MA4254 Discrete Optimization

Final Examination Helpsheet

AY2024/25 Semester 1 · Prepared by Tian Xiao *@snoidetx*

1 ILP Formulation

Mixed integer linear programs (standard form):

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} + \mathbf{d}^\top \mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{b} \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0};\ \mathbf{x} \in \mathbb{Z}^n;\ \mathbf{y} \in \mathbb{R}^k. \end{aligned}$$

Examples of ILP formulation:

KNAPSACK: item $1, \cdots, n$; weight $w_1, \cdots, w_n$; cost $c_1, \cdots, c_n$;

$$\begin{aligned} \max \quad & \textstyle\sum_i c_i x_i \\ \text{s.t.} \quad & \textstyle\sum_i w_i x_i \leq B \\ & x_i \in \{0,1\}. \quad (i \text{ picked?}) \end{aligned}$$

SHIPPINGCONTAINERS: item $1, \cdots, n$; size $a_{1\cdots n}$; container $1, \cdots, m$; size $b_{1\cdots m}$; binary feasibility problem: whether a feasible point exists.

$$\begin{aligned} & \textstyle\sum_{j=1}^m x_{ij} = 1, \forall i \\ & \textstyle\sum_{i=1}^n a_i x_{ij} \leq b_j, \forall j \\ & \textstyle\sum_{j=1}^m b_j y_j \leq Q \text{ (ship capacity)} \\ & x_{ij} \leq y_j;\ x_{ij}, y_j \in \{0,1\}. \end{aligned}$$

FACILITYLOCATION (relax): service locations $1, \cdots, m$; customers $1, \cdots, m$; cost for j to be switched on: c_j; cost for i to be served by j: d_{ij};

$$\begin{aligned} \min \quad & \textstyle\sum_j c_j y_j + \sum_j \sum_i d_{ij} x_{ij} \\ \text{s.t.} \quad & \textstyle\sum_j x_{ij} = 1, \forall i \\ & x_{ij} \leq y_j, \forall i, j \\ & x_{ij}, y_j \in \{0,1\}. \end{aligned}$$

PACKING: item $1, \cdots, n$; disjoint subset $F_1, \cdots, F_n$; find collection of subsets with highest value; incident matrix $A_{ij} = 1$ if $j \in F_i$;

$$\begin{aligned} \max \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{x} \leq \mathbf{1} \\ & \mathbf{x} \in \{0,1\}^m. \end{aligned}$$

COVERING: item $1, \cdots, n$; disjoint subset $F_1, \cdots, F_n$; optimize objective when all items are covered. incident matrix $A_{ij} = 1$ if $j \in F_i$;

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{x} \geq \mathbf{1} \\ & \mathbf{x} \in \{0,1\}^m. \end{aligned}$$

PARTITIONING: item $1, \cdots, n$; disjoint subset $F_1, \cdots, F_n$; find partition that maximizes objective. incident matrix $A_{ij} = 1$ if $j \in F_i$;

$$\begin{aligned} \max \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{x} = \mathbf{1} \\ & \mathbf{x} \in \{0,1\}^m. \end{aligned}$$

DISJUNCTION: One of $\mathbf{a}^\top \mathbf{x} \geq b$ and $\mathbf{a}'^\top \mathbf{x} \geq b'$ needs to be satisfied.

$$\begin{aligned} \max \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{a}^\top \mathbf{x} \geq yb \\ & \mathbf{a}'^\top \mathbf{x} \geq (1-y)b' \\ & y \in \{0,1\}, \mathbf{x} \geq 0. \end{aligned}$$

MOREDISJUNCTION: At least k inequalities from $\mathbf{a}_i^\top \mathbf{x} \geq b_i$ need to be satisfied.

$$\begin{aligned} \max \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{a}_i^\top \mathbf{x} \geq y_i b_i \\ & \textstyle\sum_i y_i \geq k \\ & y_i \in \{0,1\}, \mathbf{x} \geq 0. \end{aligned}$$

EVENMOREDISJUNCTION: Assume $\mathbf{a}_i^\top \mathbf{x} \geq \gamma\ \forall \mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in C$, we want $\mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in C$ and at least k from $\mathbf{a}_i^\top \mathbf{x} \geq b_i$. Let $y_i = 1$ if constraint i is satisfied.

$$\begin{aligned} & \mathbf{a}_i^\top \mathbf{x} \geq y_i(b_i - \gamma) + \gamma \\ & \textstyle\sum_i y_i \geq k \\ & y_i \in \{0,1\}, \mathbf{x} \geq 0, \mathbf{x} \in C. \end{aligned}$$

NON-LINEARTOLINEAR: $\text{nnz}(\mathbf{x})$: no. of non-zero entries in $\mathbf{x}$.

$$\min_{\mathbf{x} \in [-M,M]^n} \ \text{nnz}(\mathbf{x}) \text{ s.t. } \mathbf{A}\mathbf{x} \leq \mathbf{b}.$$

$\Downarrow$ MILP

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \quad & \textstyle\sum_{i=1}^n z_i \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & -Mz_i \leq x_i \leq Mz_i, \forall i \\ & \mathbf{z} \in \{0,1\}^n, \mathbf{x} \in \mathbb{R}^n. \end{aligned}$$

JOBSCHEDULING: people $1, \cdots, n$; job $1, \cdots, n$; cost for person j doing job i: c_{ij};

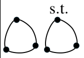
$$\begin{aligned} \min \quad & \textstyle\sum_j \sum_i c_{ij} x_{ij} \\ \text{s.t.} \quad & \textstyle\sum_j x_{ij} = 1, \forall i \\ & \textstyle\sum_i x_{ij} = 1, \forall j \\ & x_{ij} \in \{0,1\}, \forall i, j. \end{aligned}$$

k-COLORING: $G = (V, E)$; k-coloring: Each node has a color s.t. adjacent nodes have different color.

$$\begin{aligned} \min \quad & \textstyle\sum_j y_j \text{ (color } j \text{ needed?)} \\ \text{s.t.} \quad & \textstyle\sum_j x_{ij} = 1, \forall i \\ & x_{uj} + x_{vj} \leq y_j, \forall (u,v) \in E \\ & x_{ij}, y_j \in \{0,1\}. \end{aligned}$$

TSP (each city once + no sub-tour): Optimal route for driver to traverse n cities and return. Graph $G = (V, E), S \subseteq \{1, \cdots, n\}$; $\delta(S)$: subset of edges from S to S';

$$\begin{aligned} \min_{\{x_e\}} \quad & \textstyle\sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \textstyle\sum_{e \in \delta(i)} x_e = 2, \forall i \\ & \textstyle\sum_{e \in \delta(S)} x_e \geq 2, \\ & \forall \emptyset \subset S \subset V \\ & x_e \in \{0,1\}. \end{aligned}$$



ASYMMETRICTSP: Optimal route for driver to traverse n cities, but $c_{ij} \neq c_{ji}$. $\delta^+(S)$: S to S'; $\delta^-(S)$: S' to S;

$$\begin{aligned} \min \quad & \textstyle\sum_{a \in A} c_a x_a \\ \text{s.t.} \quad & \textstyle\sum_{a \in \delta^+(i)} x_a = 1, \forall i \\ & \textstyle\sum_{a \in \delta^-(i)} x_a = 1, \forall i \\ & \textstyle\sum_{a \in \delta^-(S)} x_a \geq 1, \\ & \forall 2 \leq |S| \leq |V| - 1 \\ & x_a \in \{0,1\}. \end{aligned}$$

2 LP and Lagrange Duality

Convexity:

- Convex set: If $\mathbf{x}, \mathbf{x}' \in D$, then $\lambda \mathbf{x} + (1-\lambda)\mathbf{x}' \in D$ for all $\lambda \in [0,1]$.
- Convex function: $f(\lambda \mathbf{x} + (1-\lambda)\mathbf{x}') \leq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{x}')$.
  - ▷ Any local minimum is also a global minimum.
  - ▷ If f differentiable, then convex iff $f(\mathbf{x}') \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top(\mathbf{x}' - \mathbf{x})$ for all $\mathbf{x}, \mathbf{x}'$.
  - ▷ If f twice differentiable, then convex iff $\nabla^2(\mathbf{x}) \succeq \mathbf{0}$ for all $\mathbf{x}$.
  - ▷ If f_1, f_2 convex, $\alpha_1, \alpha_2 > 0$, then $\alpha_1 f_1 + \alpha_2 f_2$ convex.
  - ▷ If $f_1, \cdots, f_L$ convex, then $\max_{\ell \in [L]} f_\ell$ convex.
  - ▷ If h linear/affine and g convex, then $g \circ h$ convex.
  - ▷ Jensen's inequality: For any random vector $\mathbf{X}$ and convex function f, $f(\mathbb{E}[\mathbf{X}]) \leq \mathbb{E}[f(\mathbf{X})]$.

Convex optimization: (1) f_0 and all f_i are convex; (2) all h_i affine.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f_0(\mathbf{x}) \\ \text{s.t.} \quad & f_i(\mathbf{x}) \leq 0, \quad \forall i = 1, \cdots, m_{\text{ineq}} \\ & h_i(\mathbf{x}) = 0, \quad \forall i = 1, \cdots, m_{\text{eq}}. \end{aligned}$$

Lagrangian: $L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f_0(\mathbf{x}) + \sum_{i \in [m_{\text{ineq}}]} \lambda_i f_i(\mathbf{x}) + \sum_{i \in [m_{\text{eq}}]} \nu_i h_i(\mathbf{x})$.

- $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ are *Lagrangian multipliers*.
- Lagrangian dual: $g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \min_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$.
- Lagrangian dual problem: $\max_{\boldsymbol{\lambda}, \boldsymbol{\mu}} g(\boldsymbol{\lambda}, \boldsymbol{\nu})$ s.t. $\boldsymbol{\lambda} \geq \mathbf{0}$.
- Weak duality: $g(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*) \leq f_0(\mathbf{x}^*)$.
- Strong duality: If original problem is convex and a mild regularity condition holds, then $g(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*) = f_0(\mathbf{x}^*)$.
  - ▷ Slater's condition: There exists at least one feasible $\mathbf{x}$ s.t. all $f_i(\mathbf{x}) < 0$ and all $_i(\mathbf{x}) = 0$.
  - ▷ Another sufficient condition: All f_i are linear.

Lagrangian of LP:

$$(\mathbf{P}) \ \min_{\mathbf{x}} \ \mathbf{c}^\top \mathbf{x} \ \text{ s.t. } \ \mathbf{A}\mathbf{x} = \mathbf{b}; \mathbf{x} \geq \mathbf{0}. \quad \Leftrightarrow \quad (\mathbf{D}) \ \max_{\boldsymbol{\nu}} \ \mathbf{b}^\top \boldsymbol{\nu} \ \text{ s.t. } \ \mathbf{A}^\top \boldsymbol{\nu} \leq \mathbf{c}.$$

- If we replace by $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, then add constraint $\boldsymbol{\nu} \geq \mathbf{0}$.
- Strong duality: $\min(\mathbf{P}) = \max(\mathbf{D})$.

Examples of convex optimization formulation:

MAXFLOW: Directed graph $G = (V, E)$; source s, sink t;

$$\begin{aligned} \max_{\{f_{uv}\}} \quad & \textstyle\sum_{v:(s,v) \in E} f_{sv} \\ \text{s.t.} \quad & 0 \leq f_{uv} \leq c_{uv} \\ & \textstyle\sum_{u:(u,v) \in E} f_{uv} = \sum_{v:(v,w) \in E} f_{vw}, \\ & \forall v \in V \setminus \{s, t\}. \end{aligned}$$

MATCHING:

$$\begin{aligned} \max \quad & \textstyle\sum_{i,j} x_{ij} w_{ij} \\ \text{s.t.} \quad & \textstyle\sum_j x_{ij} = 1, \forall i \\ & \textstyle\sum_i x_{ij} = 1, \forall j \\ & x_{ij} \in \{0,1\}. \end{aligned}$$

- Dual of MAXFLOW:

$$\begin{aligned} \min_{\boldsymbol{\lambda}, \boldsymbol{\mu}} \quad & \textstyle\sum_{(u,v) \in E} c_{uv} \lambda_{uv} \\ \text{s.t.} \quad & \mu_s = 1, \mu_t = 0 \\ & \lambda_{uv} \geq \mu_u - \mu_v, \lambda_{uv} \geq 0, \ \forall (u,v) \in E. \end{aligned}$$

  - ▷ Max flow = min cut.

3 LP Geometry

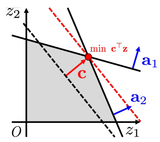
Standard form: Every LP can be converted to standard form (P).

- Maximization with $\mathbf{c}$ is minimization with $-\mathbf{c}$.
- $a_i x_i \leq b \Leftrightarrow a_i x_i + s_i = b; s_i \geq 0$.
- x unconstrained $\Leftrightarrow x = x^+ - x^-; x^+, x^- \geq 0$.

Polyhedron: $\{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$.

- Convex sets defined by linear inequalities (equalities).
- Polytype: Bounded polyhedron.
- Extreme point: Let $S \subseteq \mathbb{R}^n$ be a set (polyhedron or otherwise). We say that $\mathbf{y} \in S$ is an *extreme point* of S if, whenever $\mathbf{y} = \theta \mathbf{z}_1 + (1-\theta)\mathbf{z}_2$ for some $0 < \theta < 1$ and $\mathbf{z}_1, \mathbf{z}_2 \in S$, it must hold that $\mathbf{y} = \mathbf{z}_1 = \mathbf{z}_2$.
- Vertex: Let $P \subseteq \mathbb{R}^n$ be a polyhedron. We say that $\mathbf{y} \in P$ is a *vertex* of P if there is a direction $\mathbf{c} \in \mathbb{R}^n$ such that $\mathbf{c}^\top \mathbf{y} < \mathbf{c}^\top \mathbf{z}$ for all $\mathbf{z} \in P \setminus \{\mathbf{y}\}$.

Basic feasible solutions:



- Assumptions ($\mathbf{A} \in \mathbb{R}^{m \times n}$):
  1. The polyhedron P is not empty.
  2. The linear map $\mathbf{A}$ has full row rank.
- Basis: m linearly dependent columns of $\mathbf{A}$.
- Basic solution: Let $\mathbf{B}$ be a basis. Then $\mathbf{x} =$
  - ▷ $\mathbf{B}^{-1}\mathbf{b}$ for the columns in basis
  - ▷ $\mathbf{0}$ for the columns not in basis

  is a basic solution.
- Basic feasible solution: Basic solution that $\geq \mathbf{0}$ (feasible).

Thm. 3.1: Suppose $P = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$ is a non-empty polyhedron, and let $\mathbf{x} \in P$. The following are equivalent:

- $\mathbf{x}$ is a vertex;
- $\mathbf{x}$ is an extreme point;
- $\mathbf{x}$ is a basic feasible solution with non-negative entries.

4 U and TU

Polyhedron integrality: A polyhedron $P \subseteq \mathbb{R}^n$ is *integral* if all its extreme points are integer vectors.

- If so, then solving the relaxation of ILP is **tight**.

Unimodularity (U): A square matrix $\mathbf{A} \in \mathbb{Z}^{m \times m}$ is *unimodular* if its determinant if its determinant is ± 1. A matrix $\mathbf{A} \in \mathbb{Z}^{m \times n}$ with **full row rank** is *unimodular* if the sub-matrix obtained by taking any m columns of A is either singular or unimodular (i.e., $\in \{-1, 0, 1\}$).

Thm. 4.1: Let $\mathbf{A} \in \mathbb{Z}^{m \times n}$ be a matrix with **full row rank**. Then $\mathbf{A}$ is unimodular if and only if the set $P(\mathbf{b}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$ is integral for any $\mathbf{b} \in \mathbb{Z}^m$ for which the polyhedron $P(\mathbf{b})$ is non-empty.

Proof. This can be proven by using Cramer's Rule.

- This applies only to **standard form**.
- Prop. 4.2: A non-empty bounded polyhedron has at least one extreme point.
- Prop. 4.3: Let P be a non-empty polyhedron with at least one extreme point. The optimal solution of the LP $\{\min \mathbf{c}^\top \mathbf{x} : \mathbf{x} \in P\}$ is either $-\infty$ or is attained (possibly non-uniquely) at an extreme point of P.
- Prop. 4.4: Any non-empty polyhedron of the form $\{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$ has at least one basic feasible solution (and hence an extreme point).

Other results: (from tutorial)

- $\mathbf{U}$ is a square invertible matrix. $\mathbf{U}$ is unimodular if and only if $\mathbf{U}$ and $\mathbf{U}^{-1}$ are both integer-valued matrices (proven by Cramer's rule).
- $\mathbf{U}$ is a square invertible matrix. $\mathbf{U}$ is unimodular if and only if for all $\mathbf{x}$, $\mathbf{U}\mathbf{x}$ is integral if and only if $\mathbf{x}$ is integral.
- **Unimodular operations:**
  1. Switch two columns;
  2. Multiply a column by -1;
  3. Add an integer multiple of a column to another;

  Let $\mathbf{U}$ be a square invertible matrix. $\mathbf{U}$ is unimodular if (and only if) it can be derived from the identity matrix via above operations.

Total unimodularity (TU): We say that a matrix $\mathbf{A} \in \mathbb{Z}^{m \times n}$ is *totally unimodular* if the determinant of each square sub-matrix of A is in $\{-1, 0, 1\}$.

Thm. 4.6: Let $\mathbf{A} \in \mathbb{Z}^{m \times n}$ be a matrix. Then $\mathbf{A}$ is totally unimodular if and only if the set $P(\mathbf{b}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} \leq \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$ is integral for any $\mathbf{b} \in \mathbb{Z}^m$ for which the polyhedron $P(\mathbf{b})$ is non-empty.

Proof. $\mathbf{A}$ is TU
$\Leftrightarrow [\mathbf{A}\ \mathbf{I}]$ is U (Prop. 4.7)
$\Leftrightarrow \{(\mathbf{x}, \mathbf{s}) : \mathbf{A}\mathbf{x} + \mathbf{s} = \mathbf{b}; \mathbf{x}, \mathbf{s} \geq \mathbf{0}\}$ is integral (Thm. 4.1)
$\Leftrightarrow \{\mathbf{x} : \mathbf{A}\mathbf{x} \leq \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$ is integral (Prop. 4.8).

- Prop. 4.7: $\mathbf{A}$ is TU if and only if $[\mathbf{A}\ \mathbf{I}_{m \times m}]$ is unimodular.
- Prop. 4.8: $\mathbf{x}^*$ is an extreme point of $P(\mathbf{b}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} \leq \mathbf{b}; \mathbf{x} \geq \mathbf{0}\}$ if and only if $[\mathbf{x}^*\ \mathbf{s}^*] := [\mathbf{x}^*\ \mathbf{b} - \mathbf{A}\mathbf{x}^*]$ is an extreme point of $Q(\mathbf{b}) = \{(\mathbf{x}, \mathbf{s}) : \mathbf{A}\mathbf{x} + \mathbf{s} = \mathbf{b}; \mathbf{x}, \mathbf{s} \geq \mathbf{0}\}$.

Prop. 4.9: $\mathbf{A} \in \mathbb{Z}^{m \times n}$. Then $\mathbf{A}$ is TU if and only if $\mathbf{A}^\top$ is TU.

Prop. 4.10: $\mathbf{A} \in \mathbb{Z}^{m \times n}$. Then $[\mathbf{A}\ \mathbf{I}], [\mathbf{A}\ \mathbf{A}], [\mathbf{A}\ -\mathbf{A}], [\mathbf{A}\ -\mathbf{A}\ \mathbf{I}], [\mathbf{A}\ -\mathbf{A}\ \mathbf{I}\ -\mathbf{I}]$ are all TU.

Thm. 4.11: Let $\mathbf{A} \in \mathbb{Z}^{m \times n}$ be a matrix. Then $\mathbf{A}$ is TU if and only if the set $\{\mathbf{x} : \mathbf{a} \leq \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$ is integral for all integral vectors $\mathbf{a}, \mathbf{b}, \mathbf{l}, \mathbf{u}$ for which the polyhedron is non-empty.

Sufficient conditions for TU:

- Thm. 4.12: A matrix $\mathbf{A} \in \{-1, 0, 1\}^{m \times n}$ is TU if both of the following conditions hold:
  1. Each column of A contains at most 2 non-zero entries;
  2. It is possible to split the row indices $\{1, \cdots, m\}$ into two disjoint sets I_1, I_2 such that whenever a column (indexed by j) has 2 non-zero entries, it holds that $$\sum_{i \in I_1} A_{ij} = \sum_{i \in I_2} A_{ij}.$$

  In other words, if the two non-zeros have the same sign then one lies in I_1 and one lies in I_2, whereas if the two non-zeros have different signs then both lie in I_1 or both lie in I_2.
  - ▷ Cor. 4.13: A matrix $\mathbf{A} \in \{-1, 0, 1\}^{m \times n}$ is TU if each of its columns has at most one $+1$ entry and at most one -1 entry.
- Thm. 4.14: Let $\mathbf{A}$ be an $\times n$ integer-valued matrix, and for any $\mathcal{J} \subseteq \{1, \cdots, n\}$, let $\mathbf{A}_{\mathcal{J}}$ denote the $m \times \mathcal{J}$ sub-matrix obtained by keeping only the columns in $\mathcal{J}$. Then $\mathbf{A}$ is TU if and only if $\mathbf{A}_{\mathcal{J}}$ admits an equitable column-bicoloring for all non-empty $\mathcal{J} \subseteq \{1, \cdots, n\}$.
  - ▷ Equitable bicoloring: We say that an integer-valued matrix $\mathbf{A}$ admits an *equitable column-bicoloring* if it is possible to partition the columns indices $\mathcal{J}$ into two sets $\mathcal{J}_a$ and $\mathcal{J}_b$ so that the difference between the sums of the columns in these subsets is a vector with entries in $\{-1, 0, 1\}$: $$\sum_{i \in \mathcal{J}_a} A_i - \sum_{j \in \mathcal{J}_b} A_j \in \{-1, 0, 1\}^m,$$ where A_i is the i-th column of $\mathbf{A}$. Equivalently, there exists some $\mathbf{z} \in \{-1, +1\}^{|\mathcal{J}|}$ such that each entry of $\mathbf{A}\mathbf{z}$ has absolute value at most one.
  - ▷ Cor. 4.15: $\mathbf{A}$ is TU if and only if every sub-matrix obtained by taking a non-empty subset of the rows of $\mathbf{A}$ admits an equitable row-bicoloring.
- Prop. 4.16: The node-edge incidence matrix of an undirected bipartite graph is TU.
  - ▷ $A_{ij} = 1$ if node i is in edge j and 0 otherwise.
  - ▷ Graph matching: Given a bipartite graph, a *matching* is a subset of non-intersecting edges (i.e., edges for which no two of them have a common node). A matching is said to be *perfect* if all nodes are selected.
- Prop. 4.17: The node-edge incidence matrix of a directed graph is TU.
  - ▷ $A_{ij} = 1$ if edge j starts from node i, -1 if edge j ends at node i, 0 otherwise.

5 Rounding

General approach: Rounding the LP solution to get α-approximation $\hat{\mathbf{c}}^\top \mathbf{x} \leq \text{RR} \cdot \text{OPT(ILP)}$ s.t. $\alpha = \text{RR}$.

- α-approximation: $\hat{\mathbf{x}}$ is an α-approximation if it is a feasible integer solution s.t. $\mathbf{c}^\top \mathbf{x} \leq \alpha \cdot \text{OPT(ILP)}$.
- Integrality gap: $\text{IG} = \text{OPT(ILP)}/\text{OPT(LP)} \geq 1$.
- Rounding ratio: $\text{RR} = \mathbf{c}^\top \mathbf{x}/\text{OPT(LP)} \geq 1$.

WEIGHTEDVERTEXCOVER problem (2-approximation): Given undirected $G = (V, E)$ with non-negative vertex weights $w : V \to \mathbb{R}$, find the vertex cover with minimum total weight.

$$\begin{aligned} \min \quad & \textstyle\sum_{v \in V} w_v x_v \quad (x_v\text{: whether } v \text{ is selected}) \\ \text{s.t.} \quad & x_u + x_v \geq 1, \forall (u,v) \in E \\ & x_v \in \{0,1\}, \forall v \in V \xrightarrow{\text{relax}} x_v \in [0,1]. \end{aligned}$$

- Rounding: If $x_v^{\text{LP}} \geq 1/2$, set $x_v = 1$; else set $x_v = 0$.
- Feasibility: At least one of $x_u, x_v \geq 1/2$, so after rounding $x_u + x_v \geq 1$ is guaranteed.
- Approximation: $\forall v,\ x_v \leq 2x_v^{\text{LP}}$, so $\sum w_v x_v \leq 2\text{OPT(LP)} \leq 2\text{OPT(ILP)}$.

Randomized rounding: Suppose LP returns values $x_i \in [0,1]$,

(†) For each x_i, round to 1 w.p. x_i; OR
(‡) Sample a $\lambda \sim \text{U}[0,1]$, for each x_i round to 1 if $x \geq \lambda$.

SETCOVERING problem: Let $E = [n]$ be an index set, $\mathcal{F} = \{F_1, \cdots, F_m\}$ be a collection of subsets of E. Find the set covering with minimum total cost.

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \quad (x_i\text{: whether } F_i \text{ is selected}) \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{x} \geq \mathbf{1} \quad (A_{ij}\text{: whether } j \text{ is in } F_i) \\ & \mathbf{x} \in \{0,1\}^m \xrightarrow{\text{relax}} \mathbf{x} \in [0,1]^m. \end{aligned}$$

- Rounding: Use (†) until feasible.
- Feasiblity: In one (†), $\Pr[e \text{ not picked}] = \prod_{i:e \in F_i}(1 - x_i^{\text{LP}}) \leq \exp(-\sum_{i:e \in F_i} x_i^{\text{LP}}) \leq 1/e$. Setting $\log n + 2$ times of (†), probability of all of them failing is at most $(1/e)^{\log n + 2} = e^{-2}/n$. Since there are n elements, we get coverage w.p. at least $1 - e^{-2}$.
- Approximation: Assume (†) is repeated for $\log n + 2$ times. We know $\mathbb{E}[\mathbf{c}^\top \mathbf{x}] = \mathbf{c}^\top \mathbf{x}^{\text{LP}}$. So $\mathbb{E}[\mathbf{c}^\top \mathbf{x}_{\text{final}}] \leq (\log n + 2)\text{OPT(ILP)}$. W.p. at least 0.9, $\mathbf{c}^\top \mathbf{x}_{\text{final}} \leq 10(\log n + 2)\text{OPT(ILP)}$.

TSP is hard to approximate: We show that efficient approximation of TSP will solve HAMILTONIANCYCLE problem (NP-hard) efficiently:

1. Let $G = (V, E)$ be a weighted graph in which every edge is included, $w(e) = 1$ if $e \in E_0$ and $w(e) = n\alpha + 1$ if $e \notin E_0$.
2. If a Hamiltonian cycle exists in G_0, then TSP approximated solution is at most $n\alpha$. Otherwise, it is at least $n\alpha + 1$.
3. So can solve HAMILTONIANCYCLE using TSP approximation.

6 Submodularity

Submodularity: $\forall S \subseteq T \subseteq V,\ e \in V \setminus T,\ \Delta(e|S) \geq \Delta(e|T)$.

- Related notions:
  - ▷ Monotonicity: $S \subseteq T \subseteq V \Rightarrow f(S) \leq f(T)$.

- ▷ Modularity: $\Delta(e|S) = \Delta(e|T)$.
- ▷ Supermodularity: $\Delta(e|S) \le \Delta(e|T)$.
- Equivalent definitions:
  - ▷ $\forall S, T,\ f(S) + f(T) \ge f(S \cup T) + f(S \cap T)$.
  - ▷ $\forall S, e, e',\ \Delta(e|S) \ge \Delta(e|S \cup \{e'\})$.
  - ▷ (If f monotone) $\forall S, T,\ f(T) \le f(S) + \sum_{e \in T \setminus S} \Delta(e|S)$.
- Relation to concavity:
  - ▷ Diminishing returns;
  - ▷ (Non-monotone case) Any local maximum is within 1/2 of global maximum.
  - ▷ Maximization (unconstrained or constrained) can be done approximately efficiently.
  - ▷ $f(S) = g(|S|)$ is submodular if g is concave.
- Relation to convexity:
  - ▷ Unconstrained minimization can be done exactly efficiently.
  - ▷ An extension from sets to continuous values called *Lovász extension* is a convex function.
- Properties: Suppose f_1, f_2 are submodular:
  - ▷ Linear combinations: $c_1, c_2 > 0 \Rightarrow c_1 f_1 + c_2 f_2$ submodular.
  - ▷ Concave of modular: g modular, h concave $\Rightarrow h \circ g$ submodular.
  - ▷ Residual: $f(S) = f_1(S \cup B) - f_1(B)$ submodular for any B.
  - ▷ Conditioning: $f(S) = f_1(S \cap A)$ submodular for any A.
  - ▷ Reflection: $f(S) = f_1(V \setminus S)$ submodular.
  - ▷ Truncation: If f_1 also monotone, then $f(S) = \min\{c, f_1(S)\}$ is submodular for any c.
  - ▷ Minimum: $\min\{f_1, f_2\}$ is submodular if either $f_1 - f_2$ or $f_2 - f_1$ is monotone.
- Examples:
  - ▷ $f(S) =$ area covered by activating all sensors in S.
  - ▷ Let $\mathbf{X}$ be a matrix, V be the set of column indices, $\mathbf{X}_S$ is the submatrix indexed by $S \subseteq V$. Then $r_S = \operatorname{rank}(\mathbf{X}_S)$ is monotone submodular.
  - ▷ $f(S) =$ total number of users influenced by advertising to S (in a graph).
  - ▷ $f(S) =$ representativeness of images in S.
  - ▷ $f(S) =$ number of edges between S and S^c is submodular but non-monotone.
  - ▷ $f(S) = H(\mathbf{X}_S)$ where *entropy* $H_X = \sum_x P_X(x) \log \frac{1}{P_X(x)}$ is monotone submodular.

Cardinality-constrained submodular maximization:

$$\begin{aligned} \max_{S \in \mathcal{S}} \quad & f(S) \\ \text{s.t.} \quad & \mathcal{S} = \{S : |S| \le k\}. \end{aligned}$$

- **Greedy algorithm**: For k times, add $e = \arg\max_{e \in V \setminus S_{i-1}} \Delta(e|S_{i-1})$.
- Useful fact: $1 - x \le e^{-x}, \forall x \in \mathbb{R}$.
- Approximation: If f monotone submodular with $f(\emptyset) = 0$, then $f(S_k) \ge (1 - 1/e) f(S_k^*)$.
- Generalization: If we perform ℓ instead of k iterations, then $f(S_\ell) \ge (1 - e^{-\ell/k}) f(S_k^*)$.

Proof. $f(S^*) \le f(S^* \cup S_i)$ (monotonicity)

$$\begin{aligned} &= f(S_i) + \textstyle\sum_{j=1}^k \Delta(e_j^*|S_i \cup \{e_1^*, \cdots, e_{j-1}^*\}) \\ &\le f(S_i) + \textstyle\sum_{j=1}^k \Delta(e_j^*|S_i) \quad \text{(submodularity)} \\ &\le f(S_i) + \textstyle\sum_{j=1}^k \Delta(e_{i+1}^*|S_i) \quad \text{(greedy)} \\ &\le f(S_i) + k(f(S_{i+1}) - f(S_i)). \end{aligned}$$

So $f(S^*) - f(S_{i+1}) \le (1 - 1/k)(f(S^*) - f(S_i))$. Since $(1 - 1/k)^\ell \le e^{-\ell/k}$, we have proven the theorem.

- **Lazy greedy algorithm**: For each e maintain its upper bound $(e, \rho(e))$ and sort. If at a round the marginal contribution of e is still larger than $\rho(e')$ after it, then we choose e without considering other elements.
- **Stochastic greedy algorithm**: Sample only N elements from $V \setminus S_{i-1}$ and choose the one with largest gain.
  - ▷ Choosing $N = (n/k)\log(1/\varepsilon)$ ensures overlapping with S^* w.p. at least $1 - \varepsilon$.
  - ▷ Time: $O(n \log(1/\varepsilon))$.
  - ▷ Approximation: $(1 - 1/e - \varepsilon)$ on expectation.

7 Matroids

Independence system: Let N be a finite set, and let $\mathcal{I}$ be a collection of subsets of N. We say that the tuple $(N, \mathcal{I})$ is an *independence system* if

1. $\emptyset \in \mathcal{I}$; AND
2. Hereditary: $A \in \mathcal{I}$ implies $B \in \mathcal{I}$ for all $B \subseteq A$.

- Each $S \in \mathcal{I}$ is an *independent set*. N is also called *ground set*.
- $S \notin \mathcal{I}$ is *dependent*.
- For $T \subseteq N$, an independent set $S \in \mathcal{I}$ satisfying $S \subseteq T$ is *maximal with respect to T* if $S \cup \{i\}$ is dependent for all $i \in T \setminus S$.
- Any maximally independent subset of T is a *basis* of T.
- Rank $r(T)$: Cardinality of largest sized basis of T. $r(\cdot)$ is called the *rank function*.

Matroid:

1. An independence system $(N, \mathcal{I})$ is a *matroid* if for any two independent sets $A \in \mathcal{I}$ and $B \in \mathcal{I}$, if B contains more elements than A, then there exists $x \in B \setminus A$. $A \cup \{x\} \in \mathcal{I}$.
2. An independence system $(N, \mathcal{I})$ is a *matroid* if for all subsets $T \subseteq N$, every maximal independent set (basis) of T has cardinality $r(T)$.

- Thm. 7.1: An independence system $(N, \mathcal{I})$ is a matroid iff its rank function $r(\cdot)$ is submodular.

8 Exact Solutions via Greedy Algorithms

BestIndependentSet problem: Given a matroid $(N, \mathcal{I})$, find the independent set with the highest weight.

$$\begin{aligned} \max_x \quad & \textstyle\sum_{j \in N} c_j x_j \\ \text{s.t.} \quad & \textstyle\sum_{j \in T} x_j \le r(T),\ \forall T \subseteq N \\ & x_j \in \{0, 1\},\ \forall j \in N \xrightarrow{\text{relax}} x_j \ge 0. \end{aligned}$$

- r is non-negative, non-decreasing, submodular and $r(\emptyset) = 0$.
- **Greedy algorithm**:
  1. Re-label the set s.t. all weights are sorted in decreasing order $c_1 \ge \cdots \ge c_k > 0 \ge c_{k+1} \ge \cdots \ge c_n$.
  2. Define the set $\mathcal{S}^j = \{1, \cdots, j\}$ and $\mathcal{S}^0 = \emptyset$.
  3. Pick (♦) $x_j = r(\mathcal{S}^j) - r(\mathcal{S}^{j-1})$ if $1 \le j \le k$ and 0 if $j > k$.
  - ▷ Greedy since we choose the largest x_j not violating $x_1 + \cdots + x_j \le f(\mathcal{S}^j)$.

- Dual problem:

$$\begin{aligned} \min_{\{y_T\}_{T \subseteq N}} \quad & \textstyle\sum_{T \subseteq N} r(T) y_T \\ \text{s.t.} \quad & \textstyle\sum_{T: j \in T} y_T \ge c_j,\ \forall j \in N \\ & y_T \ge 0,\ \forall T \subseteq N. \end{aligned}$$

  - ▷ Optimal solution: (◊) $y_S = \begin{cases} c_j - c_{j+1} & \text{if } S = \mathcal{S}^j, 1 \le j \le k \\ c_k & \text{if } S = \mathcal{S}^j \\ 0 & \text{otherwise} \end{cases}$

- Prop. 8.1: If r is a submodular, non-decreasing function satisfying $r(\emptyset) = 0$, then (♦) and (◊) are the primal and dual optimal solutions to the primal and dual problem respectively.

Proof. Primal feasibility: For a given set T,

$$\begin{aligned} \textstyle\sum_{j \in T} x_j &= \textstyle\sum_{j: j \in T; j \le k} (f(\mathcal{S}^j) - f(\mathcal{S}^{j-1})) \\ &\le \textstyle\sum_{j: j \in T; j \le k} (f(\mathcal{S}^j \cap T) - f(\mathcal{S}^{j-1} \cap T)) \text{(submodularity)} \\ &\le \textstyle\sum_{j: j \le k} (f(\mathcal{S}^j \cap T) - f(\mathcal{S}^{j-1} \cap T)) \\ &= f(\mathcal{S}^k \cap T) - f(\emptyset) \\ &= f(\mathcal{S}^k \cap T) \\ &\le f(T) \text{(non-decreasing)}. \end{aligned}$$

The non-negativity constraint is obvious as f is non-decreasing.
Dual feasibility: For $j \le k$, we have

$$\begin{aligned} \textstyle\sum_{T: j \in T} y_T &= y_{\mathcal{S}^j} + \cdots + y_{\mathcal{S}^k} \\ &= (c_j - c_{j+1}) + \cdots + (c_{k-1} - c_k) + c_k \\ &= c_j. \end{aligned}$$

For $j > k$, we have $\sum_{T: j \in T} y_T = 0 + \cdots + 0 \ge c_j$.
Since both objectives are equal, by strong duality, we know that both solutions are optimal.

- If r is integer-valued and we add the integrality constraint $x_j \in \mathbb{Z}$, then greedy algorithm gives integer solution which is still optimal.

MinimumSpanningTree problem: Given undirected $G = (V, E)$, find the spanning tree of minimum weight.

- **Greedy algorithm**: Starting from the empty graph, repeatedly add the smallest weight edge that does not form a cycle. Stop once there are $|V| - 1$ edges.
- Equivalence to BestIndependentSet: Set $c_{\max} - c_{ij}$ in MinimumSpanningTree as the weight c_j in BestIndependentSet; sort them in increasing order.

JobScheduling problem: Each job $1, \cdots, n$ takes a unit amount of time and has a deadline $d_{1 \dots n}$. Determine the optimal order to maximize the total reward.

- **Greedy algorithm**:
  1. List of jobs $\mathcal{J} \leftarrow \emptyset$.
  2. Sort the job from highest reward to lowest: $j_1, \cdots, j_n$.
  3. Starting from j_1, if adding j to $\mathcal{J}$ (just before its deadline) is still feasible, then do.
- Prop. 8.2: Given a set S of jobs, let $N_t(S)$ be the number of jobs whose deadline is t or earlier. The following are equivalent:
  1. There exists a schedule that completes all jobs in S on time;
  2. $N_t(S) \le t$ for all $t = 1, \cdots, n$.
  3. If the jobs in S are run sequentially in monotonically increasing order of deadline, then no job is late.
  - ▷ This implies that the set of all feasible job subsets forms a matroid.

9 Computational Complexity

Problem: A *problem* is specified by a set of inputs, each of which has an associated correct output.

Algorithm: An *algorithm* is a computer program that is guaranteed to produce the correct output for a given problem.

Order of growth: Let $f(n)$ and $g(n)$ be real functions, then

1. $f(n) = O(g(n))$ if $\exists c$ s.t. $f(n) \le cg(n)$ when n is large enough.
2. $f(n) = \Omega(g(n))$ if $\exists c$ s.t. $f(n) \ge cg(n)$ when n is large enough.
3. $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.
4. $f(n) = o(g(n))$ if $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$.

- Stirling's approximation: $\log(n!) = n \log n - n + O(\log n) = O(n \log n)$.
- Polynomial time: $O(n^k)$ for some constant k.

Difficulty of problem:

- Decision problem: One that has a binary answer, YES or NO.
- Class $\mathcal{P}$: A decision problem is in $\mathcal{P}$ if it is solvable by an algorithm whose runtime in polynomial with respect to the number of bits used to specify the problem input.
  - ▷ The decision counterpart of LP is in $\mathcal{P}$.
- Class $\mathcal{NP}$: A decision problem is in $\mathcal{NP}$ if there exists a *certifying procedure* s.t. the following conditions are true:
  1. Any YES instance of the problem has an associated *certificate* whose size is polynomial with respect to the original input.
  2. Given the original input and the certificate, the certifying procedure is able to confirm with certainty that the answer is YES in polynomial time.

Thm. 9.1: $\mathcal{P} \subseteq \mathcal{NP}$.

Reduction: We say that ρ_1 reduces to ρ_2 if it is possible to solve ρ_1 by solving at most a polynomial number of instances of ρ_2 (each with polynomial input size), plus polynomial-time additional computation.

- If algorithm for ρ_2 is correct, we can solve ρ_1.
- NP-hard: ρ_0 is NP-hard if all problems in $\mathcal{NP}$ reduce to ρ_0.
- NP-complete: ρ_0 is NP-complete if ρ_0 is NP-hard and $\rho_0 \in \mathcal{NP}$.
- Prop. 9.2: If ρ_0 is NP-complete/hard and we can reduce it to some other problem $\rho_0' \in / \notin \mathcal{NP}$, then ρ_0' is NP-complete/hard.
- BooleanSatisfiability problem (or more specifically 3-SAT) is NP-complete: given n Boolean variables $x_{1 \dots n}$ and m disjunctive logical clauses $c_{1 \dots m}$, decide whether it is possible to assign $x_{1 \dots n}$ s.t. all clauses are TRUE.
- Reduce 3-SAT to SubsetSum:

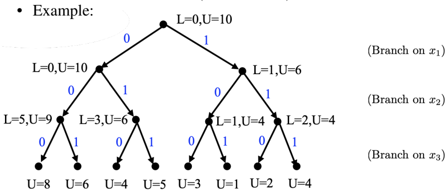
	x_1	x_2	x_3	c_1	c_2
y_1	1	0	0	1	0
z_1	1	0	0	0	1
y_2	0	1	0	1	1
z_2	0	1	0	0	0
y_3	0	0	1	0	0
z_3	0	0	1	1	1
s_1	0	0	0	1	0
t_1	0	0	0	1	0
s_2	0	0	0	0	1
t_2	0	0	0	0	1
k	1	1	1	3	3

10 General Global Optimization

Branch and bound: Consider the problem $\min_x \mathbf{c}^\top \mathbf{x}$ s.t. $x \in \mathcal{F}$,

1. Select a sub-problem $\mathcal{F}_i$ that is active (i.e., not eliminated);
2. If $\mathcal{F}_i$ is empty, eliminate $\mathcal{F}_i$; otherwise, compute $\ell(\mathcal{F}_i)$ (e.g., via LP relaxation);
3. If $\ell(\mathcal{F}) > U$, eliminate $\mathcal{F}_i$; otherwise, if we are able to solve $\mathcal{F}_i$ completely, we can possibly update the upper bound U and eliminate $\mathcal{F}_i$ from our list of sub-problems. Otherwise, break $\mathcal{F}_i$ into further sub-problems and add these problems to our list of sub-problems.
4. Return to Step 1 and continue until there are no active sub-problems. Return the point x^* that produced the latest updated (best) value of U.

- **Best-first-search**: The next node to be branched is the one that has smallest lower bound (for minimize).
- Example:



  - ▷ Suppose we search 1 before 0.
  - ▷ BFS: $1 \to 0 \to 11 \to 10 \to 01 \to 00 \to 111 \to 110 \to 101$.
  - ▷ DFS: $111 \to 110 \to 101 \to 100 \to 0 \to 01 \to 00$.
  - ▷ Best-first-search: $0 \to 1 \to 10 \to 101$.
- Limitation: Certain non-binary ILP may have infinite number of sub-problems.

Cutting plane: Consider ILP in standard form and its relaxation:

$$\begin{aligned} \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \ge \mathbf{0}; x \in \mathbb{Z}^n \xrightarrow{\text{relax}} \text{delete}. \end{aligned}$$

Suppose we obtain an LP solution $\hat{\mathbf{x}}^{\text{LP}}$ that is non-integral. We then seek an inequality that all ILP solutions satisfy, but excludes $\hat{\mathbf{x}}^{\text{LP}}$, $\boldsymbol{\alpha}^\top \mathbf{x} \le \beta$ (*valid inequality*) s.t. $\boldsymbol{\alpha}^\top \hat{\mathbf{x}}^{\text{LP}} > \beta$ (*cutting plane*). If both conditions are satisfied, the inequality is a *valid cut*.

- **Gomory cuts**: Suppose the optimal basis is formed by the first m columns denoted as $\mathbf{B}$: $\mathbf{A} = [\mathbf{B} \quad \mathbf{A}_{S^c}]$. $\mathbf{x}^{\text{LP}}$ has the form $\begin{bmatrix} \mathbf{B}^{-1}\mathbf{b} \\ 0 \end{bmatrix}$. Hence $\begin{bmatrix} \mathbf{I} \\ \bar{\mathbf{A}} = \mathbf{B}^{-1}\mathbf{A}_{S^c} \end{bmatrix} \mathbf{x} = \mathbf{B}^{-1}\mathbf{b} = \bar{\mathbf{b}}$. For each row $h \in S$, $x_h + \sum_{j \in S^c} \bar{A}_{hj} x_j = \bar{b}_h$.
  - ▷ Prop. 10.1: For any h s.t. $\bar{b}_h$ is not integral, the following is a cutting plane:
  $$x_h + \sum_{j \in S^c} \lfloor \bar{A}_{hj} \rfloor x_j \le \lfloor \bar{b}_h \rfloor,$$
  - ▷ Equivalent cut:
  $$\left(\sum_{j \in S^c} (\lfloor \bar{A}_{hj} \rfloor - \bar{A}_{hj}) x_j \right) + s = \lfloor \bar{b}_h \rfloor - \bar{b}_h, \quad s \ge 0.$$
  - ▷ Example: $x_1 + x_2 + \frac{4}{3}x_3 - \frac{1}{2}x_5 = \frac{4}{5}$, where x_3 and x_4 are non-basic variables. We can rewrite this as $x_1 + x_2 + x_3 - x_5 = -\frac{1}{3}x_3 - \frac{1}{2}x_5 + \frac{4}{5}$, which is at most $\frac{4}{5}$ and hence at most 0. So the Gomory cut is $-\frac{1}{3}x_3 - \frac{1}{2}x_5 + \frac{4}{5} \le 0$.

Dynamic programming:

- TSP problem: For $i = 1, \cdots, n$, evaluate $C(\mathcal{S}, k)$ for all subsets $\mathcal{S} \subset V$ with $|\mathcal{S}| = i$ and all $k \in V$.
  - ▷ Recurrence: $C(\mathcal{S}, k) = \min_{m \in \mathcal{S} \setminus \{k\}} \{C(\mathcal{S} \setminus \{k\}, m) + c_{mk}\}$ refers to the min cost of paths starting from 1 ending at k visiting all nodes in $\mathcal{S}$.
  - ▷ Time: $O(n^2 2^n)$.
- Knapsack problem: Iteratively build up the table of $W_i(C)$ values starting from $i = C = 0$. At each step, proceed to any non-computed entry for which the quantities needed have been computed. Continue until the entire table has been completed.
  - ▷ Recurrence: $W_{i+1}(C) = \min\{W_i(C), W_i(C - c_{i+1}) + w_{i+1}\}$ refers to the least weight accumulated to attain value at least C using items $1, \cdots, i+1$. Alternatively, we can use $C_{i+1}(W) = \max\{C_i(W), C_i(W - w_{i+1}) + c_{i+1}\}$ refering to the max value s.t. accumulated weight equals W.
  - ▷ Time: $O(n^2 c_{\max})$ (first); $O(nB)$ (second).

Appendix: Mathematical Facts

Determinant:

- **Cofactor expansion**: For a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, we have
$$\det(\mathbf{A}) = \textstyle\sum_{j=1}^n A_{ij} C_{ij},$$
$$\text{where } C_{ij} = (-1)^{i+j} \det(\mathbf{A}_{ij}).$$
- Cramer's rule: $\mathbf{A} \in \mathbb{R}^{n \times n}$ is invertible. Then the solution to the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is given by
$$x_i = \frac{\det(\mathbf{A}_i)}{\det(\mathbf{A})},$$
where $\mathbf{A}_i$ is the matrix obtained by replacing the i-th column of $\mathbf{A}$ with vector $\mathbf{b}$:
$$[A_1 \quad \cdots \quad A_{i-1} \quad \mathbf{b} \quad A_{i+1} \quad \cdots \quad A_n].$$
- $\det(\mathbf{A}^\top) = \det(\mathbf{A})$.
- $\det(\mathbf{A}^{-1}) = \frac{1}{\det(\mathbf{A})}$.
- $\det(\mathbf{A}\mathbf{B}) = \det(\mathbf{A})\det(\mathbf{B})$.

Trace:

- $\operatorname{tr}(\mathbf{A} + \mathbf{B}) = \operatorname{tr}(\mathbf{A}) + \operatorname{tr}(\mathbf{B})$.
- $\operatorname{tr}(\mathbf{A}^\top) = \operatorname{tr}(\mathbf{A})$.
- $\operatorname{tr}(\mathbf{A}^\top B) = \operatorname{tr}(\mathbf{A}\mathbf{B}^\top) = \operatorname{tr}(\mathbf{B}^\top \mathbf{A}) = \operatorname{tr}(\mathbf{B}\mathbf{A}^\top)$.