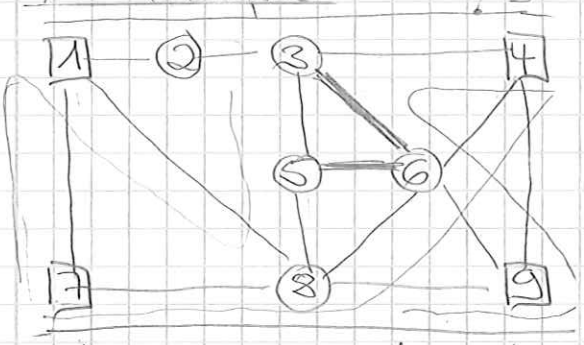
Design & operation of Traffic & Telecommunication Networks

1. Introduction

1.1 Example



$T = \{1, 4, 7, 9\}$ — terminal nodes

$S = \{2, 3, 5, 6, 8\}$ — Steiner nodes

$V = S \cup T$ — nodes

$E = \{12, 17, 18, \ldots, 89\}$ — edges

$G = (V, E)$ — graph (network)

$d_{19} = d_{29} = 1$ — demand

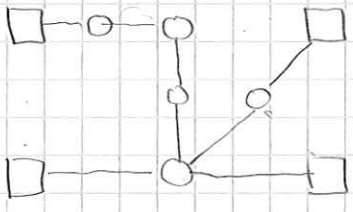
$u \equiv 1$ — capacities

$c_e = \begin{cases} 1, & e \neq 36, 56 \\ 2, & e = 36, 56 \end{cases}$ — costs

$c_b \equiv 1$

$\mathcal{P} = \{1234, 789, 7183, 784, 43698\}$

a) Connect V at min cost

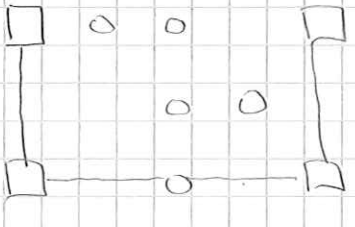


X_1

Min. spanning tree problem (MST) ✓

$c(X_1) = 9$

b) Connect T at min cost

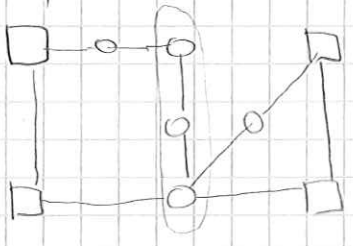


Y_1 ✓

Steiner tree problem (STP)

$c(Y_1) = 4$

c) Connect V at min cost s.t. 1 edge can fail



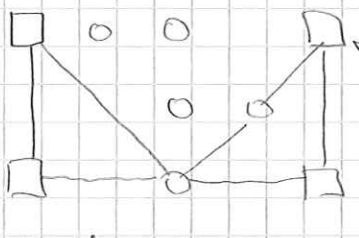
lower bound 10, HC 11

X_2

2-edge connected graph problem (2EC(V))

$c(X_2) = 11$

d) Connect T at min cost s.t. 1 edge can fail

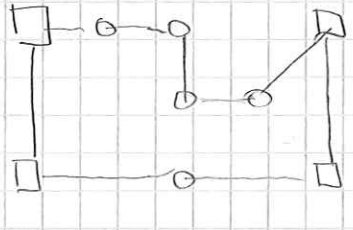


Y_2'

2-edge connected subgraph problem (2EC(T))

$c(Y_2') = 7$

e) Connect V at min cost s.t. 1 node can fail



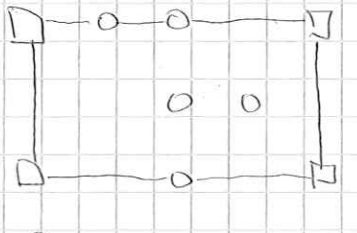
node fails ⇒ edge fails

X_3

2-node connected graph problem (2NC(V))

$c(X_3) = 11$

f) Connect T at min cost s.t. 1 node can fail



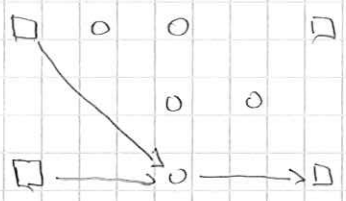
dito

Y_3'

2-node connected subgraph prob (2NC(T))

$c(Y_3') = 7$

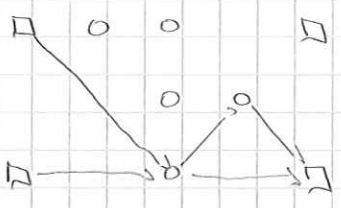
g) Route demand at min cost by shortest paths



X_4

R_4 routing

h) S.t. edge capacities



X_5

R_5

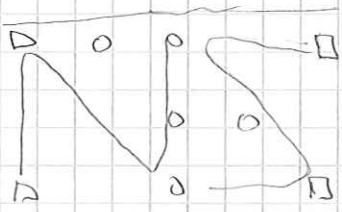
uncapacitated network design problem (UNDP)

$c(X_4) = 3$, $c(R_4) = 2+3 = 4$

capacitated network design problem (CNDP)

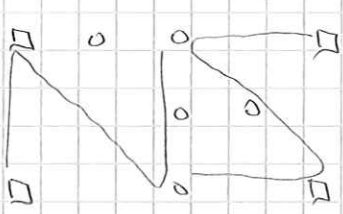
$c(X_5) = 5$, $c(R_5) = 5$

i) Connect V at min cost by paths



P_6

j) Connect T at min cost by paths



P_6'

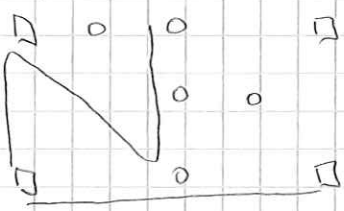
spanning set problem (SSP)

$c(P_6) = 3$

Steiner connectivity problem (SCP)

$c(P_6') = 2$

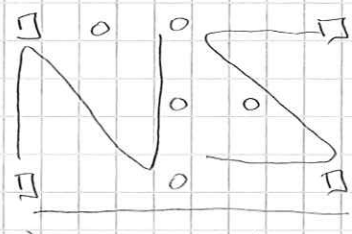
k) Route demand at min cost by shortest paths



P_7

R_7

l) St. path capacities



P_8

R_8

uncapacitated line planning problem (ULPP)

$c(P_7) = 2$, $c(R_7) = 4$

capacitated line planning problem (LPP)

$c(P_8) = 3$, $c(R_8) = 4$

1.2 "Definition": A network design problem involves a (supply) network of potential nodes, edges, paths etc. with associated costs, travel times, capacities etc. that can be installed, such that a transportation demand between certain nodes can be routed according to some routing scheme subject to connectivity, robustness etc. requirements at min total construction and routing costs.

1.3 Definition (graph theoretical notation):

a) V finite set of nodes

$E \subseteq \binom{V}{2} = \{\{u,v\} : u,v \in V\}$ set of unordered edges

$A \subseteq \binom{V}{2} = \{(u,v) : u,v \in V\}$ set of ordered arcs

In c.a., we write uv for $\{u,v\}$ and (u,v).

$W \in E/A$ loop

$G = (V, E)$ undirected graph

$D = (V, A)$ (directed) graph

$H = (W, F)$ with $W \subseteq V$, $F \subseteq E$ subgraph of G

$G[W] = (W, E \cap \binom{W}{2})$ induced by $W \subseteq V$

b) $\delta(W) = \{uv : u \in W, v \notin W\} \subseteq E$ cut induced by $W \subseteq V$

$\delta^+(W) = \{uv : u \in W, v \in W\} \subseteq A$ outcut "

$\delta^-(W) = \{uv : u \notin W, v \in W\} \subseteq A$ incut "

$\delta(v) := \delta(\{v\})$ etc.

$\deg(v) := |\delta(v)|$ etc. degree

$\gamma(v) := \{u : uv \in \delta(v)\}$ etc.

u, v adjacent $\Leftrightarrow uv \in E/A$

c) $(v_0, v_0v_1, v_1, \dots, v_{k-1}, v_{k-1}v_k, v_k)$, $v_i \neq v_j$ except possibly $v_0 = v_k$

$v_iv_{i+1} \in E$ v_0-v_k path, $v_0 = v_k$ cycle, $k = |V|$ Hamiltonian cycle

s,t connected $:\Leftrightarrow \exists$ s-t path

G connected $\Leftrightarrow \exists$ s-t path $\forall s \neq t$

$G[W]$ component $:\Leftrightarrow W$ max. s.t. $s,t \in W$ connected

$e \in E$ bridge $:\Leftrightarrow G \setminus e := (V, E \setminus \{e\})$ has more components than G

d) G tree $:\Leftrightarrow G$ connected and contains no cycle

G forest $:\Leftrightarrow G$ contains no cycle

D arborescence with root r $:\Leftrightarrow \exists!$ rt-path $\forall t \neq r$

D branching $:\Leftrightarrow D$ disjoint union of arborescences

G bipartite $:\Leftrightarrow V = X \dot\cup Y$, $E \subseteq \{xy : x \in X, y \in Y\}$

$\Leftrightarrow G = (X \dot\cup Y, E)$

G complete $:\Leftrightarrow E = \binom{V}{2}$

$\Leftrightarrow$ G bridge connected $\Leftrightarrow \exists$ k s-t paths w. disjoint edges

G k-node connected $\Leftrightarrow$ w. disjoint nodes $\exists$ k s-t paths $\forall s \neq t$

G k-edg connected $:\Leftrightarrow$ $\forall s,t$ k-edge connected $\forall s,t$

$F \subseteq E$ s,t-cut $:\Leftrightarrow \exists W \subseteq V : s \in W, t \notin W, F = \delta(W)$

$W \subseteq V$ s,t-node cut $:\Leftrightarrow s,t \notin W$ and s not connected in $G[V \setminus W]$.

$Y \subseteq E$ Steiner tree for terminals $T \subseteq V$:$\Leftrightarrow$
$(V(Y) \supseteq T, Y)$ is a tree

$Q \subseteq V$ clique :$\Leftrightarrow$ $G[Q]$ is complete

$S \subseteq V$ stable :$\Leftrightarrow$ $G[S] = (S, \emptyset)$

e) $c \in \mathbb{Q}^E$ edge costs

$c(F) = \sum_{e \in F} c_e$ etc.

$\min c(P)$ shortest st-path problem
P st-path

$u \in \mathbb{Q}^E_{\geq 0}$ edge capacities

$x \in \mathbb{Q}^E_{\geq 0}$ st-flow :$\Leftrightarrow$ $x(\delta^+(v)) = x(\delta^-(v))$ $\forall v \neq s,t$

$v(x) := x(\delta^+(s)) - x(\delta^-(s))$ value of st-flow x

x feasible for u :$\Leftrightarrow$ $x \leq u$

$\max_{x \text{ feasible flow}} v(x)$ max-flow problem

$\min_{x \text{ feasible flow of value } v} c^T x$ min cost flow problem

$(d_{st}) \in \mathbb{Q}^{V \times V}_{\geq 0}$ demand matrix (undirected case: symmetric)

$x \in \mathbb{Q}^{V \times V \times E}_{\geq 0}$ multicommodity flow

:$\Leftrightarrow$ x^{st} st-flow $\forall st$

x feasible for u, d :$\Leftrightarrow$ $v(x^{st}) = d_{st}$, $\sum_{st} x^{st}_e \leq u_e$

$\min_{x \text{ feasible multicommodity flow}} c^T x$:$\Leftrightarrow$ multicommodity flow problem

1.4 Definition (linear programming terminology):

$\min c^T x, Ax \leq b$ linear program (LP)

$c \in \mathbb{R}^n$ objective (function)

$A \in \mathbb{R}^{m \times n}$ constraint matrix

$b \in \mathbb{R}^m$ right-hand side

$\ell \leq x \leq u$ lower and upper bounds, $\ell, u \in \mathbb{R}^n$

$P(A,b) = \{x \in \mathbb{R}^n : Ax \leq b\}$ polyhedron

$P(A,b)$ polytope :$\Leftrightarrow$ $P(A,b)$ bounded

Minimum spanning tree problem etc.

$x \in P(A,b)$ vertex $\Leftrightarrow \nexists\, y,z \in P : x = \lambda z + (1-\lambda) y$, $\lambda \in [0,1]$

$\min c^T x$, $Ax = b$, $x \geq 0$ LP in standard form

$P = (A,b) = \{x \geq 0 : Ax = b\}$

$\max y^T b$, $y^T A \leq c^T$ dual of $\min c^T x$, $Ax = b$, $x \geq 0$

$P(A,b)$ integer $\Leftrightarrow$ all vertices of $P(A,b)$ are integer

1.5 Theorem (LP duality):

a) $P(A,b)$ non-empty, bounded $\Leftrightarrow \exists V \in \mathbb{R}^{n \times k} : P(A,b) = \mathrm{conv}(V)$

b) $\Leftrightarrow$ " $\Rightarrow \min c^T x$ has an optimal vertex solution $x \in P(A,b)$

c) $P(A,b)$ non-empty, bounded $\Leftrightarrow$

$\min c^T x$, $Ax = b$, $x \geq 0$ $= \max y^T b$, $y^T A \leq c^T$

d) $P^=(A,b)$ non-empty, A full row rank, $\forall$ vertex $\Rightarrow$

$\exists B \subseteq \{1, \dots, n\}$ basis : $(a_{ij})_{\substack{i=1,\dots,m \\ j \in B}}$ regular, $v = A_B^{-1} b$

1.6 Definition (Integer programming terminology):

$\min c^T x + d^T y$, $Ax + Dy \overset{=}{\geq} b$, $x, y \geq 0$, $x \in \mathbb{Z}^n$ (mixed) integer program

$\min c^T x + d^T y$, $Ax + Dy \overset{=}{\geq} b$, $x, y \geq 0$ LP-relaxation

1.7 Observation:

$\min c^T x + d^T y$, $Ax + Dy \overset{=}{\geq} b$, $x, y \geq 0 \leq \min c^T x + d^T y$, $Ax + Dy \overset{=}{\geq} b$, $x, y \geq 0$, $x \in \mathbb{Z}^n$

1.8 Definition (Duality gap, totally unimodular):

a) $\min_{Ax+Dy=b,\, x,y \geq 0,\, x \in \mathbb{Z}^n} c^T x + d^T y \;-\; \min_{Ax+Dy=b,\, x,y \geq 0} c^T x + d^T$ duality gap

b) $A \in \mathbb{R}^{m \times n}$ totally unimodular $\Leftrightarrow \det A' \in \{0, \pm 1\}$ for every square submatrix of A

1.9 Corollary: If (A,D) is totally unimodular, the LP-relaxation of $\min c^T x + d^T y$, $Ax + Dy = b$, $x, y \geq 0$, $x \in \mathbb{Z}^n$ is integer.

1.10 Definition (Complexity theory terminology):

a) $P: \Pi \subseteq \{0,1\}^* \to \{0,1\}$ (decision problem) (e.g., G connected?)

$I \in \Pi$ instance ($P(I)=1$: true, $P(I)=0$: false) (e.g., G)

$A: \Pi \to \Pi'$, $I \mapsto A(I)$ ... Algorithm

b) $\langle I \rangle :=$ length of bit string I (size of I)

$\langle z \rangle := \lceil \log_2 |z| + 1 \rceil + 1 \quad \forall z \in \mathbb{Z}$

$\langle p/q \rangle := \langle p \rangle + \langle q \rangle \quad \forall p/q \in \mathbb{Q}$

$\langle x \rangle := \sum \langle x_i \rangle \quad \forall x \in \mathbb{Q}^n$

$\langle A \rangle := \sum \langle a_{ij} \rangle \quad \forall A \in \mathbb{Q}^{m \times n}$

$\langle G \rangle := \sum_{uv \in E} \langle u \rangle + \langle v \rangle$ if G is stored as edge list

$:= \left\langle \left(s_{ve} := \begin{cases} 1, & v \in e \\ 0, & v \notin e \end{cases} \right)_{v \times E} \right\rangle$ " as node-edge incidence matrix

$:= \left\langle \left(s_{uv} := \begin{cases} 1, & uv \in e \\ 0, & uv \notin e \end{cases} \right)_{V \times V} \right\rangle$ " as node-node incidence matrix

etc.

[Algorithm A solves (decision) problem P :$\Leftrightarrow$ $A(I) = P(I) \; \forall I \in \Pi(P)$]

c) $f_A: \mathbb{N} \to \mathbb{N}$, $n \mapsto$ max # of elementary operations (addition, multiplication, comparison, assignment) that A needs on a random access machine to process (times the size of the largest number involved) an instance of size n (run time function of A)

A is polynomial $\Leftrightarrow \exists$ polynomial $p: f_A(n) \le p(n) \; \forall n \in \mathbb{N}$

d) Algorithm A solves (decision) problem P non-deterministically

:$\Leftrightarrow \forall I \in P^{-1}(0): \exists J \in \{0,1\}^*$: A can prove $P(I)=0$ using J

A is a non-deterministically polynomial

:$\Leftrightarrow A(\cdot, J(\cdot))$ is polynomial

e) $A: \mathcal{P} \to \mathcal{P}'$, $I \to A(I)$: " $P(I) = P'(A(I))$ "

transformation from problem P to P'

$\star$ polynomial transformation $\Leftrightarrow$ $\star$ polynomial

$\mathcal{P}$ set of problems which can be solved by polynomial alg.

NP " " " non-deterministic " "

NPC " , to which every problem in NP can be transformed (NP-complete)

$P: \Pi \in [0,1]^* \to \mathbb{Q}$ optimization problem

$P': (\Pi, L) \mapsto P(I) \leq L$? decision problem assoc. with P

NPH set of optimization problems, for which the associated decision problem is in NPC (NP hard)

f) Let $f, g: \mathbb{N} \to \mathbb{N}$ be functions

$g = O(f) :\Leftrightarrow \exists c \in \mathbb{N} : g(n) \leq c \cdot f(n)$ for $n \in \mathbb{N}$

g is of the order of f

1.11 Definition (satisfiability problem):

Instance: literals (boolean variables and their negations) $x_1, \dots, x_n$

clauses (disjunctions of literals) $C_1(x), \dots, C_m(x)$

question: Is there an assignment $x \mapsto [0,1]^n$ such that $\bigwedge_{i=1}^{m} C_i(x) = 0$ (satisfying truth assignment)?

1.12 Theorem (Cook [1971]): SAT $\in$ NPC

1.13 Theorem (Karp [1971]):

a) Hamiltonian cycle is NP-complete

b) Clique $\geq k$ "

c) Stable set $\geq k$ "

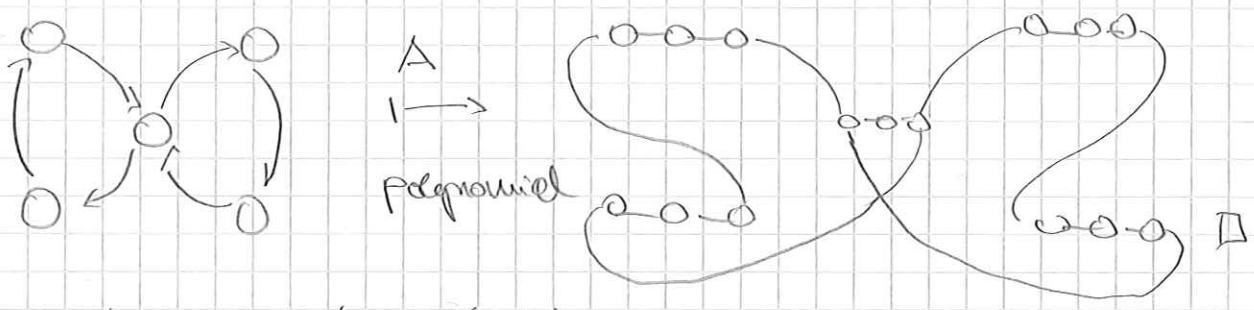
e) Integer programming is NP-hard.

d) Subset sum is NP-complete

1.14 example: Directed Hamiltonian cycle is NP-complete.

1.15 (Subset Sum Problem):
Instance: $a_1, \dots, a_n \in \mathbb{N}$
Question: $\exists I \subseteq [n] : a(I) = a([n])/2$?

Proof:



□

1.15 Algorithm (ggT(a,b)):

Input: $a, b \in \mathbb{N}$, $b \geq a$

Output: ggT(a,b)

Data structure: $c \in \mathbb{N}$

1. do {		
2. $c \leftarrow b$,	$O(1)$	b iterations
3. $b \leftarrow a$	$O(1)$	
4. $a \leftarrow c \bmod b$	$O(1)$	
5. while $(a \neq 0)$	$O(1)$	
6. output b	$O(1)$	

1.16 Theorem: $f_{ggT}(n) = O(n^2)$ ...

Proof: b is at least halved in every iteration, i.e., $k \leq \log_2 b \leq \langle b \rangle \leq n$, the largest number is b. $\Rightarrow f_{ggT}(n) \leq n^2 = O(n^2)$. □