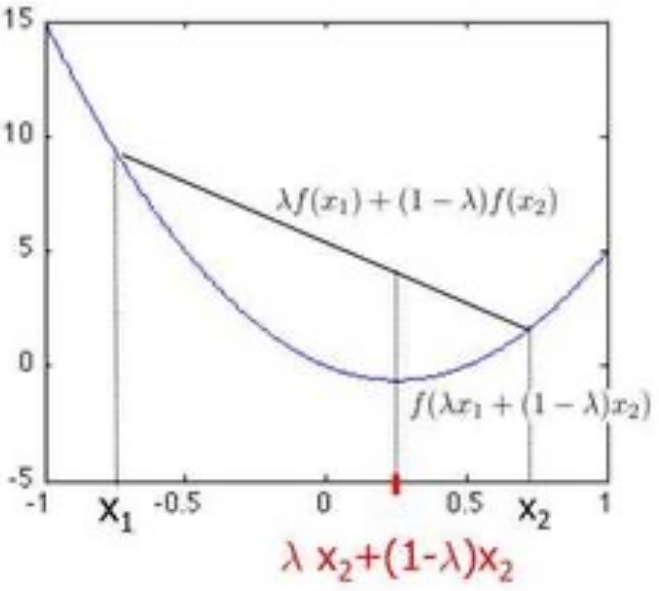
f is ***concave*** if and only

$$\forall x_1, x_2, \quad \forall \lambda \in (0,1),$$
$$f(\lambda x_1 + (1-\lambda)x_2) \geq \lambda f(x_1) + (1-\lambda) f(x_2)$$



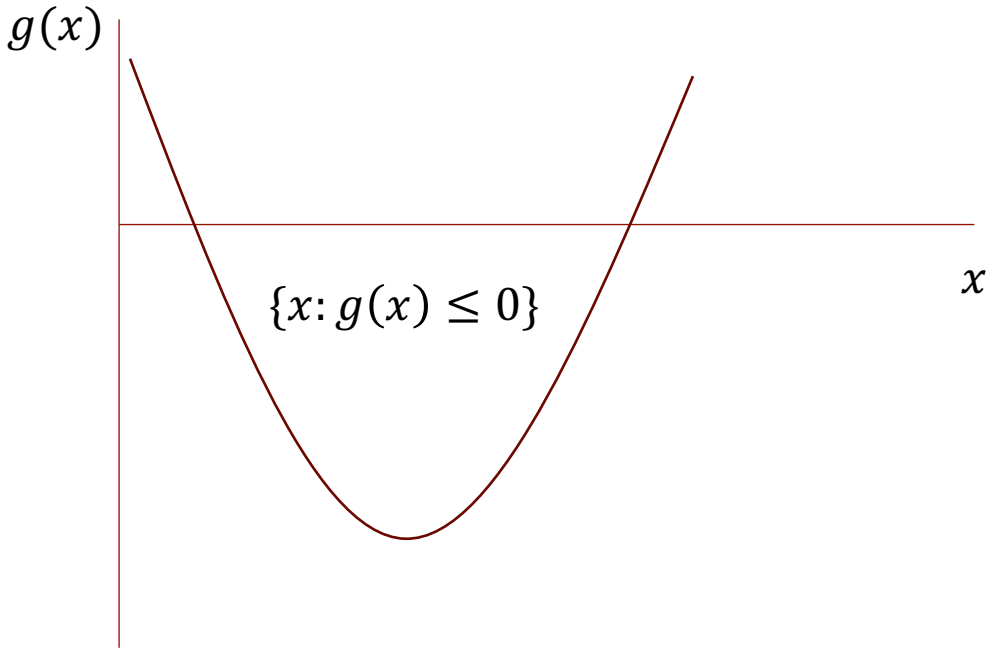
f is ***convex*** if and only

$$\forall x_1, x_2, \quad \forall \lambda \in (0,1),$$
$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$



1D: Optimization: Mathematical classification

- If $g(x)$ is convex in x, then $\{x: g(x) \leq 0\}$ is a convex set.
- Hence, $g(x) \leq 0$ is called the standard form of constraints



1D: Optimization: Mathematical classification

- Why do we care about convexity?
- Because we can solve an unconstrained convex optimization

$$\min f(x)$$

by solving

$$\nabla_x f(x) = 0$$

(if f is differentiable.)

- We can also solve a constrained convex problem in a systematic manner
  - An optimal solution is guaranteed.
  - Note: convex problem requires both convex objective function and convex constraints!

1D: Optimization: Mathematical classification

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\begin{aligned}
\text{s.t.} \quad & x(0) = 0, x(T) = D, \\
& x(t+1) = x(t) + v(t), t = 0,1, \dots, T-1, \\
& -\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2, \dots, T, \\
& 0 \le v(t) \le \bar{v}, t = 0,1, \dots, T.
\end{aligned}$$

- The 1D trajectory planning problem is convex.
- So you can solve it using optimization tools in Python/MATLAB…

1D: feasibility

- Recall that the trajectory planning problem is subject to the following constraints

$$x(0) = 0, x(T) = D,$$
$$x(t+1) = x(t) + v(t), t = 0,1, \dots, T-1,$$
$$-\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2, \dots, T,$$
$$0 \le v(t) \le \bar{v}, t = 0,1, \dots, T.$$

- Any set of values for $\{v(t); t = 1,2, \dots, T\}$ satisfying the above are called a feasible solution（可行解）.
- If there exists at least one feasible solution, the problem is said to be feasible.

1D: feasibility

- When would the problem be infeasible?

$$x(0) = 0, x(T) = D,$$
$$x(t+1) = x(t) + v(t), t = 0,1, \dots, T-1,$$
$$-\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2, \dots, T,$$
$$0 \le v(t) \le \bar{v}, t = 0,1, \dots, T.$$

- T too small; D too large
- $\bar{a}$ too small; $\bar{v}$ too small
- Practical interpretation?

1D: existence of optimal solution(s)

- Definition: consider an optimization problem

$$\min f(x)$$
$$s.t.\ g(x) \leq 0$$

A feasible solution x^* is said to be an optimal solution if for any feasible solution x', we have

$$f(x^*) \leq f(x').$$

- Example:

$$\min x^T x \text{ s.t. } x \in \mathbb{R}^2$$

has a unique optimal solution

$$x^* = [0\ 0]^T$$

1D: existence of optimal solution(s)

- A feasible optimization problem does not always have an optimal solution!
- Counter example 1:

$$\min x \qquad \text{s.t. } x \leq 0$$

- Counter example 2:

$$\min \frac{1}{x} \qquad \text{s.t. } x \geq 1$$

- For most engineering problems, if your formulation is consistent with reality, then an optimal solution must exist.
- Otherwise, your formulation is wrong.

1D: computation of optimal solution(s)

- Within the scope of this course, you can solve the trajectory planning problem

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\begin{aligned}
\text{s.t.} \quad & x(0) = 0, x(T) = D, \\
& x(t+1) = x(t) + v(t), t = 0,1,\dots,T-1, \\
& -\bar{a} \le v(t) - v(t-1) \le \bar{a}, t = 1,2,\dots,T, \\
& 0 \le v(t) \le \bar{v}, t = 0,1,\dots,T.
\end{aligned}$$

using Python/MATLAB.

- You just need to know how to code the optimization model and how to run the optimization tool.

1D: computation of optimal solution(s)*

- Beyond the scope of this course, there is extensive theory on computing optimal solutions.
- *People spend career on this topic.*
- Analytical approaches: exactly characterize the structure and properties of optimal solutions
- Numerical approaches: use iterative algorithms to asymptotically search the optimal solutions
- Learning-based approaches: observe the optimal solution of similar problems and predict the most likely optimal solution

2D: Problem formulation

- Data
  - Vehicle parameters
  - Locations of the origin and the destination
- Decision variables
  - Spatio-temporal trajectory
  - Denoted by $\{x(t) \in \mathbb{R}^2; t = 0,1,2, \dots\}$
- Constraints
  - Vehicle dynamics
- Objective
  - Fuel + time

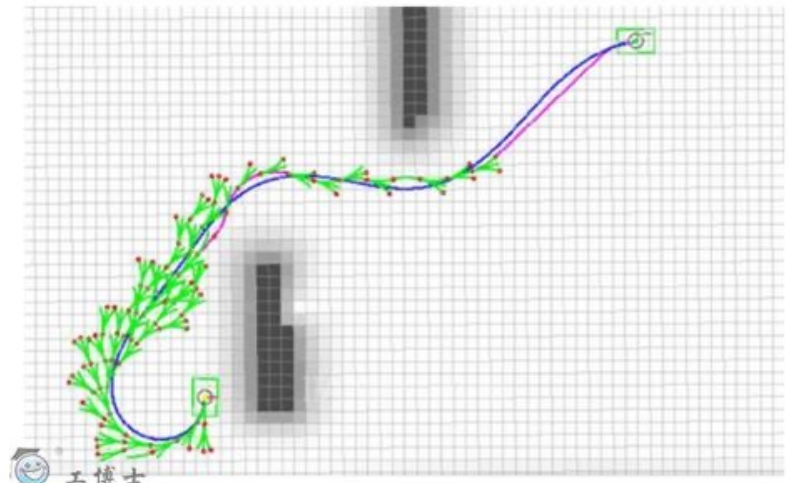
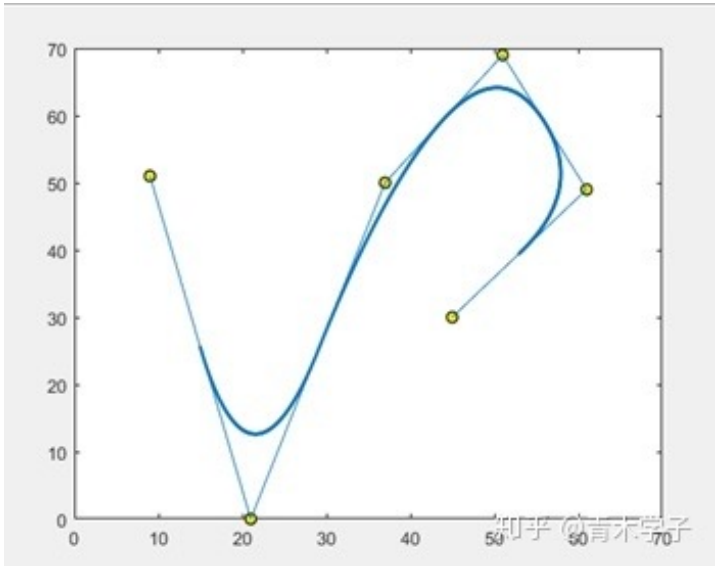


图2 启发式搜索到最优路径



2D: Problem formulation

- Data
  - Location of origin: $x = [0,0]$ [m]
  - Location of destination: $x = [D_1, D_2]$ [m]
  - Road space $\mathcal{X}$
  - Maximal speed $\bar{v}$ [m/s]
  - Maximal acceleration (speed increment) $\bar{a}$ [m/s^2]
  - Maximal angular speed $\bar{\omega}$ [rad/s]
  - Maximal angular acceleration $\bar{\phi}$ [rad/s^2]
  - Fuel rate f [L/s]: $f = \alpha v^2$.
  - Rationale: air drag is proportional to v^2. (Let's use this simple model for now.)
  - Assume unit time step $\Delta t = 1$ [s]

2D: Problem formulation

- Decision variables
  - Speed $v(t)$ for $t = 0,1,2, \ldots$
  - Angular speed $\omega(t)$ for $t = 0,1,2, \ldots$
  - Speed & angular speed will determine acceleration and position.
- Objective function
  - One-step fuel cost: $\alpha v(t)^2$
  - Cumulative cost: $\sum_{t=0}^{T} \alpha v(t)^2$

2D: Problem formulation

- Constraints are as follows.
  - Origin & destination & road space

$$x(0) = [0,0], \qquad \theta(0) = \theta_0$$
$$x(T) = [D_1, D_2]$$
$$x(t) \in \mathcal{X}$$

  - Kinematic equation

$$x(t+1) = x(t) + \begin{bmatrix} v(t)\cos\theta(t) \\ v(t)\sin\theta(t) \end{bmatrix}$$
$$\theta(t+1) = \theta(t) + \omega(t)$$
$$a(t) = v(t) - v(t-1)$$
$$\phi(t) = \omega(t) - \omega(t-1)$$

  - Technological constraint (saturation)

$$-\bar{a} \le a(t) \le \bar{a}, \qquad 0 \le v(t) \le \bar{v}$$
$$-\bar{\phi} \le \phi(t) \le \bar{\phi}, \qquad -\bar{\omega} \le \omega(t) \le \bar{\omega}$$

2D: Problem formulation

A standard optimization formulation:

$$\min \sum_{t=0}^{T} \alpha v(t)^2$$

$$\text{s.t.} \quad x(0) = [0,0],\ \theta(0) = \theta_0, x(T) = [D_1, D_2],\ x(t) \in \mathcal{X},$$

$$x(t+1) = x(t) + \begin{bmatrix} v(t)\cos\theta(t) \\ v(t)\sin\theta(t) \end{bmatrix},$$

$$\theta(t+1) = \theta(t) + \omega(t), a(t) = v(t) - v(t-1),$$

$$\phi(t) = \omega(t) - \omega(t-1),$$

$$-\bar{a} \le a(t) \le \bar{a},\ 0 \le v(t) \le \bar{v}$$

$$-\bar{\phi} \le \phi(t) \le \bar{\phi},\ -\bar{\omega} \le \omega(t) \le \bar{\omega},$$

Outline

- Path planning in Euclidean spaces
  - One-dimension
  - Two-dimension
- Path planning on networks
  - Minimum spanning tree problem
  - Traveling salesman problem
  - Chinese postman problem

Plane vs. network

- Sometimes, it is natural to discretize a Euclidean space as a network of nodes and links.
- That is, we first reduce the space for trajectories.

Network model

- In a network, the nodes typically stand for street intersections, towns, and cities, and the links for street segments, country roads, and airplane travel times.
- Network is also commonly used in other area such as electrical circuits, project management, and social networks.

Network model

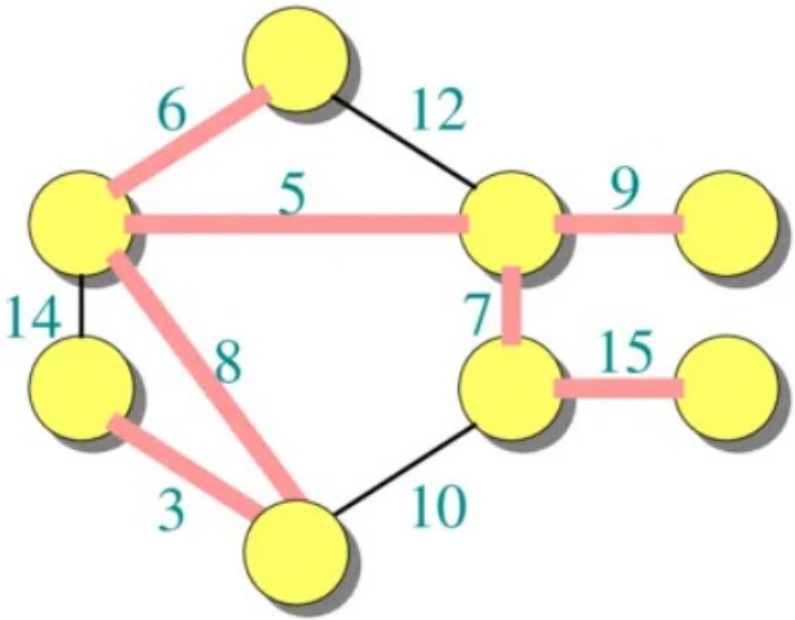
- Arcs can have lengths, which can be either real distances such as the length of a road, or other similar quantities such as travel time or cost.
- Nodes can have weights, which can stand for travel demand/supply, population, or electricity use.
- A sequence of connected nodes is a path.
- A path with its origin and destination coinciding is a cycle.
- Two nodes connected by a links are called adjacent, and so are two links incident to the same node.

Minimum spanning tree (MST) problem

- Consider a newly built residential community.
- We now want to connect all buildings in the community with electricity.
- Cables can only follow the course of roads connecting the buildings.
- Our objective is to minimize the total lengths of the cables while ensuring connectivity of every building.

Minimum spanning tree (MST) problem

- Consider a connected network; i.e., no isolated node or sub-network.
- Find a smallest (i.e., with smallest number of links) sub-network that connects all nodes ("spanning").
- One can show that such a network must be a tree.



Minimum spanning tree (MST) problem

- Decision variable:

$$x_e \in \{0,1\}, \qquad e \in E.$$

- Objective function:

$$\min \sum_{e \in E} x_e \, .$$

- Constraint: the links that we select, i.e., the set of links with $x_e = 1$, forms a tree.
- How to mathematically formulate the tree constraint?

sub-tour elimination

Minimum spanning tree (MST) problem

Suppose that there are n nodes in the network.

- To connect all the n nodes, we need at least $n-1$ links.
- One can show that a minimum spanning tree consists of exactly $n-1$ links.
- Hence, we can use the constraint

$$\sum_{e \in E} x_e = n - 1.$$

Then, how do we ensure a tree?

- Note: tree = no cycles, or "sub-tours" $C \subset E$.
- Let $\mathcal{C}$ be the set of all possible sub-tours in the network.

$$\sum_{e \in C} x_e \leq |C| - 1, \qquad \forall C \in \mathcal{C}.$$

cardinality of C

Minimum spanning tree (MST) problem

In conclusion, we formulate MST problem as follows:

$$\begin{aligned} \min\ & 0 \quad \text{(dummy objective function)} \\ \text{s.t.}\ & \textstyle\sum_{e\in E} x_e = n-1, \\ & \textstyle\sum_{e\in C} x_e \le |C|-1,\ \forall C \in \mathcal{C}, \\ & x_e \in \{0,1\},\ e \in E. \end{aligned}$$

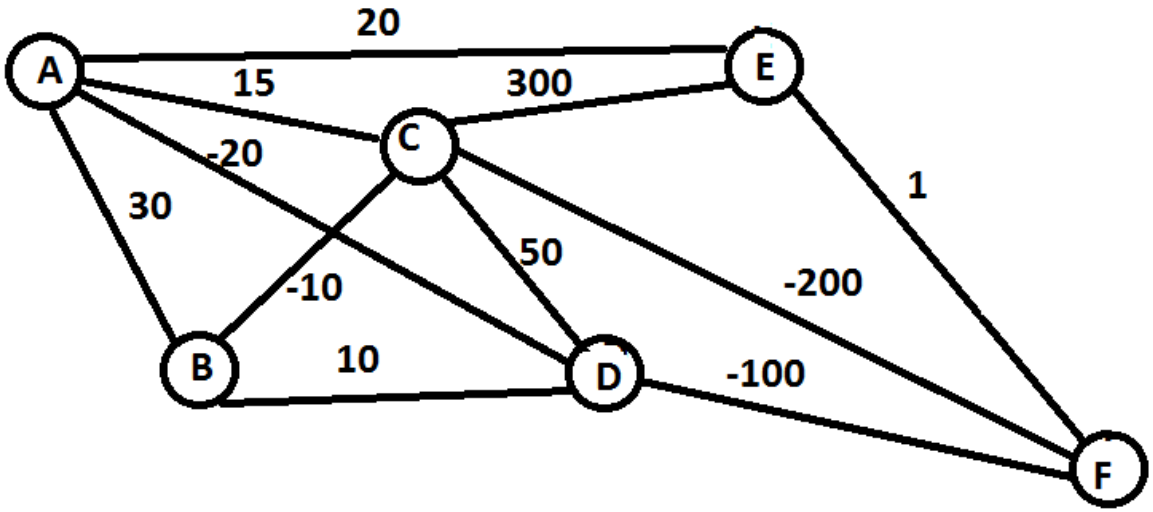
- Integer programming with linear constraints.
- Essentially a feasibility problem rather than optimization problem.
- Feasibility guaranteed theoretically if the network is connected.

Traveling salesman problem (TSP)

- Suppose that a courier has n packages to deliver.
- The courier wants to deliver all n packages with the minimal travel cost, say, shortest total distance.
- The courier needs to return to the storage station to report completion of task.

Traveling salesman problem (TSP)

- Mathematically, we consider the destinations of the n packages as n nodes of a connected network.
- The objective of this problem is to find the shortest tour that passes through every node of the network.



Traveling salesman problem (TSP)

Some history:

- Knight's tour problem (1759), by Leonhard Euler.
- Traveling salesman problem (1959), by George Dantzig.

Traveling salesman problem (TSP)

- We can formulate the TSP as follows:

$$\begin{aligned} \min\ & \textstyle\sum_{(i,j)\in E} f_{ij} \\ \text{s.t.}\ & \textstyle\sum_{i\in In(j)} f_{ij} \geq 1, j \in N, \\ & \textstyle\sum_{i\in In(j)} f_{ij} = \sum_{k\in Out(j)} f_{jk}, j \in N, \\ & f_{ij} \in \mathbb{Z}_{\geq 0},\ (i,j) \in E. \end{aligned}$$

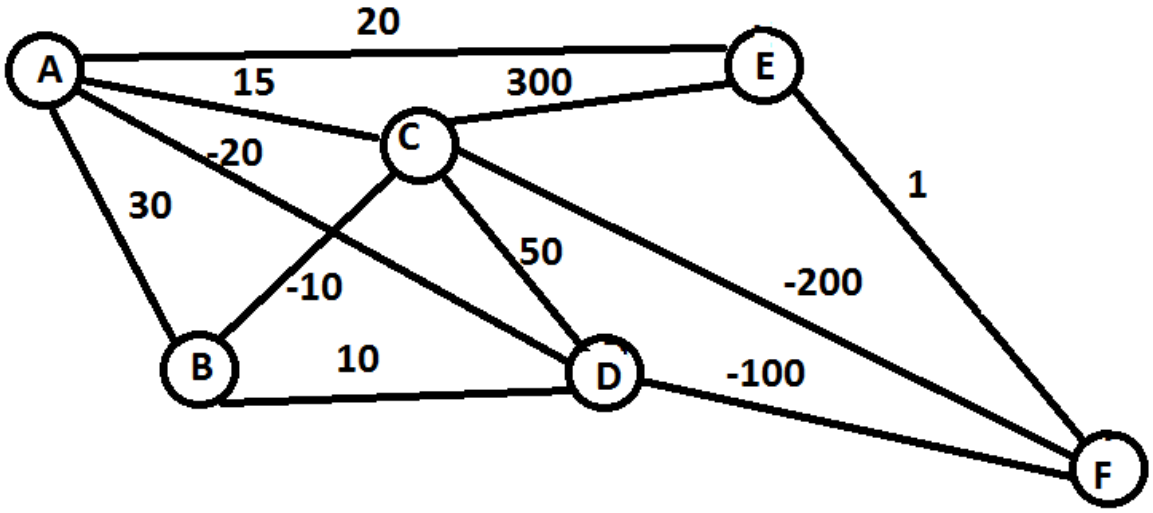
- [Not required] Is the integer constraint necessary?
- This problem has led to a considerable number of heuristics for network optimization.
- The major metrics for a heuristic are the worst-case performance, the average performance, and the computational load.

Chinese postman problem (CPP)

- Consider a sweeper truck that needs to clean all streets in a local community.
- The truck must sweep every street at least once.
- To minimize operation cost, we want to minimize the distance to be covered by the sweeper truck.

Chinese postman problem (CPP)

- Mathematically, we consider a network with m links.
- Link (i, j) has a length of c_{ij}.
- The objective of this problem is to find the shortest tour that covers every link of the network.



Chinese postman problem (CPP)

Some history:

- Seven Bridges of Königsberg (1739), by Leonhard Euler.
- **Kwan Mei-Ko**, *Graphic Programming Using Odd or Even Points*, Chinese Math., 1:273-277, 1962. (管梅谷. 奇偶点图上作业法[J]. 数学学报, 1960(03):263-266.)

Königsberg
(Kaliningrad since 1946)

Kwan Mei-Ko

- 1934-
- B.S. Math, East China Normal University, 1957.
- Former Professor & President of Shandong Normal University.

Chinese postman problem (CPP)

- Kwan's article referred to optimizing a postman's route, was written by a Chinese author, and appeared in a Chinese math journal.
- Based on this Alan J. Goldman suggested the name "Chinese Postman problem" to Jack Edmonds when Edmonds was in Goldman's Operations Research group at the U.S. National Bureau of Standards (now NIST).
- Edmonds appreciated its "catchiness" and adopted it. Goldman was also indirectly influenced by recalling an Ellery Queen mystery named "The Chinese Orange Mystery".

(Alan J. Goldman, personal communication, 14 December 2003)

Chinese postman problem (CPP)

- We can formulate as follows:

$$\begin{aligned} \min\ & \textstyle\sum_{(i,j)\in E} c_{ij} f_{ij} \\ \text{s.t.}\ & f_{ij} \geq 1,\ (i,j) \in E, \\ & \textstyle\sum_{i\in In(j)} f_{ij} = \sum_{k\in Out(k)} f_{jk}, \\ & f_{ij} \in \mathbb{Z}_{\geq 0},\ (i,j) \in E. \end{aligned}$$

- If the solution is a cycle that traverses every edge exactly once, then it is called an Euler tour.
- Similarly, a path that traverses every edge exactly once is called an Euler path.
- Not all graphs have an Euler cycle or path.

Summary

- Path planning in Euclidean spaces
  - One-dimension
  - Two-dimension
- Path planning on networks
  - Minimum spanning tree problem
  - Traveling salesman problem
  - Chinese postman problem