CO 450/650 Combinatorial Optimization

Keven Qiu
Instructor: Bill Cook
Fall 2023

Contents

List of Algorithms

Part I

Introduction

Chapter 1

Introduction

Definition: Combinatorial Optimization

A subfield of mathematical optimization which involves searching for an optimal object in a finite collection of objects.

Typically, the collection has a concise representation, while the number of objects is large. Objects include graphs, networks, and matroids.

The main tool in combinatorial optimization is linear programming duality.

Chapter 2

Linear Programming

Definition: Linear Programming

The problem of finding a vector x that maximizes a given linear function $c^T x$, where x ranges over all vectors satisfying a given system $Ax \leq b$ of linear inequalities.

2.1 Farkas' Lemma

Lemma (Farkas' Lemma for Inequalities)

The system $Ax \leq b$ has a solution x if and only if there is no vector y satisfying $y \geq 0$, $y^T A = 0$, and $y^T b < 0$.

Proof. Suppose $Ax \leq b$ has a solution $\overline{x}$ and suppose there exists a vector $\overline{y} \geq 0$ satisfying $\overline{y}^T A = 0$ and $\overline{y}^T b < 0$. Then we obtain the contradiction

$$0 > \overline{y}^T b \geq \overline{y}^T (A\overline{x}) = (\overline{y}^T A)\overline{x} = 0$$

Now suppose that $Ax \leq b$ has no solution. If A has only one column, then the result is easy. Otherwise, apply Fourier-Motzkin elimination to obtain a system $A'x' \leq b'$ with one less variable. Since $A'x' \leq b'$ also has no solution, we can assume by induction that there exists a vector $y' \geq 0$ satisfying $y'^T A' = 0$ and $y'^T b' < 0$. Now since each inequality in $A'x' \leq b'$ is the sum of positive multiples of inequalities in $Ax \leq b$, we can use y' to construct a vector y satisfying the conditions in the theorem. ■

Lemma (Farkas' Lemma)

The system $Ax = b$ has a nonnegative solution if and only if there is no vector y satisfying $y^T A \geq 0$ and $y^T b < 0$.

Proof. Define

$$A' = \begin{bmatrix} A \\ -A \\ -I \end{bmatrix}, b' = \begin{bmatrix} b \\ -b \\ 0 \end{bmatrix}$$

Then $Ax = b$ has a nonnegative solution x if and only if $A'x' \leq b'$ has a solution x'. Applying Farkas' Lemma for Inequalities to $A'x' \leq b'$ gives the result. ■

Corollary

Suppose the system $Ax \leq b$ has at least one solution. Then every solution x of $Ax \leq b$ satisfies $c^T x \leq \delta$ if and only if there exists a vector $y \geq 0$ such that $y^T A = c$ and $y^T b \leq \delta$.

2.2 Duality

Consider the LP:

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \end{aligned}$$

and dual LP

$$\begin{aligned} \min \quad & y^T b \\ \text{s.t.} \quad & y^T A = c^T \\ & y \geq 0 \end{aligned}$$

Theorem (Weak Duality)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$. Suppose that $\overline{x}$ is a feasible solution to $Ax \leq b, x \geq 0$ and $\overline{y}$ is a feasible solution to $y^T A \geq c^T, y \geq 0$. Then

$$c^T \overline{x} \leq \overline{y}^T b$$

Proof.

$$c^T \overline{x} \leq (\overline{y}^T A)\overline{x} = \overline{y}^T (A\overline{x}) \leq \overline{y}^T b$$

■

Theorem (Duality Theorem)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, then

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

provided that both sets are nonempty.

2.3 Complementary Slackness

Consider the LP and dual LP

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

Definition: Complementary Slackness Conditions

Suppose $\overline{x}, \overline{y}$ are feasible solutions to the primal and dual.

- If a component of $\overline{x} > 0$, then the corresponding inequality in $y^T A \geq c^T$ is satisfied by $\overline{y}$ with equality, i.e.

$$(\overline{y}^T A - c^T)\overline{x} = 0$$

- If a component of $\overline{y} > 0$, then the corresponding inequality in $Ax \leq b$ is satisfied by $\overline{x}$ with equality, i.e.

$$\overline{y}^T(b - A\overline{x}) = 0$$

Theorem (Complementary Slackness Theorem)

Let $\overline{x}$ be a feasible solution of $\max\{c^T x : Ax \leq b, x \geq 0\}$ and let $\overline{y}$ be a feasible solution of $\min\{y^T b : y^T A \geq c^T, y \geq 0\}$. Then the following are equivalent:

(a) $\overline{x}, \overline{y}$ are optimal solutions.

(b) $c^T\overline{x} = \overline{y}^T b$

(c) The complementary slackness conditions hold.

Proof. ((a) $\iff$ (b)) By Duality Theorem.

((b) $\iff$ (c)) By Weak Duality, we have $c^T\overline{x} \leq \overline{y}^T A\overline{x} \leq \overline{y}^T b$. So,

$$\begin{aligned} c^T\overline{x} = \overline{y}^T b &\iff \overline{y}^T A\overline{x} = c^T\overline{x} \text{ and } \overline{y}^T b = \overline{y}^T A\overline{x} \\ &\iff (\overline{y}^T A - c^T)\overline{x}) = 0 \text{ and } \overline{y}^T(b - A\overline{x}) = 0 \end{aligned}$$

■

Chapter 3

Graph Theory

Definition: Graph

A graph $G = (V, E)$ is a set of vertices/nodes V and a set of edges E.
We define $n = |V|$ and $m = |E|$.

Definition: Digraph

A digraph $G = (V, E)$ is a set of vertices/nodes V and a set of edges E, sometimes called arcs, where each $e \in E$ has two ends, one called the head $h(e)$ and the other called the tail $t(e)$.

Definition: Forward Arc

In a path $v_0, e_1, v_1, \ldots, e_k, v_k$, $e_i \in P$ is called forward if $t(e_i) = v_{i-1}$ and $h(e_i) = v_i$.

Definition: Backward Arc

In a path $v_0, e_1, v_1, \ldots, e_k, v_k$, $e_i \in P$ is called backward if $t(e_i) = v_i$ and $h(e_i) = v_{i-1}$.

Definition: Dipath

If all arcs in a path P are forward, then P is a dipath.

Definition: Dicircuit

A dipath that is a circuit.

Definition: Degree

The degree of a vertex v of a graph G is the number of edges incident with v, denoted $\deg_G(v)$.

Definition: Subgraph

H is a subgraph of G if $E(H) \subseteq E(G)$ and $E(H) \subseteq E(G)$.

Definition: Spanning Subgraph

H is spanning if $V(H) = V(G)$.

Definition: Path

A sequence $P = v_0, e_1, v_1, \ldots, e_k, v_k$ where $v_0, \ldots, v_k \in V(G)$, $e_1, \ldots, e_k \in E(G)$, and $e_i = v_{i-1}v_i$.

We call P a v_0v_1-path. The length of P is the number of edges in P.

Definition: Simple Path

A path $v_0, e_1, v_1, \ldots, e_k, v_k$ where all v_i are distinct.

Definition: Edge-Simple Path

A path $v_0, e_1, v_1, \ldots, e_k, v_k$ where all e_i are distinct.

Definition: Closed Path

A path $v_0, e_1, v_1, \ldots, e_k, v_k$ where $v_0 = v_k$.

Definition: Circuit/Cycle

An edge-simple, closed path where $v_0, \ldots, v_{k-1}$ are distinct.

Definition: Connected

A graph is connected if every pair of vertices is joined by a path.

Theorem

A graph G is connected if and only if there is no set $A \subseteq V$ where $\emptyset \neq A \neq V$ with $\delta(A) = \emptyset$.

Definition: Connected Component

A maximal connected subgraph.

Definition: Cut Vertex

A vertex v of a connected graph G where $G - v$ is not connected.

Definition: Forest

A graph with no circuits.

Definition: Tree

A connected forest.

Definition: Cut

Let $R \subseteq V$, then

$$\delta(R) = \{vw : vw \in E, v \in R, w \notin R\}$$

Definition: rs-Cut

A cut for which $r \in R, s \notin R$.

Definition: Directed Spanning Tree

A directed spanning tree rooted at r is a spanning tree that contains a dipath from r to each $v \in V$.

Chapter 4

Complexity Classes

Definition: Decision Problem

A problem with a yes-no answer.

Definition: $\mathcal{P}$

Decision problems that can be solved in polynomial time.

Definition: $\mathcal{NP}$

Decision problems in which we can certify the answer is yes in polynomial time.

Definition: co-$\mathcal{NP}$

Decision problems in which we can certify the answer is no in polynomial time.

A good characterization means the problem is in $\mathcal{NP} \cap co - \mathcal{NP}$.

Definition: $\mathcal{NP}$-Hard

A problem X is $\mathcal{NP}$-hard if every other problem Y in $\mathcal{NP}$ can be reduced to X.

S. Cook (1971) proved that the satisfiability problem (SAT) is $\mathcal{NP}$-hard. R. Karp (1972) used Cook's result to show 21 well-known combinatorial optimization problems are also $\mathcal{NP}$-hard.

To show that the traveling salesman problem (TSP) is $\mathcal{NP}$-hard, we show that any example of SAT can be formulated as a TSP, of size polynomial in the size of SAT. Then, since Cook shows SAT is $\mathcal{NP}$-hard, TSP is also $\mathcal{NP}$-hard.

Part II

Polyhedral Combinatorics

Chapter 5

Integrality of Polyhedra

5.1 Convex Hull

Definition: Convex Combination

$x = \lambda_1 v_1 + \cdots + \lambda_k v_k$ for some vectors $v_1, \ldots, v_k$ and nonnegative scalars $\lambda_1, \ldots, \lambda_k$ such that $\lambda_1 + \cdots + \lambda_k = 1$.

Definition: Convex Hull

The convex hull of a finite set S, denoted $conv.hull(S)$, is the set of all vectors that can be written as a convex combination of S.
It is also defined as the smallest convex set containing S.

Proposition

Let $S \subseteq \mathbb{R}^n$ be a finite set and let $w \in \mathbb{R}^n$. Then

$$\max / \min\{w^T x : x \in S\} = \max / \min\{w^T x : x \in conv.hull(S)\}$$

Theorem (Minkowski)

If S is finite, then $conv.hull(S)$ is a polyhedron.

$$\begin{aligned} \max\{w^T x : x \in S\} &= \max\{w^T x : x \in conv.hull(S)\} \\ &= \max\{w^T x : Ax \leq b\} \\ &= \min\{y^T b : y^T A = w^T, y \geq 0\} \end{aligned}$$

So we can use LP duality to attack combinatorial problems. If we understand $Ax \leq b$, then the problem is in co-$\mathcal{NP}$. Thus, if we have an algorithm to produce the inequalities in $Ax \leq b$ (separation), then the problem is in $\mathcal{P}$ (Ellipsoid method).

5.2 Polytopes

Definition: Polyhedron

A set of the form $\{x : Ax \leq b\}$.

In combinatorial optimization, we typically have $x \geq 0$ as a constraint, so we have polyhedra of the form $\{x : Ax \leq b, x \geq 0\}$.

Definition: Polytope

A polyhedron $P \subseteq \mathbb{R}^n$ is a polytope if there exists $\ell, u \in \mathbb{R}^n$ such that $\ell \leq x \leq u$ for all $x \in P$.

Definition: Convex Set

Let P be a polyhedron, $x_1, x_2 \in P$, and $0 \leq \lambda \leq 1$. If $\lambda x_1 + (1 - \lambda)x_2 \in P$, then P is a convex set.

Definition: Valid Inequality

An inequality $w^T x \leq t$ is valid for a polyhedron P if $P \subseteq \{x : w^T x \leq t\}$.

Definition: Hyperplane

The solution set of $w^T x = t$ where $w \neq 0$.

Definition: Supporting Hyperplane

With respect to a polyhedron P, a hyperplane is supporting if $w^T x \leq t$ is valid for P and $P \cap \{x : w^T x = t\} \neq \emptyset$.

Definition: Face

The intersection of a polyhedron with one of its supporting hyperplanes.

The null set and the polyhedron itself is a face.

Definition: Proper Face

Faces which are not the null set or the polyhedron itself.

Proposition

A nonempty set $F \subseteq P = \{x : Ax \leq b\}$ is a face of P if and only if for some subsystem $A^\circ x \leq b^\circ$ of $Ax \leq b$, we have $F = \{x \in P : A^\circ x = b^\circ\}$.

Proof. ($\implies$) Suppose F is a face of P. Then there exists a valid inequality $w^T x \leq t$ such that $F = \{x \in P : w^T x = t\}$.

Consider the LP problem $\max\{w^T x : Ax \leq b\}$. The set of optimal solutions is precisely F. Now let y^* be an optimal solution to the dual problem $\min\{y^T b : y^T A = w, y \geq 0\}$ and let $A^\circ x \leq b^\circ$ be those inequalities $a_i^T x \leq b_i$ whose corresponding dual variable y_i^* is positive. By complementary slackness, we have $F = \{x : Ax \leq b, A^\circ x = b^\circ\}$ as required.

($\Longleftarrow$) Conversely, if $F = \{x \in P : A^\circ x = b^\circ\}$ for some subsystem $A^\circ x \leq b^\circ$ of $Ax \leq b$, then add the inequalities $A^\circ x \leq b^\circ$ to obtain an inequality $w^T x \leq t$. Every $x \in F$ satisfies $w^T x = t$ and every $x \in P \setminus F$ satisfies $w^T x < t$ as required. ■

Proposition

Let F be a minimal nonempty face of $P = \{x : Ax \leq b\}$. Then $F = \{x : A^\circ x = b^\circ\}$ for some subsystem $A^\circ x \leq b^\circ$ of $Ax \leq b$.
Moreover, the rank of the matrix A° is equal to the rank of A.

Definition: Vertex/Extreme Point

A vector $x \in P$ is called a vertex/extreme point if $\{x\}$ is a face of P.
Equivalently, $x \in P$ is a vertex/extreme point if x cannot be written as $\frac{1}{2}x_1 + \frac{1}{2}x_2$ for points $x_1, x_2 \in P, x_1 \neq x_2$.

Note: Not all polyhedra have vertices, but if $P \subseteq \mathbb{R}^n_+$, then P has vertices.

LP Fact

If a polyhedron P has vertices, then the set of optimal LP solutions contains at least one vertex of P.
Moreover, if all vertices of P are integral, then the LP always has an integral optimal solution.

Definition: Pointed Polyhedron

A polyhedron P is pointed if it has at least one vertex.

$\{(x_1, x_2) \in \mathbb{R}^2 : x_1 \geq 0\}$ is a polyhedron with no vertex.

Proposition

If a polyhedron P is pointed, then every minimal nonempty face of P is a vertex.

Proposition

Let $P = \{x : Ax \leq b\}$ and $v \in P$. Then v is a vertex of P if and only if v cannot be written as a convex combination of vectors in $P \setminus \{v\}$.

Theorem

A polytope is equal to the convex hull of its vertices.

Proof. Let P be a nonempty polytope. Since P is bounded, P must be pointed. Let

$v_1, \ldots, v_k$ be the vertices of P. Clearly, $conv.hull(\{v_1, \ldots, v_k\}) \subseteq P$. So suppose there exists

$$u \in P \setminus conv.hull(\{v_1, \ldots, v_k\})$$

Then by proposition, there exists an inequality $w^T x \leq t$ that separates u from

$$conv.hull(\{v_1, \ldots, v_k\})$$

Let $t^* = \max\{w^T x : x \in P\}$ and consider the face $F = \{x \in P : w^T x = t^*\}$. Since $u \in P$, we have $t^* > t$. So F contains no vertex of P, a contradiction. ■

Theorem

A set P is a polytope if and only if there exists a finite set V such that P is the convex hull of V.

5.3 Integral Polytopes

Definition: Rational Polyhedron

A polyhedron that can be defined by rational linear systems.

Definition: Integral Polyhedron

A rational polyhedron where every nonempty face contains an integral vector.

Definition: Pointed Integral Polyhedron

A pointed rational polyhedron is integral if and only if all its vertices are integral.

Theorem

A rational polytope P is integral if and only if for all integral vectors w, the optimal value of $\max\{w^T x : x \in P\}$ is an integer.

Proof. To prove sufficiency, suppose that for all integral vectors w, the optimal value of $\max\{w^T x : x \in P\}$ is an integer. Let $v = (v_1, \ldots, v_n)^T$ be a vertex of P and let w be an integral vector such that v is the unique optimal solution to $\max\{w^T x : x \in P\}$. By multiplying w by a large positive integer if necessary, we may assume $w^T v > w^T u + u_1 - v_1$ for all vertices u of P other than v. This implies that if we let $\overline{w} = (w_1 + 1, w_2, \ldots, w_n)^T$, then v is an optimal solution to $\max\{\overline{w}^T x : x \in P\}$. So $\overline{w}^T v = w^T v + v_1$. But, by assumption, $w^T v$ and $\overline{w}^T v$ are integers. Thus, v_1 is an integer. We can repeat this for each component of v, so v must be integral. ■

5.4 Total Unimodularity

Proposition

Let A be an integral, nonsingular, $m \times n$ matrix. Then $A^{-1}b$ is integral for every integral vector $b \in \mathbb{R}^m$ if and only if $\det(A) = 1$ or -1.

Proof. ($\Longleftarrow$) Suppose $\det(A) = \pm 1$. By Cramer's Rule, we know that A^{-1} is integral, which implies $A^{-1}b$ is integral for every integral b.

($\Longrightarrow$) Conversely, suppose $A^{-1}b$ is integral for all integral vectors b. Then, in particular, $A^{-1}e_i$ is integral for all $i = 1, \ldots, m$. This means that A^{-1} is integral. So $\det(A)$ and $\det(A^{-1})$ are both integers. But, $\det(A) \cdot \det(A^{-1}) = 1$, this implies $\det(A) = \pm 1$. ■

Definition: Unimodular

A matrix A of full row rank is unimodular if A is integral and each basis of A has determinant ± 1.

Theorem (Veinott & Dantzig 1968)

Let A be an integral $m \times n$ matrix of full row rank. Then the polyhedron defined by $Ax = b, x \geq 0$ is integral for every integral vector $b \in \mathbb{R}^m$ if and only if A is unimodular.

Proof. ($\Longleftarrow$) Suppose A is unimodular. Let $b \in \mathbb{R}^m$ be an integral vector and let $\overline{x}$ be a vertex of $\{x : Ax = b, x \geq 0\}$. The nonnegativity constraints implies the polyhedron has vertices. Then there are n linearly independent constraints satisfied by $\overline{x}$ with inequality. It follows that the columns of A corresponding to the nonzero components of $\overline{x}$ are linearly independent. Extending these columns to a basis B of A, we have the nonzero components of $\overline{x}$ are contained in the integral vector $B^{-1}b$. So $\overline{x}$ is integral.

($\Longrightarrow$) Conversely, suppose $\{x : Ax = b, x \geq 0\}$ is integral for all integral vectors b. Let B be a basis of A and let v be an integral vector in $\mathbb{R}^m$. By previous proposition, it suffices to show that $B^{-1}v$ is integral. Let y be an integral vector such that $y + B^{-1}v \geq 0$ and let $b = B(y + B^{-1}v)$. Note b is integral. Furthermore, by adding zero components to the vector $y + B^{-1}v$, we can obtain a vector $z \in \mathbb{R}^n$ such that $Az = b$. Then, z is a vertex of $\{x : Ax = b, x \geq 0\}$, since z is a polyhedron and satisfies n linearly independent constraints with equality: the m equations $Ax = b$ and the $n - m$ equations $x_i = 0$ for the columns i outside B. So z is integral, and thus, $B^{-1}v$ is integral. ■

Definition: Totally Unimodular (TU)

A matrix is totally unimodular if all of its square submatrices have determinant 0, 1, or -1.

It is easy to see that A is totally unimodular if and only if $\begin{bmatrix} A & I \end{bmatrix}$ is unimodular where $I \in \mathbb{R}^{m \times m}$.

Theorem (Hoffman-Kruskal)

Let A be an $m \times n$ integral matrix. Then the polyhedron defined by $Ax \leq b, x \geq 0$ is integral for every integral vector $b \in \mathbb{R}^m$ if and only if A is totally unimodular.

Proof. Applying the linear programming trick of adding slack variables, we have that for any integral b, the polyhedron $\{x : Ax \leq b, x \geq 0\}$ is integral if and only if the polyhedron $\{z : \begin{bmatrix} A & I \end{bmatrix} z = b, z \geq 0\}$ is integral. So the result follows from previous theorem. ■

Theorem

Let A be an $m \times n$ totally unimodular matrix and let $b \in \mathbb{R}^m$ be an integral vector. Then the polyhedron defined by $Ax \leq b$ is integral.

Proof. Let F be a minimal face of $\{x : Ax \leq b\}$. Then, by proposition, $F = \{x : A^\circ x = b^\circ\}$ for some subsystem $A^\circ x \leq b^\circ$ of $Ax \leq b$, with A° having full row rank. By reordering the columns, if necessary, we may write A° as $\begin{bmatrix} B & N \end{bmatrix}$ where B is a basis of A°. It follows

$$\overline{x} = \begin{bmatrix} B^{-1}b^\circ \\ 0 \end{bmatrix}$$

is an integral vector in F. ■

Theorem

Let A be a $0, \pm 1$ valued matrix where each column has at most one $+1$ and at most -1. Then A is totally unimodular.

Proof. Let N be a $k \times k$ submatrix of A. If $k = 1$, then $\det(N)$ is either 0 or ± 1. So we may suppose that $k \geq 2$ and proceed by induction on k. If N has a column having at most one nonzero, then expanding the determinant along this column, we have that $\det(N)$ is either 0 or ± 1, by induction. On the other hand, if every column of N has both a $+1$ and a -1, then the sum of the rows of N is 0 and hence $\det(N) = 0$. ■

Proposition

A is totally unimodular if and only if A^T is totally unimodular.

5.5 Optimization and Separation

There are many ways to be given a polyhedron P. Either in the form $\{x : Ax \leq b\}$, $conv.hull(\{a_1, \ldots, a_m\})$, or a graph's convex hull of matchings or TSP tours.

Optimization Problem

Given a bounded rational polyhedron $P \subseteq \mathbb{R}^n$ and a rational objective vector $c \in \mathbb{Q}^n$, solve $\max\{c^T x : x \in P\}$.

Separation Problem

Given a bounded rational polyhedron $P \subseteq \mathbb{R}^n$ and a rational vector $\overline{x} \in \mathbb{Q}^n$, either

1. assert $\overline{x} \in P$, or
2. find $c \in \mathbb{Q}^n, \delta \in \mathbb{Q}$ such that $c^T x \leq \delta$ is satisfied for all $x \in P$ (valid) and $c^T \overline{x} > \delta$ (violated by x).

Such a $c^T x \leq \delta$ is called a separating inequality and a $c^T x = \delta$ is called a separating hyperplane.

Definition: Well-Described Polyhedron

A triple (P, n, φ) where $P \subseteq \mathbb{Q}^n$ is a rational polyhedron with facet complexity at most φ.

Theorem (Grötschel, Lovász, Schrijver 1982)

For a well-described polyhedron, optimization can be solved in polynomial time if and only if separation can be solved in polynomial time.

This theorem is known as the optimization $\equiv$ separation theorem. We can use the ellipsoid method for solving LPs.

This is a fantastic result, as it shows that polynomial time solvability for combinatorial problems, such as TSP or matchings, is directly tied to our ability to understanding the convex hull of solutions. This is due to Minkowski.

Edmonds gives a polynomial time optimization for $\mathcal{PM}(G)$ and by GLS, we have a polynomial time separation for $\mathcal{PM}(G)$. But it is much easier to go the other way, first describing a polynomial time separation algorithm.

5.6 Total Dual Integrality

Definition: Totally Dual Integral (TDI)

A rational linear system $Ax \leq b$ is totally dual integral if for every integral w such that the dual LP

$$\min\{y^T b : y^T A = w^T, y \geq 0\}$$

has an optimal solution, it has an optimal solution that integer valued.

Theorem (Total Dual Integrality Theorem)

If $Ax \leq b$ is totally dual integral and b is integral, then the polyhedron $P = \{x : Ax \leq b\}$ is integral.

Proof. If $Ax \leq b$ is TDI and b is integral, then for all integral w such that the optima exist, both sides of the duality equation

$$\max\{w^T x : Ax \leq b\} = \min\{y^T b : y^T A = w^T, y \geq 0\}$$

have integer optimal solutions. ■

Proving a system is TDI is one way to prove that a polyhedron is integral.

5.7 Cutting Planes

Given a finite $Q \subseteq \mathbb{R}^n, w \in \mathbb{R}^n$, find $\min\{w^T x : x \in Q\}$.

$$\begin{aligned}\min\{w^T x : x \in Q\} &= \min\{w^T x : x \in conv.hull(Q)\} \\ &= \min\{w^T x : Ax \leq b\} \\ &= \max\{y^T b : y^T A = w^T, y \geq 0\}\end{aligned}$$

In general, we do not know $Ax \leq b$, but we can apply cutting planes if we know how to find at least some inequalities valid for Q. We need to be able to solve the separation problem to drive the cutting plane method.

Algorithm 1 Cutting Plane Algorithm

1: Start with LP relaxation $\min\{w^T x : x \in Q\}$ (all constraints valid for Q)
2: x^* be an optimal solution to the LP
3: **while** $x^* \notin Q$ **do**
4: Add to the LP one or more inequalities valid for Q but violated by x^*
5: Solve new LP and x^* is new optimal solution

Note: We can stop the method anytime, using a dual LP solution to give a lower bound for the problem.

For many classes of problems, there are known families of good inequalities to use.

Part III

Optimal Trees and Paths

Chapter 6

Minimum Spanning Trees

6.1 Problem

Definition: Spanning Tree

A subgraph $T \subseteq G$ where $V(T) = V(G)$, T is connected, and T is acyclic.

Lemma

An edge $e = uv$ of G is an edge of a circuit of G if and only if there is a path in $G \setminus e$ from u to v.

Minimum Spanning Tree Problem (MST)

Given a connected graph G and a real cost c_e for each $e \in E$, find a minimum cost spanning tree of G.

Lemma

A spanning connected subgraph of G is a spanning tree if and only if it has exactly $n - 1$ edges.

6.2 Kruskal's Algorithm

Theorem

Kruskal's algorithm finds a MST.

This is a polynomial time algorithm and is very fast in practice for sparse graphs. We can maintain F with a union-find data structure.

Algorithm 2 Kruskal's Algorithm (Minimum Spanning Tree)

Sort E to $\{e_1, \ldots, e_m\}$ so that $c_{e_1} \leq \cdots \leq c_{e_m}$
$F = \emptyset, H = (V, F)$
for $i = 1$ to m **do**
    if ends of e_i are in different components of H **then**
        $F \leftarrow F \cup \{e_i\}$
return H

6.3 Linear Programming

Definition: $\kappa : E \to \mathbb{N}$

For $A \subseteq E$, $\kappa(A)$ is the number of components in the subgraph (V, A) of G.

The maximum number of tree edges in A is $|V| - \kappa(A)$.

We can formulate the MST problem as an integer program.

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & \sum(x_e : e \in A) \leq |V| - \kappa(A), \ \forall A \subsetneq E \\ & \sum(x_e : e \in E) = |V| - 1 \\ & x_e \in \{0, 1\}, \ \forall e \in E \end{aligned}$$

We can relax the integer program to get the following linear program.

Definition: MST LP

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & x(A) \leq |V| - \kappa(A), \ \forall A \subsetneq E \\ & x(E) = |V| - 1 \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

We replace the minimization with a maximization in the primal to write the dual.

Definition: MST Dual LP

$$\begin{aligned} \min \quad & \sum((|V| - \kappa(A))y_A : A \subseteq E) \\ \text{s.t.} \quad & \sum(y_A : e \in A) \geq -c_e, \ \forall e \in E \\ & y_A \geq 0, \ \forall A \subsetneq E \end{aligned}$$

6.3.1 Complementary Slackness Conditions

Let T be a tree found by Kruskal's algorithm. Define the characteristic vector of T

$$x_e^0 = \begin{cases} 1 & \text{if } e \in E(T) \\ 0 & \text{if } e \notin E(T) \end{cases}$$

Definition: MST Complementary Slackness Conditions

(i) For all $e \in E$, if $x_e^0 > 0$, then $\sum(y_A^0 : e \in A) = -c_e$.

(ii) For all $A \subsetneq E$, if $y_A^0 > 0$, then $\sum(x_e^0 : e \in A) = |V| - \kappa(A)$.

Theorem (Edmonds 1971)

Let x^0 be the characteristic vector of an MST with respect to costs c_e. Then x^0 is an optimal solution to the MST LP.

Proof. We show that x^0 is optimal for the LP and x^0 is the characteristic vector generated by Kruskal's algorithm.

Let $e_1, \ldots, e_m$ be the order in which Kruskal's algorithm considers the edges. Let $R_i = \{e_1, \ldots, e_i\}$ for $1 \leq i \leq m$. Let y^0 be the dual solution.

- $y_A^0 = 0$ if A is not one of the R_i's.
- $y_{R_i}^0 = c_{e_{i+1}} - c_{e_i}$ for $1 \leq i \leq m-1$.
- $y_{R_m}^0 = -c_{e_m}$

It follows from the ordering of the edges, $y_A^0 \geq 0$ for $A \neq E$. Now consider the first constraint, then where $e = e_i$, we have

$$\sum(y_A^0 : e \in A) = \sum_{j=i}^{m} y_{R_j}^0 = \sum_{j=i}^{m-1} (c_{e_{j+1}} - c_{e_j}) - c_{e_m} = -c_{e_i} = -c_{e_i} = -c_e$$

All of the inequalities hold with equality. So y^0 is a feasible dual solution and complementary slackness condition (i) holds.

Now suppose $y_A^0 > 0$ for some $A \subsetneq E$. Thus, $A = R_i$ for some i. Consider the constraint

$$\sum(x_e^0 : e \in R_i) \leq |V| - \kappa(R_i)$$

If this does not hold with equality, then there is some edge of R_i having ends in two different components of $(V, R_i \cap T)$ and this would have been added to T by Kruskal's algorithm. So (x^0, y^0) satisfy the complementary slackness conditions, which means they are optimal solutions to their LPs. Therefore, T is an MST. ■

6.4 Spanning Tree Polytope

Definition: Spanning Tree Polytope

$conv.hull\{x^H : H \text{ is a spanning tree}\}$ where

$$x_e^H = \begin{cases} 1 & \text{if } e \in E(H) \\ 0 & \text{if } e \notin E(H) \end{cases}$$

Theorem

The spanning tree polytope is the solution set to the following linear system:

$$\sum(x_e : e \in A) \leq |V| - \kappa(A), \ \forall A \subsetneq E$$
$$\sum(x_e : e \in E) = |V| - 1$$
$$x_e \geq 0, \ \forall e \in E$$

Proof. Let P be the solution set of the linear system. We showed that for any edge costs $(c_e : e \in E)$, the LP

$$\max\{\sum(-c_e x_e : e \in E) : x \in P\}$$

has an integral optimal solution. So every vertex of P is integral. ■

Chapter 7

Shortest Paths

Shortest Path Problem

Given a digraph G, a vertex $r \in V$, and a real cost vector $(c_e : e \in E)$, find for each $v \in V$, a minimum-cost dipath from r to v.

Note: You can provide a solution to the shortest path problem for r by giving a directed spanning tree rooted at r.

Proof. For each $v \in V \setminus \{r\}$, all shortest paths have at most one arc having head v_j, since the only such arc we need is the last arc of one min-cost rv-dipath.

So the union of the arc sets of all the shortest paths has exactly $|V| - 1$ arcs and thus is a tree. ■

Important Case

If $c_e \geq 0$ for all $e \in E$, then this problem is handled by Dijkstra's algorithm, which starts at r and grows the tree vertex by vertex.

However, the Hamiltonian dipath problem (does G have a simple dipath P with $V(P) = V(G)$) is $\mathcal{NP}$-hard.

When G has negative-cost dicircuits, this is a problem, since there is no shortest path as we can loop around the dicircuit an infinite amount of times. There do exist polynomial time algorithms that either finds a shortest path or detect a negative-cost dicircuit.

Definition: Feasible Potential

$y = (y_v : v \in V)$ is a feasible potential if it satisfies $y_v + c_{vw} \geq y_w$ for all $vw \in E$.

Proposition

Let y be a feasible potential and let P be an rs-dipath. Then $c(P) \geq y_s - y_r$.

Proof. Suppose that P is $v_0, e_1, v_1, \ldots, e_k, v_k$ where $v_0 = r$ and $v_k = s$. Then

$$c(P) = \sum_{i=1}^{k} c_{e_i} \geq \sum_{i=1}^{k} (y_{v_i} - y_{v_{i-1}}) = y_{v_k} - y_{v_0} = y_s - y_r$$

■

So a potential y provides a stopping rule. A dipath P and a potential y with $c(P) = y_s - y_r$ implies P is optimal.

7.1 Ford's Algorithm

Definition: Incorrect

Given vertex values $(y_v : v \in V)$, the edge vw is incorrect if $y_v + c_{vw} < y_w$.

To correct vw, we set $y_w = y_v + c_{vw}$ and predecessor$(w) = v$.

Algorithm 3 Ford's Algorithm (Shortest Path)

1: $y_r = 0, y_v = \infty$ for all $v \in V \setminus \{r\}$
2: predecessor$(r) = \emptyset$, predecessor$(v) = -1$ for all $v \in V \setminus \{r\}$
3: **while** y is not a feasible potential **do**
4: Find an incorrect arc vw and correct vw

Theorem

If there are no negative-cost dicircuits, then Ford's algorithm terminates in a finite number of steps.
At termination, for each $v \in V$, the predecessors define a shortest rv-dipath of cost y_v.

Specialized versions like Ford-Bellman run in polynomial time.

7.2 Linear Programming

Definition: Shortest Path LP

$$\begin{aligned} \max \quad & y_s - y_r \\ \text{s.t.} \quad & y_w - y_v \leq c_{vw}, \ \forall vw \in E \end{aligned}$$

Definition: Shortest Path Dual LP

$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & \sum(x_{wv} : wv \in E) - \sum(x_{vw} : vw \in E) = \begin{cases} 0 & \text{if } v \in V \setminus \{r, s\} \\ -1 & \text{if } v = r \\ 1 & \text{if } v = s \end{cases} \\ & x_{vw} \geq 0, \ \forall vw \in E \end{aligned}$$

Any rs-dipath is a solution to the dual LP, so if the dual LP has an optimal solution, then it has an optimal solution that is an rs-dipath.

The constraint matrix for the LP is totally unimodular.

Theorem

Let G be a digraph, $r, s \in V$, and $c \in \mathbb{R}^E$. If there exists a minimum-cost dipath from r to v for every $v \in V$, then

$$\min\{c(P) : P \text{ an } rs\text{-dipath}\} = \max\{y_s : y \text{ a feasible potential}\}$$

The vertices of the polyhedron defined by the dual LP constraints are the vectors x^P of simple dipaths.

$$x_e^P = \begin{cases} 1 & \text{if } e \in E(P) \\ 0 & \text{if } e \notin E(P) \end{cases}$$

Note: This is *not* the convex hull of simple dipaths.

Since the matrix is totally unimodular, we could add $x_{vw} \leq 1$ for all $vw \in E$, but this will not give simple dipaths.

Part IV

Network Flows

Chapter 8

Maximum Flow

8.1 Problem

Definition: Net Flow/Excess

$$f_x(v) = x(\delta(\overline{v})) - x(\delta(v)) = \sum(x_{wv} : w \in V, wv \in E) - \sum(x_{vw} : w \in V, vw \in E)$$

Definition: rs-Flow

A vector x that satisfies $f_x(v) = 0$ for all $v \in V$.

Definition: Value of rs-Flow

$$f_x(s)$$

Maximum Flow Problem

Given a digraph $G = (V, E)$, with source r and sink s, find an rs-flow of maximum value.

8.2 Augmenting Path Algorithm

Definition: Augmenting Path

An rs-path P is x-augmenting if for all forward arcs e we have $x_e < u_e$, and for all reverse arcs e we have $x_e > 0$.

Given a flow $x = (x_e : e \in E)$ and augmenting path P, we can augment flow x by the largest ε. There is some forward arc with $x_e + \varepsilon = u_e$ or some reverse arc has $x_e - \varepsilon = 0$. The value ε is called the x-width of P.

Algorithm 4 Ford-Fulkerson Algorithm (Maximum Flow/Minimum Cut)

$x = 0$
while there is an x-augmenting path P **do**
    $\varepsilon_1 = \min(u_e - x_e : e \text{ forward in } P)$
    $\varepsilon_2 = \min(x_e : e \text{ reverse in } P)$
    $\varepsilon = \min(\varepsilon_1, \varepsilon_2)$ // x-width of P
    if $\varepsilon = \infty$ **then**
        No maximum flow
return x is maximum flow, set R of vertices reachable by an x-augmenting path from r is minimum cut

Definition: Auxiliary Digraph

$G(x)$, depending on G, u, x, where $V(G(x)) = V$ and $vw \in E(G(x))$ if and only if $vw \in E$ and $x_{vw} < u_{vw}$ or $wv \in E$ and $x_{wv} > 0$.

rs-dipaths in $G(x)$ corresponding to x-augmenting paths in G. Each iteration of Ford-Fulkerson can be performed in $O(m)$ time, using breadth-first search.

Theorem

If u is integral and the maximum flow value is $K < \infty$, then the maximum flow algorithm terminates after at most K augmentations.

8.2.1 Shortest Augmenting Paths

Theorem (Dinits 1970, Edmonds & Karp 1972)

If each augmentation of the augmenting path algorithm on a shortest augmenting path, then there are at most nm augmentations.

Corollary

The augmenting path algorithm with breadth-first search solves the maximum flow problem in time $O(nm^2)$.

Let $d_x(v, w)$ be the least length of a vw-dipath in $G(x)$. $d_x(v, w) = \infty$ if no vw-dipath exists.

Consider a typical augmentation from flow x to flow x' determined by the augmenting path P having vertex-sequence $v_0, \ldots, v_k$.

Lemma

For each $v \in V$, $d_{x'}(r, v) \geq d_x(r, v)$ and $d_{x'}(v, s) \geq d_x(v, s)$.

Proof. Suppose that there exists a vertex v such that $d_{x'}(r, v) < d_x(r, v)$ and choose such v so that $d_{x'}(r, v)$ is as small as possible. Clearly, $d_{x'}(r, v) > 0$. Let P' be a rv-dipath in $G(x')$

of length $d_{x'}(r,v)$ and let w be the second-last vertex of P'. Then

$$d_x(r,v) > d_{x'}(r,v) = d_{x'}(r,w) + 1 \geq d_x(r,w) + 1$$

It follows that wv is an arc of $G(x')$, but not of $G(x)$, otherwise $d_x(r,v) \leq d_x(r,w) + 1$, so $w = v_i$ and $v = v_{i-1}$ for some i. But, this implies that $i - 1 > i + 1$, a contradiction. The second statement is similar. ■

Definition: $\tilde{E}(x)$

$$\tilde{E}(x) = \{e \in E : e \text{ is an arc of a shortest } x\text{-augmenting path}\}$$

Lemma

If $d_{x'}(r,s) = d_x(r,s)$, then $\tilde{E}(x') \subsetneq \tilde{E}(x)$.

Proof. Let $k = d_x(r,s)$ and suppose that $e \in \tilde{E}(x')$. Then e induces an arc vw of $G(x')$ and $d_{x'}(r,v) = i-1$, $d_{x'}(ws) = k-i$ for some i. Therefore, $d_x(r,v) + d_x(w,s) \leq k-1$ by previous lemma. Now suppose that $e \notin \tilde{E}(x)$, then $x_e \neq x'_e$, so e is an arc of P, a contradiction. This proves $\tilde{E}(x') \subseteq \tilde{E}(x)$.

There is an arc e of P such that e is forward and $x'_e = u_e$ or e is reverse and $x'_e = 0$. Therefore, any x'-augmenting path using e must use it in the opposite direction from P, so its length, for some i, will be at least $i + k - i + 1 + 1 = k + 23$, so $e \notin \tilde{E}(x')$. ■

Proof. (Dinits, Edmonds, Karp) It follows from previous lemma that there can be at most m augmentations per stage. Since there are at most $n - 1$ stages, there are at most nm augmentations in all. ■

8.3 Linear Programming

Definition: Maximum Flow LP

$$\begin{aligned} \max \quad & f_x(s) \\ \text{s.t.} \quad & f_x(v) = 0, \ \forall v \in V \setminus \{r,s\} \\ & 0 \leq x_e \leq u_e, \ \forall e \in E \end{aligned}$$

We give a different LP approach to this problem.

Definition: Minimum Cut LP

$$\begin{aligned} \min \quad & \sum(u_e y_e : e \in E) \\ \text{s.t.} \quad & \sum(y_e : e \in E(P)) \geq 1, \ \forall \text{ simple } rs\text{-dipaths } P \\ & y_e \geq 0, \ \forall e \in E \end{aligned}$$

Every rs-cut $\delta(R)$ gives a feasible solution

$$y_e^R = \begin{cases} 1 & \text{if } e \in \delta(R) \\ 0 & \text{if } e \notin \delta(R) \end{cases}$$

Definition: Maximum Flow LP (Dual Minimum Cut LP)

$$\begin{aligned} \max \quad & \sum(w_P : P \text{ a simple } rs\text{-dipath}) \\ \text{s.t.} \quad & \sum(w_P : e \in E(P)) \leq u_e, \ \forall e \in E \\ & w_P \geq 0, \ \forall \text{ simple dipaths } P \end{aligned}$$

Let x be a max flow. We want to find a simple rs-dipath P such that $x_e > 0$ for each $e \in E(P)$. Set $w_P = \min\{x_e : e \in E(P)\}$ and let $x_e = x_e - w_P$ for all $e \in E(P)$. We repeat until $\sum(x_{rv} : rv \in E) = 0$.

$\sum(w_P : P \text{ a simple } rs\text{-dipath})$ is equal to the original value of the flow. Therefore, the max flow equals the min cut which implies the two LPs have integral optimal solutions if u_e is integral for all $e \in E$.

Proposition

There exists a family $(P_1, \ldots, P_k)$ of rs-dipaths such that $|\{i : e \in P_i\}| \leq u_e$ for all $e \in E$ if and only if there exists an integral feasible rs-flow of value k.

Proof. ($\Longrightarrow$) We have seen family of dipaths determines a corresponding flow.

($\Longleftarrow$) Let x be a flow. We assume that x is acyclic, that is, there is no dicircuit C, each of whose arcs e has $x_e > 0$. If a dicircuit does exist, we can decrease x_e by 1 on all arcs of C. The new x remains feasible of value k.

If $k \geq 1$, we can find an arc vs with $x_{vs} \geq 1$. Then, if $v \neq r$, it follows that there is an arc wv with $x_{wv} \geq 1$ by the constraint $f_x(v) = 0$. If $w \neq r$, then the argument can be repeated producing distinct vertices, since x is acyclic, so we get a simple rs-dipath P_k on each arc e with $x_e \geq 1$. We can decrease x_e by 1 for each $e \in P_k$. The new x is an integral feasible flow of value $k - 1$, and the process is repeated. ■

Definition: Path Flow

A vector $x \in \mathbb{R}^E$ such that for some rs-dipath P and some $\alpha \in \mathbb{R}$, $x_e = \alpha$ for each $e \in P$ and $x_e = 0$ for every other arc of G.

Definition: Circuit Flow

A vector $x \in \mathbb{R}^E$ such that for some rs-dicircuit C and some $\alpha \in \mathbb{R}$, $x_e = \alpha$ for each $e \in C$ and $x_e = 0$ for every other arc of G.

Proposition

Every rs-flow of nonnegative value is the sum of at most m flows, each of which is a path flow or a circuit flow.

Proposition

For any rs-cut $\delta(R)$ and any rs-flow x, we have

$$f_x(s) = x(\delta(R)) - x(\delta(\overline{R}))$$

Proof. We add the equations $f_x(v) = 0$ for all $v \in \overline{R} \setminus \{s\}$ as well as the identity $f_x(s) = f_x(s)$. The right hand side sums to $f_x(s)$.

For any arc vw with $v, w \in R$, x_{vw} occurs in none of the equations, so it does not occur in the sum. If $v, w \in \overline{R}$, then x_{vw} occurs in the equation for v with a coefficient of -1, and in the equation for w with a coefficient of $+1$, so it has a coefficient of 0 in the sum. If $v \in R, w \notin R$, then x_{vw} occurs in the equation for w with a coefficient of 1, and so has coefficient 1 in the sum. If $v \notin R, w \in R$, then x_{vw} occurs in the sum with a coefficient of -1. So, the left hand side sums to $x(\delta(R)) - x(\delta(\overline{R}))$, as required. ■

Corollary

For any feasible rs-flow x and any rs-cut $\delta(R)$,

$$f_x(s) \leq u(\delta(R))$$

Proof. Using previous proposition, since $x(\delta(R)) \leq u(\delta(R))$ and $x(\delta(\overline{R})) \geq 0$. ■

Theorem (Max-Flow Min-Cut)

If there is a maximum rs-flow, then

$$\max\{f_x(s) : x \text{ is a feasible } rs\text{-flow}\} = \min\{u(\delta(R)) : \delta(R) \text{ is an } rs\text{-cut}\}$$

Proof. By previous corollary, we need only show that there exists a feasible flow x and a cut $\delta(R)$ such that $f_x(s) = u(\delta(R))$. Let x be a flow of maximum value. Let $R = \{v \in V : \text{there exists an } x\text{-augmenting } rv\text{-path}\}$. Clearly $r \in R$ and $s \notin R$, since there can be no x-augmenting path.

For every arc $vw \in \delta(R)$, we must have $x_{vw} = u_{vw}$, since otherwise adding vw to the x-augmenting rv-path would yield such a path to w, but $w \notin R$. Similar, for every arc $vw \in \delta(\overline{R})$, we have $x_{vw} = 0$. Then by proposition, $f_x(s) = x(\delta(R)) - x(\delta(\overline{R})) = u(\delta(R))$. ■

Theorem

A feasible flow x is maximum if and only if there is not x-augmenting path.

Proof. ($\Longrightarrow$) If x is maximum, there is no x-augmenting path.

($\Longleftarrow$) If there is no x-augmenting path, then the construction of the proof of Max-Flow Min-Cut yields a cut $\delta(R)$ with $f_x(s) = u(\delta(R))$, so x is maximum, by corollary. ■

Theorem

If u is integral and there exists a maximum flow, then there exists a maximum flow that is integral.

Proof. Choose an integral flow x of maximum value. If there is an x-augmenting path, then since x and u are integral, the new flow can be chosen integral, contradicting the choice of x. Hence there is no x-augmenting path, so x is a maximum flow, by previous theorem. ■

Corollary

If x is a feasible rs-flow and $\delta(R)$ is an rs-cut, then x is maximum and $\delta(R)$ is minimum if and only if $x_e = u_e$ for all $e \in \delta(R)$ and $x_e = 0$ for all $e \in \delta(\overline{R})$.

Proof. Combine Max-Flow Min-Cut theorem with the proof of corollary. ■

Part V

Matchings

Chapter 9

Matchings

Definition: Matching

A set $M \subseteq E$ such that no vertex of G is incident with more than one edge in M.

Definition: M-Covered

A vertex v is covered by M if some edge of M is incident with v.

Definition: M-Exposed

A vertex v is exposed if v is not M-covered.

The number of vertices covered by M is $2|M|$ and number of M-exposed vertices is $|V| - 2|M|$.

Definition: Maximum Matching

A matching of maximum cardinality, denoted $\nu(G)$.

Definition: Deficiency

The minimum number of exposed vertices for any matching of G, denoted by $\text{def}(G)$.

Note $\text{def}(G) = |V| - 2\nu(G)$.

Definition: Perfect Matching

A matching that covers all vertices.

Definition: Tight Odd Circuit

An odd circuit C is tight if

$$\nu(G) \geq \nu(G \times C) + \frac{|V(C)| - 1}{2}$$

holds with equality.

Definition: Inessential

A vertex v of G is inessential if there is a maximum matching of G not covering v.

Lemma

Let $G = (V, E)$ be a graph and let $vw \in E$. If v, w are both inessential, then there is a tight odd circuit C using vw.
Moreover, C is an inessential vertex of $G \times C$.

9.1 Bipartite Matching

Definition: Bipartite

$G = (V, E)$ is bipartite if $V = V_1 \cup V_2$, where V_1, V_2 disjoint and every edge has one end in V_1 and the other end in V_2.

Definition: Vertex Cover

A set $C \subseteq V$ such that every edge has at least one in C.

Lemma

If M is a matching and C is a cover, then $|M| \leq |C|$.

Proof. Every $e \in M$ has at least one end in C. No vertex in C meets more than one edge in M. ■

Definition: Minimum Cover

A cover of minimum cardinality, denoted $\tau(G)$.

Theorem (König)

If G is bipartite, $\nu(G) = \tau(G)$.

Proof. We note that $\nu(G) \leq \nu^*(G)$ and $\tau(G) \geq \tau^*(G)$. By using LP duality and the matching LP (*Matching LP*), we show that $\nu(G) = \nu^*(G)$. We also have the matching LP in the form of $Mx^+ = (1, \ldots, 1)^T$. Since M is totally unimodular, then M^T is also totally

unimodular. So the dual LP has all integral vertices, implying $\tau(G) = \tau^*(G)$. So,

$$\nu(G) = \nu^*(G) = \tau^*(G) = \tau(G)$$

■

9.2 Alternating Paths

Definition: M-Alternating

A path P is M-alternating if its edges are alternately in and not in M.

Definition: M-Augmenting

An M-alternating path P is M-augmenting if the ends of P are distinct and are both M-exposed.

Definition: Symmetric Difference

For sets S and T, let $S\Delta T$ denote the symmetric difference, which is defined as

$$S\Delta T = (S \cup T) \setminus (S \cap T)$$

Let a path P be an M-augmenting path. Then we can obtain a larger matching $M' = M\Delta E(P)$ with $|M'| = |M| + 1$.

Theorem (Petersen 1891, Berge 1957)

A matching M in a graph G is maximum if and only if there is no M-augmenting path.

Proof. ($\implies$) Suppose there exists an M-augmenting path P joining v and w. Then $N = M\Delta E(P)$ is a matching that covers all vertices covered by M, plus v and w. So, M is not maximum.

($\impliedby$) Conversely, suppose that M is not maximum and some other matching N satisfies $|N| > |M|$. Let $J = N\Delta M$. Each vertex of G is incident with at most two edges of J, so J is the edge set of some vertex disjoint paths and circuits of G. For each such path or circuit, the edges alternately belong to M or N. Therefore, all circuits are even and contain the same number of edges of M and N. Since $|N| > |M|$, there must be at least one path with more edges of N than M. This path is an M-augmenting path. ■

9.3 Matching LP

Definition: Matching LP

P is the set of solutions to

$$\begin{aligned} x(\delta(v)) &\leq 1, \ \forall v \in V \\ x_e &\geq 0, \ \forall e \in E \end{aligned}$$

Let $\overline{x}$ be a vertex of P. We show that $\overline{x}$ is integral, which implies that $M = \{e \in E : \overline{x}_e = 1\}$ is a matching and $\nu(G) = \nu^*(G)$.

Recall that for a polyhedron $P = \{x : Ax \leq b\} \subseteq \mathbb{R}^n$, $\overline{x} \in P$ is a vertex if and only if $\overline{x}$ is the unique solution to $A'x = b'$ for some subset of n inequalities $A'x \leq b'$ from $Ax \leq b$.

For our matching P, let $E^+ := \{e : \overline{x}_e > 0\}$ and $E^0 := \{x : \overline{x}_e = 0\}$. We write $\overline{x} = (\overline{x}^+, \overline{x}^0)$ split by (E^+, E^0).

Since $\overline{x}$ is a vertex, there exists $V^+ \subseteq V$ such that $\overline{x}$ is the unique solution to

$$\begin{aligned} \sum(x_e : e \in \delta(v) \cap E^+) &= 1, \ \forall v \in V^+ \\ x_e &= 0, \ \forall e \in E^0 \end{aligned}$$

Restricting to E^+, we can write the system of equations as

$$Mx^+ = (1, \ldots, 1)^T$$

By Cramer's Rule, the solution to the system is $(\overline{x}_1^+, \ldots, \overline{x}_k^+)$, where

$$\overline{x}_j^+ = \frac{\det(M^j)}{\det(M)}$$

with M^j obtained from M by replacing the jth column by $(1, \ldots, 1)^T$.

Claim: $\det(M) = 1$ or $\det(M) = -1$.

This gives that $\overline{x}_j^+$ is integer for all j, so $\overline{x}$ is integer. Thus, $\nu(G) = \nu^*(G)$.

Lemma

Let $G = (V, E)$ be a bipartite graph. Let A be the $|V| \times |E|$ matrix $\left[A_{ve}\right]$ with

$$A_{ve} = \begin{cases} 1 & \text{if } e \in \delta(v) \\ 0 & \text{if } e \notin \delta(v) \end{cases}$$

then A is totally unimodular.

Proof. By induction of the number of rows k of the submatrix B of A. If B is 1×1, then this is obvious.

Suppose it is true for $k = 1, \ldots, t-1$ and let B be a $t \times t$ submatrix of A.

1. If B has a column of all 0's, then $\det(B) = 0$.
2. If a column of B has exactly one 1, then we compute $\det(B)$ by expanding on that column and use induction.
3. Otherwise, every column of B has exactly two 1's.

We can partition the rows of B into W_1 and W_2, so that every column has exactly one 1 in W_1 and exactly one 1 in W_2 (W_1 are vertices in V_1, W_2 in V_2 from G being bipartite).

Now multiplying each row in W_1 by 1 and each row in W_2 by -1 and summing, we get the row vector of all 0's. So $\det(B) = 0$. ■

9.4 Tutte's Theorem

Let A be a subset of the vertices which $G - A$ has k components $H_1, \ldots, H_k$ having an odd number of vertices. Let M be a matching of G. For each i, either H_i has an M-exposed vertex or M contains an edge having just one end in $V(H_i)$. All such edges have their other ends in A and since M is a matching, all these ends must be distinct. Therefore, there can be at most $|A|$ edges and so the number of M-exposed vertices is at least $k - |A|$.

Definition: Odd Count $\mathrm{oc}(H)$

The number of components of H that contain an odd number of vertices.

Thus, for any $A \subseteq V$,

$$\nu(G) \leq \frac{1}{2}(|V| - \mathrm{oc}(G - A) + |A|)$$

If A is a cover of G, then there are $|V| - |A|$ odd components of $G - A$ (each is a single vertex), so the right hand side reduces to $|A|$. This bound is at least as strong as that provided by covers.

Theorem (Tutte-Berge Formula)

For a graph $G = (V, E)$, we have

$$\max\{|M| : M \text{ a matching}\} = \min\left\{\frac{1}{2}(|V| - \mathrm{oc}(G \setminus A) + |A|) : A \subseteq V\right\}$$

Theorem (Tutte's Matching Theorem 1947)

A graph $G = (V, E)$ has a perfect matching if and only if for all $A \subseteq V$, $\mathrm{oc}(G \setminus A) \leq |A|$.

If $\mathrm{oc}(G \setminus A) > |A|$, then A is called a Tutte set.

9.5 Maximum Matching

Maximum Matching Problem

Given a graph G, find a maximum matching of G.

Definition: Maximum Matching ILP

$$\begin{aligned} \max \quad & \sum(x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) \leq 1, \ \forall v \in V \\ & x_e \geq 0, \ \forall e \in E \\ & x_e \text{ integer}, \ \forall e \in E \end{aligned}$$

Definition: Maximum Matching LP Relaxation

$$\begin{aligned} \max \quad & \sum(x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) \leq 1, \ \forall v \in V \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Definition: Minimum Cover Dual LP

$$\begin{aligned} \min \quad & \sum(y_v : v \in V) \\ \text{s.t.} \quad & y_u + y_v \geq 1, \ \forall e = (u, v) \in E \\ & y_v \geq 0, \ \forall v \in V \end{aligned}$$

Let M be a matching and C be a cover, then

$$x_e^M = \begin{cases} 1 & \text{if } e \in M \\ 0 & \text{if } e \notin M \end{cases}, y_v^C = \begin{cases} 1 & \text{if } v \in C \\ 0 & \text{if } v \notin C \end{cases}$$

So, $\nu(G) \leq \nu^*(G)$ and $\tau(G) \geq \tau^*(G)$, and by LP duality, we have

$$\nu(G) \leq \nu^*(G) = \tau^*(G) \leq \tau(G)$$

9.6 Perfect Matching

9.6.1 Blossom Algorithm for Perfect Matching

Suppose we have a matching M of G and a fixed M-exposed vertex r of G. We can iteratively build up sets A, B of vertices such that each vertex in A is the other end of an odd-length M-alternating path beginning at r, and each vertex in B is the other end of an even-length M-alternating path beginning at r.

Begin with $A = \emptyset, B = \{r\}$, and use the rule: if $vw \in E, v \in B, w \notin A \cup B, wz \in M$, then add w to A, z to B. The set $A \cup B$ and edges in the construction form a tree T rooted at r.

Definition: M-Alternating Tree

A tree T such that

- every vertex of T other than r is covered by an edge of $M \cap E(T)$;
- for every vertex v of T, the path in T from v to r is M-alternating.

We let the vertex sets at odd and even distances from the root as $A(T)$ and $B(T)$ respectively. Note that $|B(T)| = |A(T)| + 1$ since all other vertices other than r come in matched pairs, one in $A(T)$ and one in $B(T)$.

Algorithm Sketch: Given a matching M, if M is not perfect, search for an augmenting path P. Recall that $M' = M \Delta E(P)$ gives us a larger matching since $|M'| = |M| + 1$.

If the algorithm does not find an augmenting path, then we need to certify that G has no perfect matching. We can use Tutte's Matching Theorem to find a Tutte set A where $\text{oc}(G \setminus A) > |A|$.

Let $r \in V$ be M-exposed. Grow an M-alternating tree T.

Choose an edge $vw \in E$ with $v \in B(T)$ and $w \notin A(T)$.

Case 1: w is M-exposed.
We have an augmenting path from r to w, so we can augment this path to get a larger matching. We reset T since r is now M-covered.

Case 2: $w \notin V(T)$ and w is M-covered.
We can grow T by adding vw and the edge $e \in M$ having w as an end.

Case 3: $w \in B(T)$.
Let C be the circuit formed from vw and the path in T joining v and w. $|C|$ is odd since vertices in $B(T)$ are at even distances from r.

Note: $|M \cap E(C)| = \frac{|C|-1}{2}$ so M is a near-perfect matching of C.

Definition: Blossom

Let $v, w \in B(T)$ and $vw \in E(G)$. The odd circuit in $T + vw$ is a blossom.

Definition: Shrink

Let C be an odd circuit in G. Define $G' = G \times C$ as the subgraph obtained from G by shrinking C; G' has vertex set $(V - V(C)) \cup \{C\}$ and edge set $E \setminus \gamma(V(C))$.

Definition: Pseudonode

The vertex after shrinking a blossom.

Let $G' \times C$ denote the graph obtained by shrinking C. The near-perfect matching allows us to un-shrink an augmenting path. Note that the pseudonode is in $B(T)$.

Definition: Frustrated

An M-alternating tree T in a graph G is frustrated if every edge of G has one end in $B(T)$ and the other end in $A(T)$.

Definition: $S(v)$

Given a vertex v of G', there corresponds a set $S(v)$ of vertices of G, where

$$S(v) = \begin{cases} v & \text{if } v \in V(G) \\ \bigcup_{w \in V(C)} S(w) & \text{if } v = C \text{ is a pseudonode} \end{cases}$$

Lemma

Let M' be a matching of G' and let T be an M'-alternating tree of G' such that no element of $A(T)$ is a pseudonode. If T is frustrated, then G has no perfect matching.

Proof. When we delete $A(T)$ from G', we get a component with vertex set $S(v)$ for each $v \in B(T)$. By construction, $|S(v)|$ is odd since it is the union of an odd number of vertices and pseudonodes u each having $|S(u)|$ odd. Since $|B(T)| = |A(T)| + 1$, then $\text{oc}(G \setminus A(T)) > |A(T)|$. $A(T)$ is therefore a Tutte set. ■

Algorithm 5 Blossom Algorithm (Perfect Matching)

Input: Graph G and matching M of G
Choose an M-exposed vertex r
$T = (\{r\}, \emptyset)$
while there exists $vw \in E$ with $v \in B(T), w \notin A(T)$ **do**
    Case: w is M'-exposed
        Let P be the augmenting path from r to w, $M = M \Delta E(P)$
        if there is no M-exposed vertex in G **then**
            return Perfect matching M
        else
            $T = (\{r\}, \emptyset)$, where r is M-exposed
    Case: $w \notin V(T)$, w is M-covered
        Let $wz \in M$ and $z \notin V(T)$, $E(T) = E(T) \cup \{vw, wz\}$
    Case: $w \in B(T)$
        Let C be the circuit formed from vw and the vw-path in T
        $G = G \times C$ // Shrink C
return G has no perfect matching, $A(T)$ is the Tutte set

Theorem

The Blossom Algorithm terminates after $O(n)$ augmentations, $O(n^2)$ shrinking steps, and $O(n^2)$ tree-extension steps.
Moreover, it determines correctly whether G has a perfect matching.

9.7 Blossom Algorithm for Maximum Matching

We can extend the Blossom algorithm for perfect matchings to maximum matchings.

Algorithm 6 Blossom Algorithm (Maximum Matching)

1: **Input**: Graph G and matching M of G
2: $\mathcal{T} = \emptyset$
3: Choose an M-exposed vertex r
4: $T = (\{r\}, \emptyset)$
5: **while** there exists $vw \in E$ with $v \in B(T), w \notin A(T)$ **do**
6: **Case**: w is M-exposed
7: Let P be the augmenting path from r to w, $M = M \Delta E(P)$
8: **if** there is no M-exposed vertex **then**
9: **return** Perfect matching M
10: **else**
11: $T = (\{r\}, \emptyset)$, where r is M-exposed
12: **Case**: $w \notin V(T)$, w is M-covered
13: Let $wz \in M$ and $z \notin V(T)$, $E(T) = E(T) \cup \{vw, wz\}$
14: **Case**: $w \in B(T)$
15: Let C be the circuit formed from vw and the vw-path in T
16: $G = G \times C$ // Shrink C
17: $\mathcal{T} = \mathcal{T} \cup \{T\}, G = G \setminus V(T), M = M \setminus E(T)$
18: **if** there exists an M-exposed vertex **then**
19: Go to line 5
20: Restore the matching M
21: **return** M

Theorem

The Blossom Algorithm can be implemented to run in time $O(nm \log n)$.

Chapter 10

Weighted Matching

10.1 Minimum-Cost Perfect Matching

10.1.1 Perfect Matching Polytope

By Minkowski's theorem, we know that there is a system of inequalities for the convex hull of a set, but we typically do not know what the system is. However, Edmonds founded the matching polytope theory in the 1960s.

The integer program for the minimum-cost perfect matching problem is

$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) = 1, \ \forall v \in V \\ & x_e \geq 0, \ \forall e \in E \\ & x_e \text{ integer}, \ \forall e \in E \end{aligned}$$

If G is bipartite, then we do not need the integrality constraint. If G is non-bipartite, then there exists a circuit C with $|E(C)|$ odd.

For a general graph, the original minimum-weight perfect matching LP can achieve an optimal solution, which may be a fractional vertex of the polyhedron defined by the LP. This is caused by the odd circuits in the graph.

Definition: Perfect Matching Polytope

$$\mathcal{PM}(G) = conv.hull(\{x^M : M \text{ a perfect matching}\})$$

Edmonds' Perfect Matching Polytope Theorem

Let $S \subseteq V$ with $|S|$ odd and $|S| \geq 3$. For an undirected graph, every perfect matching must contain at least one edge in $\delta(S)$, so we arrive at the blossom inequality constraint.

Definition: Blossom Inequality

For $S \subseteq V$ with $|S|$ odd and $|S| \geq 3$,

$$x(\delta(S)) \geq 1$$

Theorem (Perfect Matching Polytope Theorem – Edmonds)

For any graph $G = (V, E)$, $\mathcal{PM}(G)$ is the solution set of the linear system

$$\begin{aligned} x(\delta(v)) &= 1, \ \forall v \in V \\ x(\delta(S)) &\geq 1, \ \forall S \subseteq V, |S| \text{ odd}, |S| \geq 3 \\ x_e &\geq 0, \ \forall e \in E \end{aligned}$$

Proof. (Schrijver) Let Q denote the solution set of the linear system. Clearly $\mathcal{PM}(G) \subseteq Q$. Suppose $Q \not\subseteq \mathcal{PM}(G)$ and let x be a vertex of Q with $x \notin \mathcal{PM}(G)$. Choose this counterexample G such that $|V| + |E|$ is as small as possible.

Claim 1: $0 < x_e < 1$ for all $e \in E$.
Proof. (**Claim 1**) Otherwise, if $x_e = 0$, then delete e. If $x_e = 1$, then delete e and the ends of e. ■

So each vertex of G has degree at least 2, which implies $|E| \geq |V|$.

Claim 2: $|E| > |V|$.
Proof. (**Claim 2**) If $|E| = |V|$, then G is a circuit and the theorem is true. ■

Since x is a vertex of Q, there are $|E|$ constraints of the linear system satisfied as an equation by x. Thus, there exists an odd $S \subseteq V$ with $3 \leq |S| \leq |V| - 3$ and $x(\delta(S)) = 1$.

Let G' be the graph obtained by shrinking S to a single vertex and G'' be obtained by shrinking $V \setminus S$ to a single vertex. Let x' and x'' be obtained by shrinking x to G' and G'' respectively. So x' and x'' satisfy the linear system for G' and G''. By induction, $x' \in \mathcal{PM}(G')$ and $x'' \in \mathcal{PM}(G'')$.

Since x is rational, x' and x'' are rational convex combinations of perfect matchings in G' and G'', i.e.

$$x' = \frac{1}{k} \sum_{i=1}^{k} x^{M_i'}, x'' = \frac{1}{k} \sum_{i=1}^{k} x^{M_i''}$$

for some k (the common denominator of the multipliers λ_i' and λ_i'' in the convex combinations).

For each edge $e \in \delta(S)$, the number of indices i with $e \in M_i'$ is $kx_e' = kx_e = kx_e''$ which is equal to the number of indices i with $e \in M_i''$.

We may assume that for each i, the two matchings M_i' and M_i'' have an edge in $\delta(S)$ in

common. So $M_i = M_i' \cup M_i''$ is a perfect matching of G and

$$x = \frac{1}{k} \sum_{i=1}^{k} x^{M_i}$$

and thus $x \in \mathcal{PM}(G)$, a contradiction. ■

10.1.2 Linear Programming

Definition: ODD

$$\text{ODD} = \{S \subseteq V : |S| \text{ odd}, |S| \geq 3, |S| \leq |V| - 3\}$$

Definition: Minimum-Cost Perfect Matching LP

$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) = 1, \ \forall v \in V \\ & x(\delta(S)) \geq 1, \ \forall S \in \text{ODD} \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Definition: Minimum-Cost Perfect Matching Dual LP

$$\begin{aligned} \max \quad & \sum(y_v : v \in V) + \sum(Y_S : S \in \text{ODD}) \\ \text{s.t.} \quad & y_v + y_w + \sum(Y_S : e \in \delta(S), S \in \text{ODD}) \leq c_e, \ \forall e = vw \in E \\ & Y_S \geq 0, \ \forall S \in \text{ODD} \end{aligned}$$

Definition: Reduced Cost

Given a dual solution (y, Y), the reduced cost of an edge $e = vw$ is

$$\bar{c}_e = c_e - y_v - y_w - \sum(Y_S : e \in \delta(S), S \in \text{ODD})$$

Definition: Complementary Slackness Conditions

Let $x = (x_e : e \in E), y = (y_v : v \in V), Y = (Y_S : S \in \text{ODD})$.

(i) If $x_e > 0$, then $\bar{c}_e = 0$.

(ii) If $Y_S > 0$, then $x(\delta(S)) = 1$.

In other words, we only want edges having reduced cost 0 and only use dual variables Y_S if $\delta(S)$ contains exactly one matching edge.

10.1.3 Blossom Algorithm

Definition: Equality Subgraph $E^=$

$$E^= = \{e \in E : \bar{c}_e = 0\}$$

We need a perfect matching in $E^=$ such that if $Y_S \geq 0$, then $|\delta(S) \cap M| = 1$.

Algorithm Sketch: Grow an alternating tree using only $E^=$ edges, starting at an M-exposed vertex. Unlike the unweighted algorithm, we need to keep the shrunk graph from iteration to iteration, because of the CS conditions on the Y_S variables. Similar to the unweighted version, we choose an edge $e = vw$ with $v \in B(T), w \notin A(T)$. If none exist, we will possibly change the dual solution to create new edges in $E^=$.

Algorithm 7 Blossom Algorithm (Minimum-Cost Perfect Matching)

Input: Graph G
$M = \emptyset, Y_S = 0 : \forall S \in \text{ODD}, y_v = 0 : \forall v \in V$
$T = (\{r\}, \emptyset)$, where r is an M-exposed vertex
while true **do**
    Case: $e = vw \in E^=$ with $v \in B(T), w \notin A(T)$
        Case: w is M-exposed
            Let P be the augmenting path from r to w, $M = M \Delta E(P)$
            if there is no M-exposed vertex **then**
                return Perfect matching M
            else
                $T = (\{r\}, \emptyset)$, where r is M-exposed.
        Case: $w \notin V(T)$, w is M-covered
            Let $wz \in M$ and $z \notin V(T)$, $E(T) = E(T) \cup \{vw, wz\}$
        Case: $w \in B(T)$
            Let C be the circuit formed from vw and the vw-path in T
            $G = G \times C$ // Shrink C
    Case: No edge $e = vw$ available
        $\varepsilon_1 = \min(\overline{c}_e : e$ joins a vertex in $B(T)$ to a vertex not in $V(T))$ // next iteration will augment M or grow T
        $\varepsilon_2 = \min(\overline{c}_e/2 : e$ joins two vertices in $B(T))$ // next iteration will shrink the odd circuit
        $\varepsilon_3 = \min(y_v : v \in A(T), v$ a pseudonode$)$
        $\varepsilon = \min(\varepsilon_1, \varepsilon_2, \varepsilon_3)$
        if $\varepsilon = \infty$ **then**
            return G has no perfect matching; there exists a Tutte set
        $y_v = y_v + \varepsilon$ if $v \in B(T)$ // y_v may be a pseudonode, so y_v is actually Y_S
        $y_v = y_v - \varepsilon$ if $v \in A(T)$ // y_v may be a pseudonode, so y_v is actually Y_S
        if $\varepsilon = \varepsilon_3$ **then**
            Unshrink the pseudonode $v \in A(T)$ that attained the minimum
            Let s, t be the vertices adjacent to v
            Keep the even-length st-path in T

The Blossom Algorithm for Minimum-Cost Perfect Matching above has runtime complexity $O(mn^2)$. This can be improved to $O(n^3)$.

10.2 Maximum-Weight Matching

Definition: Odd-Cut

The set $\delta(S)$ where $|S|$ is odd.

Perfect Matching Separation Problem

Given $x = (x_e : e \in E)$, find an odd-set S such that $x(\delta(S)) < 1$ if such a set exists.

First we set $\overline{x}_e = w_e$ for all $e \in E$. We find a minimum-weight cut in G, i.e. $\min \sum(w_e : e \in \delta(S))$ over all proper subsets $S \subseteq V$. This can be done by solving $n - 1$ max-flow min-cut problems (or other methods).

If $|S|$ is odd, then $\delta(S)$ is a minimum-weight odd-cut in G. We check if the weight < 1. If yes, then it is a separating inequality. If no, then no separating inequality exists.

If $|S|$ is even, show there exists a minimum-weight odd-cut $\delta(U)$ with $U \subseteq S$ or $U \subseteq V \setminus S$. We can solve the problem on two smaller by having two graphs, one shrinking S and the other shrinking $V \setminus S$. We call the shrunk vertex even and original vertices odd. Thus, we look for U with an odd number of odd vertices. This gives a polynomial time algorithm, but it is not efficient for large graphs. Faster algorithms use Gomory-Hu trees (Padberg & Rao).

10.2.1 Matching Polytope

Definition: Matching Polytope

$$\mathcal{M}(G) = conv.hull(\{x^M : M \text{ a matching}\})$$

Consider the maximum-weight matching LP,

$$\begin{aligned} \max \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) \leq 1, \ \forall v \in V \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

If we had a triangle graph with $c_e = 1$ for all $e \in E$, then we can get an optimal solution of $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$ with value $\frac{3}{2}$, but the optimal matching is only 1 with one edge. So we need to add inequalities.

Definition: $\gamma(S)$

For $S \subseteq V$,

$$\gamma(S) = \{vw : vw \in E, v \in S, w \in S\}$$

If $|S|$ is odd, every matching x^M satisfies $x(\delta(S)) \leq \frac{|S|-1}{2}$. This is the blossom inequality for matchings.

Note that

$$x(\gamma(S)) = \frac{1}{2} \sum(x(\delta(v)) : v \in S) - x(\delta(S))$$

Thus, for perfect matchings, we have

$$\begin{aligned} x(\delta(S)) \geq 1 &\iff -x(\gamma(S)) + \frac{1}{2}\sum(x(\delta(v)) : v \in S) \geq 1 \\ &\iff -x(\gamma(S)) + \frac{1}{2}|S| \geq 1 \\ &\iff x(\gamma(S)) \leq \frac{1}{2}|S| - 1 = \frac{|S|-1}{2} \end{aligned}$$

So for perfect matchings, the two forms of the blossom inequality are equivalent.

Definition: Blossom Inequality

For $S \subseteq V$ with $|S|$ odd and $|S| \geq 3$,

$$x(\gamma(S)) \leq \frac{|S|-1}{2}$$

Theorem (Edmonds)

$\mathcal{M}(G)$ is defined by

$$\begin{aligned} x(\delta(v)) &\leq 1, \ \forall v \in V \\ x(\gamma(S)) &\leq \frac{|S|-1}{2}, \ \forall S \in \text{ODD} \\ x_e &\geq 0, \ \forall e \in E \end{aligned}$$

Proof. *Sketch* (Schrijver) Let Q be the solution set of the inequality system. We have $\mathcal{M}(G) \subseteq Q$.

Let $x \in Q$. We must show $x \in \mathcal{M}(G)$. Let G' be a copy of the graph G. Form a graph $\tilde{G} = G \cup G'$ with additional edges (v, v') for all $v \in V$. Set $\tilde{x}_e = \tilde{x}_{e'} = x_e$ for all $e \in E$ and $\tilde{x}_{vv'} = 1 - x(\delta(v))$ for all $v \in V$.

Claim: $\tilde{x} \in \mathcal{PM}(\tilde{G})$.

The idea of the proof is to show $\tilde{x}$ satisfies each blossom constraint (the $\leq$ form). Then the claim implies $x \in \mathcal{M}(G)$. ■

The dual LP of the maximum-weight matching is

$$\begin{aligned} \min \quad & \sum(y_v : v \in V) + \sum\left(\left(\frac{|S|-1}{2}\right) Y_S : S \in \text{ODD}\right) \\ \text{s.t.} \quad & y_v + y_w + \sum(Y_S : e \in \gamma(S), S \in \text{ODD}) \geq c_e \\ & y_v \geq 0, \ \forall v \in V \\ & Y_S \geq 0, \ \forall S \in \text{ODD} \end{aligned}$$

Recall that in the Blossom algorithm, the ε used in the dual change includes $\varepsilon_1, \varepsilon_2, \varepsilon_3$. ε_2 may introduce fractional values for the dual variables, even when c_e is integer for all $e \in E$.

If c_e is integer for all $e \in E$, then the Blossom algorithm will return a dual solution (y, Y) such that Y is integer and y is $\frac{1}{2}$-integer.

Theorem (Cunningham & Marsh 1978)

For the matching LP, if $(c_e : e \in E)$ are integers, then there exists an optimal dual solution (y, Y) that is integer.

This theorem says the blossom system for matchings is totally dual integral.

Chapter 11

General Matching

By a sequence of reductions, Edmonds' results can be extended to a general matching setting. For a graph $G = (V, E)$, let M be a $|V| \times |E|$ matrix such that all entries are $0, \pm 1, \pm 2$ and the sum of absolute values of entries in each column is at most 2.

For the problem

$$\begin{aligned} \max \quad & \sum(w_e x_e : e \in E) \\ \text{s.t.} \quad & b_1 \leq Mx \leq b_2 \\ & c_1 \leq x \leq c_2 \\ & x_e \text{ integer}, \ \forall e \in E \end{aligned}$$

Edmonds and Johnson described the polyhedron and an algorithm.

Definition: 2-Factor

A set $F \subseteq E$ such that every $v \in V$ meets exactly 2 edges in F.

A 2-factor is a set of disjoint circuits covering all vertices. 2-factors are a relaxation of the TSP. In TSP, we want a single circuit covering all vertices.

$$\begin{aligned} & x(\delta(v)) = 2, \ \forall v \in V \\ & 0 \leq x_e \leq 1, \ \forall e \in E \\ & x_e \text{ integer}, \ \forall e \in E \end{aligned}$$

To describe the 2-factor polytope, we need a version of the blossom inequalities.

Definition: Blossom Inequality for 2-Factors

Let $U \subseteq V, F \subseteq \delta(U)$ where F is a matching and $|F|$ is odd, then every 2-factor satisfies

$$x(\delta(U) \setminus F) - x(F) \geq 1 - |F|$$

Adding all of these inequalities gives the convex hull of 2-factors.

Chapter 12

T-Joins

Definition: T-Join

Let $G = (V, E)$ be a graph and $T \subseteq V$ with $|T|$ even. A T-join is a set $J \subseteq E$ such that

$$|J \cap \delta(v)| \text{ is odd} \iff v \in T$$

In other words, J is a T-join if and only if the odd-degree vertices of the subgraph (V, J) are exactly the elements of T.

Optimal T-Join Problem

Given a graph $G = (V, E)$, a set $T \subseteq V$ such that $|T|$ is even, and a cost vector $c \in \mathbb{R}^E$, find a T-join J of G such that $c(J)$ is minimum.

Every T-join is the edge-disjoint union of circuits and $\frac{1}{2}|T|$ paths connecting pairs of vertices in T.

If $c_e \geq 0$ for all $e \in E$, then there exists an optimal T-join that consists of $\frac{1}{2}|T|$ paths connected pairs in T.

Special cases:

- Perfect matchings, let $T = V$.
- rs-shortest path, let $T = \{r, s\}$.

12.1 Chinese Postman Problem

A main motivation is the Chinese Postman Problem.

Chinese Postman Problem (CPP)

Given a connected graph $G = (V, E)$, edge costs $(c_e : e \in E), c_e \geq 0, \forall e \in E$, find a minimum-cost closed path that uses each edge of G at least once.

Definition: Euler Tour

A closed edge-simple path P with $E(P) = E(G)$.

Theorem

A connected graph G has an Euler tour if and only if every vertex of G has even degree.

If G has an Euler tour T, then T is an optimal solution to the CPP. If G does not have an Euler tour, then we must use some edges more than once. By the above theorem, we never need to use any edge more than twice.

Thus, to solve the CPP, we want to find a minimum-cost set of edges $J \subseteq E$ such that duplicating every edge in J gives a graph with every vertex of even degree.

12.2 Optimal T-Join Algorithm

To solve the CPP, we let $T = \{v \in V : \deg(v) \text{ odd}\}$. Edmonds gives a polynomial time algorithm by reducing the problem to a perfect matching.

We can assume $c_e \geq 0$ for all $e \in E$. To see this, let $N = \{e : c_e < 0\}$ and let $U = \{v : v \text{ has odd degree in } (V, N)\}$, $T' = T \Delta U$ and $c'_e = |c_e|$ for all $e \in E$. Now let J' be a minimum c'-weight T-join.

Claim: $J = J' \Delta N$ is a minimum c-weight T-join.

We can now reduce the problem to a perfect matching problem (Edmonds). Consider the complete graph K_T with vertex set T. For each edge $uv \in E(K_T)$, let ℓ_{uv} be the length of a shortest uv-path in G and let P_{uv} denote the path.

Using ℓ_{uv} as edge costs, find a minimum-cost perfect matching in K_T. Let M denote the optimal matching. Then the symmetric difference of the paths P_{uv} for all $uv \in M$ (keep the edges that appear an odd number of times in the collection) is a minimum-cost T-join.

Notes:

1. By setting $T = \emptyset$, we get a polynomial time algorithm to check for a negative-cost circuit in G.
2. If no negative-cost circuit, setting $T = \{r, s\}$, we get a polynomial time rs-path algorithm for undirected graphs.

3. We get a polynomial time algorithm for the CPP.

Algorithm 8 Optimal T-Join Algorithm

1: Let $N = \{e : c_e < 0\}$ and $U = \{v : v \text{ has odd degree in } (V, N)\}$
2: $c = |c|, T = T \Delta U$
3: Let K_T be the complete graph on vertex set T
4: For each $uv \in E(K_T)$, give uv edge cost $\ell_{uv} =$ cost of shortest uv-path in G
5: Find a minimum-weight perfect matching M in K_T
6: Let J be the symmetric difference of $E(P_{uv})$ for $uv \in M$ (keep edges that appear an odd number of times)
7: $J = J \Delta N$
8: **return** J

12.3 Linear Programming

Definition: T-Odd

A set $S \subseteq V$ is T-odd if $|S \cap T|$ is odd.

If $J \subseteq E$ is a T-join, then we must have $J \cap \delta(S) \neq \emptyset$ if S is T-odd.

Definition: T-Cut

The set $\delta(S)$ where $S \subseteq V$ is T-odd.

For a T-cut $\delta(S)$, every T-join satisfies $x(\delta(S)) \geq 1$.

Definition: T-Join LP

$$\begin{aligned} \min \quad & \sum (c_e x_e : e \in E) \\ \text{s.t.} \quad & x(D) \geq 1, \ \forall \ T\text{-cuts } D \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

If $c_e < 0$ for some $e \in E$ and G has a T-join, then the LP is unbounded.

Theorem (Edmonds & Johnson)

If $c_e \geq 0$ for all $e \in E$, then the minimum cost of a T-join is equal to the optimal value of the T-join LP.

Geometry: If $P = conv.hull(\{x^J : J \text{ a } T\text{-join}\})$, then the polyhedron defined by

$$\begin{aligned} x(D) &\geq 1, \ \forall \ T\text{-cuts } D \\ x &\geq 0 \end{aligned}$$

is $P + \mathbb{R}^n_+$ is the **dominant** of P, i.e. everything above P.

Definition: T-Join Dual LP

$$\begin{aligned} \max \quad & \sum(y_D : D \text{ a } T\text{-cut}) \\ \text{s.t.} \quad & \sum(y_D : e \in D, D \text{ a } T\text{-cut}) \leq c_e, \ \forall e \in E \\ & y_D \geq 0, \ \forall \ T\text{-cuts } D \end{aligned}$$

If $c_e = 1$ for all $e \in E$, then an integer dual solution gives a collection of edge-disjoint T-cuts, called a **packing** of T-cuts.

Research problem: For which (G, T) is the minimum cardinality of a T-join equal to the maximum packing of T-cuts? This is true if G is bipartite. Abdi and Guenin has results when $|T| \leq 4$.

Notes:

1. If J is a T-join and D is a T-cut, then $J \cap D \neq \emptyset$.
2. If $C \subseteq E$ is such that $C \cap J \neq \emptyset$ for all T-joins J, then C contains a T-cut.

   Proof. Consider the graph $G' = (V, E \setminus C)$. Each connected component of G' has an even number of vertices in T. So there is a T-join disjoint from C. ■
3. Minimal T-cuts are exactly the minimal edge sets that intersect all T-joins.
4. Minimal T-joins are exactly the edge sets that intersect all T-cuts.

12.4 Clutters

Definition: Clutter

A family of subsets $\mathcal{C}$ is called a clutter if for all $A, B \in \mathcal{C}$, we have $A \not\subset B$ and $B \not\subset A$.

Packing problem: Find the maximum number of disjoint members of $\mathcal{C}$.

Definition: Blocker

The blocker of $\mathcal{C}$ is the family of minimal sets that intersect all members of $\mathcal{C}$.

Theorem (Edmonds & Fulkerson)

blocker(blocker(e))=e.

Part VI

Matroids

Chapter 13

Matroid Theory

Matrix Example: Let A be a matrix in $\mathbb{R}^{m\times n}$ and let E be the column indices of A. We call $I \subseteq E$ independent if the column vectors of I are linearly independent in $\mathbb{R}^m$.

(0) $\emptyset$ is independent.

(1) If I is independent, then so is any subset of I.

(2) For any $X \subseteq E$, any two maximal independent sets in X have the same size. The size is equal to the column rank of $(A_e : e \in X)$.

A standard proof of (2) is for $X \subseteq E$,

(i) $I \subseteq X$ is independent, then $|I| \leq \text{rank}(X)$.

(ii) $I \subseteq X$ is independent and $|I| < \text{rank}(X)$, then there exists a column index $e \in X \setminus I$ such that $I \cup \{e\}$ is independent.

Graph Example: Let $G = (V, E)$ be a graph. Call $I \subseteq E$ independent if it contains no circuit.

(0) $\emptyset$ is independent.

(1) If I is independent, then so is any subset of I.

Note that I is independent if it induces a forest. So any independent set has size at most $|V| - \kappa(G)$.

(2) For any $X \subseteq E$, any two maximal independent sets in X have the same size.

13.1 Matroids

Definition: Matroid

Let E be a finite set (*ground set*) and $\mathcal{I}$ be a family of subsets of E (*independent sets*). A matroid M is a pair $(E, \mathcal{I})$ such that

(M0) $\emptyset \in \mathcal{I}$ ($\emptyset$ is independent).

(M1) If $J' \subseteq J \in \mathcal{I}$, then $J' \in \mathcal{I}$.

(M2) For all $X \subseteq E$, every maximal independent subset contained in X has the same cardinality.

Definition: Basis

An maximal independent set of a set $X \subseteq E$.

Definition: Rank

The constant size in (M2) is called the rank of X, denoted by $r(X)$.

An independent set of size $r(E)$ is a basis of the matroid M.

Definition: Dependent

A subset that is not independent.

Definition: Circuit

A minimal dependent set.

Waterloo is a center for matroid research. Tutte built the field. Matroid optimization was developed by Edmonds. Cunningham worked on matroid structure.

Matroid Optimization Problem

Let $M = (E, \mathcal{I})$ be a matroid and $c = (c_e : e \in E)$. Find a maximum-weight independent set $I \in \mathcal{I}$.

Algorithm 9 Greedy Algorithm

```
J = ∅
while there exists e ∈ E \ J with c_e > 0 and J ∪ {e} ∈ I do
    Choose e with c_e maximum
    J = J ∪ {e}
return J
```

Theorem

For any matroid M, the greedy algorithm produces a maximum-weight independent set.

Proof. Suppose not. Let $J \in \mathcal{I}$ be a set produced by greedy and let $J' \in \mathcal{I}$ be such that $c(J') > c(J)$.

Let $J = \{e_1, \ldots, e_m\}$ where the elements are chosen by greedy in the order $e_1, \ldots, e_m$. Note that $c_{e_1} \geq \cdots \geq c_{e_m}$. Let $J' = \{f_1, \ldots, f_\ell\}$ where $c_{f_1} \geq \cdots \geq c_{f_\ell}$.

Let k be the smallest index such that $c_{f_k} > c_{e_k}$. If none exist, then $\ell > m$ and in this case set $k = m + 1$.

In step k, the greedy algorithm did not add one of $f_1, \ldots, f_k$. Now since $c_{e_k} < c_{f_k}$, we have

$$\begin{gathered}
\{e_1, \ldots, e_{k-1}\} \cup \{f_1\} \notin \mathcal{I} \text{ or } f_1 \in \{e_1, \ldots, e_{k-1}\} \\
\{e_1, \ldots, e_{k-1}\} \cup \{f_2\} \notin \mathcal{I} \text{ or } f_2 \in \{e_1, \ldots, e_{k-1}\} \\
\vdots \\
\{e_1, \ldots, e_{k-1}\} \cup \{f_k\} \notin \mathcal{I} \text{ or } f_k \in \{e_1, \ldots, e_{k-1}\}
\end{gathered}$$

Thus, $\{e_1, \ldots, e_{k-1}\}$ is a maximal independent set in

$$X = \{e_1, \ldots, e_{k-1}\} \cup \{f_1, \ldots, f_k\}$$

So, by (M2), every independent set in X has size at most $k - 1$. But the subset $\{f_1, \ldots, f_k\}$ of J' has size k, a contradiction. ■

Definition: Independence System

$(E, \mathcal{I})$ satisfying (M0) and (M1).

Theorem

An independence system is a matroid if and only if for all $c = (c_e : e \in E)$, the greedy algorithm finds a maximum weight independent set.

Proof. If $(E, \mathcal{I})$ is not a matroid, let X be a set that violates (M2). Let

$$c_e = \begin{cases} 1 & \text{if } e \in X \\ 0 & \text{if } e \notin X \end{cases}$$

There exists a maximal independent set $I \subseteq X$ that is not of maximum cardinality. But I could potentially be a solution produced by the greedy algorithm. ■

Definition: Matroid Oracle

The call to test $J \subseteq E$ is an independent set.

This algorithm is polynomial time if it is polynomial in $|E|$, and the bits to express $(c_e : e \in E)$.

Definition: Submodularity

Let $M = (E, \mathcal{I})$ be a matroid with rank function r. Then for all $X, Y \subseteq E$, we have

$$r(X) + r(Y) \geq r(X \cap Y) + r(X \cup Y)$$

Proof. Choose $A \subseteq X \cap Y, B \subseteq X \setminus Y, C \subseteq Y \setminus X$ as follows.

- A is a maximal independent set in $X \cap Y$.
- $A \cup B$ is a maximal independent set in X.
- $A \cup B \cup C$ is a maximal independent set in $X \cup Y$.

Then,

$$\begin{aligned} r(X \cap Y) + r(X \cup Y) &= |A| + |A \cup B \cup C| \\ &= (|A| + |B|) + (|A| + |C|) \\ &= r(X) + (|A| + |C|) \\ &\leq r(X) + r(Y) \end{aligned}$$

■

13.2 Examples of Matroids

Definition: Forest Matroid

Let $G = (V, E)$ be a graph and let E be the finite ground set. If $\mathcal{I} = \{I \subseteq E : I \text{ contains no circuit}\}$, then $(E, \mathcal{I})$ is a forest matroid.

Definition: Uniform Matroid

Let E be a finite set with $|E| = n$ and choose an integer $k \in \{0, \ldots, n\}$. Call a subset of E independent if it has size at most k and let $\mathcal{I}$ be the collection of all independent sets. Let $U_n^k = (E, \mathcal{I})$, then U_n^k is a uniform matroid.

Proof. (M0) and (M1) hold. For any $X \subseteq E$, any maximal independent set contained in X has size $\min(k, |X|) = r(X)$. ■

Definition: Partition Matroid

Let E be a finite set and let $E_1, \ldots, E_k$ be a partition of E. Call $I \subseteq E$ independent if I picks at most one element from each partition. That is, for $i = 1, \ldots, k$, we have $|I \cap E_i| \leq 1$. Let $\mathcal{I}$ be the collection of all independent sets. Then $(E, \mathcal{I})$ is a partition matroid.

Proof. $M = (E, \mathcal{I})$ is a matroid. (M0) and (M1) hold. Suppose $X \subseteq E$. Any maximal independent set in X has size $|\{i : X \cap E_i \neq \emptyset\}| = r(X)$. ■

13.3 Maximum-Weight Basis

If $c_e > 0$ for all $e \in E$, then a maximum-weight independent set is a basis.

If some of the weights are negative, this may not be the case.

To find a maximum-weight independent basis, modify the greedy algorithm to accept negative cost elements. Alternatively, we could add a large constant Δ to all c_e.

13.4 Matroid Duality

Suppose we have a connected plane graph $G = (V, E)$. Let G^* be the planar dual of G, i.e. all faces become vertices, vertices become faces, and if two faces were adjacent, then the vertices corresponding to the faces are adjacent.

If $T \subseteq E$ is the edge set of a spanning tree of G, then $E \setminus T$ gives a spanning tree of G^*.

Thus, an independent set in the forest matroid of G^* corresponds to a set $F \subseteq E$ such that $E \setminus F$ contains a spanning tree.

Definition: Dual Matroid

For a matroid $M = (E, \mathcal{I})$ with rank function r, its dual matroid is $M^* = (E, \mathcal{I}^*)$ where

$$\mathcal{I} = \{J \subseteq E : r(E \setminus J) = r(E)\}$$

That is, J is independent in M^* if $E \setminus J$ contains a basis of M.

Theorem

The dual M^* is a matroid.

Proof. (M0) holds. Let $J \subseteq I \subseteq E$. Then

$$r(E) \geq r(E \setminus J) \geq r(E \setminus I)$$

So if $I \subseteq \mathcal{I}^*$, then $J \in \mathcal{I}^*$. Thus, (M1) holds.

Let $A \subseteq E$ and let J be a maximal member of $\mathcal{I}^*$ contained in A. Let B be a maximal member of $\mathcal{I}$ contained in $E \setminus A$. Now extend B to a maximal member of $\mathcal{I}$ contained in $E \setminus J$. Since $J \in \mathcal{I}^*$, B' is a basis of M, so $|B'| = r(E)$.

Claim: $B' \supseteq A \setminus J$.
Claim *Proof.* Suppose not. Then there exists $e \in (A \setminus J) \setminus B'$. But then $J \cup \{e\} \in \mathcal{I}^*$, since its complement contains the basis B', a contradiction. ■

So

$$\begin{aligned} r(E) &= |B'| = |A \setminus J| + |B| \\ r(E) &= |A| - |J| + r(E \setminus A) \\ |J| &= |A| + r(E \setminus A) - r(E) \end{aligned}$$

which does not depend on J. ■

13.5 Linear Programming

Definition: Matroid Polytope

$\mathcal{P}(M) = conv.hull\{x^I : I \in \mathcal{I}\}$ where

$$x_e^I = \begin{cases} 1 & \text{if } e \in I \\ 0 & \text{if } e \notin I \end{cases}$$

Definition: Maximum-Weight Independent Set LP

$$\begin{aligned} \max \quad & \sum(w_e x_e : e \in E) \\ \text{s.t.} \quad & x(A) \leq r(A), \ \forall A \subseteq E \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Definition: Maximum-Weight Independent Set Dual LP

$$\begin{aligned} \min \quad & \sum(r(A) y_A : A \subseteq E) \\ \text{s.t.} \quad & \sum(y_A : e \in A \subseteq E) \geq w_e, \ \forall e \in E \\ & y_A \geq 0, \ \forall A \subseteq E \end{aligned}$$

Definition: Complementary Slackness Conditions

(i) For all $e \in E$, if $x_e > 0$, then $\sum(y_A : e \in A \subseteq E) = w_e$.

(ii) For all $A \subseteq E$, if $y_A > 0$, then $x(A) = r(A)$.

Theorem (Edmonds)

$\mathcal{P}(M)$ is defined by the linear system

$$\begin{aligned} x(A) &\leq r(A), \ \forall A \subseteq E \\ x_e &\geq 0, \ \forall e \in E \end{aligned}$$

Proof. Let $w = (w_e : e \in E)$ and consider the max-weight independent set problem. We may assume $w_e \geq 0$ for all $e \in E$.

Consider the LP relaxation

$$\begin{aligned} \max \quad & \sum(w_e x_e : e \in E) \\ \text{s.t.} \quad & x(A) \leq r(A),\ \forall A \subseteq E \\ & x_e \geq 0,\ \forall e \in E \end{aligned}$$

We show the solution provided by the greedy algorithm is optimal for the LP. This proves the theorem, since it implies that for all w, there exists an optimal LP solution that is an independent set.

Order $E = \{e_1, \ldots, e_n\}$ such that $w_{e_1} \geq \cdots \geq w_{e_n}$. Let x^* be the solution J found by the greedy algorithm with elements processed in that order.

Define $T_0 = \emptyset, T_i = \{e_1, \ldots, e_i\}, i = 1, \ldots, n$.

Consider the dual greedy solution to the dual LP. Set

$$y^*_A = \begin{cases} w_{e_i} - w_{e_{i+1}} & \text{if } A = T_i, 1 \leq i \leq n-1 \\ w_{e_n} & \text{if } A = T_n \\ 0 & \text{all other } A \subseteq E \end{cases}$$

We have $y^*_A > 0$ for all $A \subseteq E$. Let $e_j \in E$. Then

$$\sum(y^*_A : e_j \in A \subseteq E) = \sum_{i=j}^{n} y^*_{T_i} = \sum_{i=j}^{n-1}(w_{e_i} - w_{e_{i+1}}) + w_{e_n} = w_{e_j}$$

Thus, y^* is dual feasible.

We have shown (i) of the complementary slackness conditions for all $e \in E$. To see (ii), suppose $y^*_A > 0$. Then $A = T_i$ for some $1 \leq i \leq n$. We must show $J \cap T_i$ is a maximum independent set in T_i.

Suppose not. Then there exists $e_k \in T_i \setminus J$ such that $J \cap T_i \cap \{e_k\} \in \mathcal{I}$. But in iteration k of the greedy algorithm, e_k was not added to J, a contradiction. ■

Chapter 14

Matroid Intersection

Matroid Intersection Problem

Let matroids $M_1 = (E, \mathcal{I}_1), M_2 = (E, \mathcal{I}_2)$ on some ground set E with weights $w = (w_e : e \in E)$, find a maximum-weight independent set $I \in \mathcal{I}_1 \cap \mathcal{I}_2$.

Example 1: Let $G = (V, E)$ be a bipartite graph with bipartition V_1, V_2. Let

$$\mathcal{I}_1 = \{J \subseteq E : |J \cap \delta(v)| \leq 1, \forall v \in V_1\}$$

where each $v \in V_1$ is incident with at most one element of J and

$$\mathcal{I}_2 = \{J \subseteq E : |J \cap \delta(v)| \leq 1, \forall v \in V_2\}$$

$M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ are partition matroids. A common independent set is a matching in G.

Example 2: Let $G = (V, E)$ be a directed graph with weights $w = (w_e : e \in E)$.

Definition: Branching

A forest such that each vertex has at most 1 entering arc.

Find a maximum-weight branching.

Let M_1 be the forest matroid of the undirected graph G and M_2 be the partition matroid with parts

$$X_v = \{e : e \text{ enters vertex } v\}$$

The common independent set is a branching.

Theorem (Matroid Intersection – Edmonds 1970)

For matroids M_1, M_2 on ground set E with rank functions r_1, r_2 respectively,

$$\max\{|J| : J \in \mathcal{I}_1 \cap \mathcal{I}_2\} = \min\{r_1(A) + r_2(\overline{A}) : A \subseteq E\}$$

14.1 Linear Programming

Let r_1 be the rank function for M_1 and r_2 be the rank function for M_2.

Definition: Matroid Intersection LP

$$\begin{aligned} \max \quad & \sum(w_e x_e : e \in E) \\ \text{s.t.} \quad & x(A) \leq r_1(A), \ \forall A \subseteq E \\ & x(A) \leq r_2(A), \ \forall A \subseteq E \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Definition: Matroid Intersection Dual LP

$$\begin{aligned} \min \quad & \sum(r_1(A)y_A^1 + r_2(A)y_A^2 : A \subseteq E) \\ \text{s.t.} \quad & \sum(y_A^1 + y_A^2 : e \in A \subseteq E) \geq w_e, \ \forall e \in E \\ & y_A^1, y_A^2 \geq 0, \ \forall A \subseteq E \end{aligned}$$

Theorem (Edmonds)

The convex hull of common independent sets of M_1 and M_2 is defined by

$$\begin{aligned} x(A) &\leq r_1(A), \ \forall A \subseteq E \\ x(A) &\leq r_2(A), \ \forall A \subseteq E \\ x_e &\geq 0, \ \forall e \in E \end{aligned}$$

We show that the system is totally dual integral. Recall the TDI theorem: if $Ax \leq b, x \geq 0$ is TDI and b is integral, then all vertices of $P = \{x : Ax \leq b, x \geq 0\}$ are integral.

Dual integrality implies primal integrality. So TDI gives an integral min-max theorem.

Recall the proof of König's theorem for bipartite matching. The matrix A is totally unimodular since all subdeterminants are $0, 1, -1$. If A is totally unimodular, then the LP has an integral optimal solution when the RHS is integral. A is TU if and only if A^T is TU. The dual LP has an integral optimal solution when the primal objective coefficients are integers. $x(\delta(v)) \leq 1$ for all $v \in V$ and $x_e \geq 0$ for all $e \in E$ is TDI.

Now consider matroid intersection. Consider integer weights $w_e \geq 0$ for all $e \in E$. Let $(\overline{y}^1, \overline{y}^2)$ be an optimal (possibly fractional) dual solution. Define $w_e^1 = \sum(\overline{y}_A^1 : e \in A \subseteq E)$ for all $e \in E$ and $w_e^2 = w_e - w_e^1$ for all $e \in E$. For $i = 1, 2$, consider the dual LP for the single matroid M_i.

$$\begin{aligned} \min \quad & \sum(r_i(A)y_A^i : A \subseteq E) \\ \text{s.t.} \quad & \sum(y_A^i : e \in A \subseteq E) \geq w_e^i, \ \forall e \in E \\ & y_A^i \geq 0, \ \forall A \subseteq E \end{aligned}$$

Lemma

$\overline{y}^i$ is optimal for the dual LP i for $i = 1, 2$.

Lemma

If $\hat{y}^i$ is optimal for the dual LP i for $i = 1, 2$, then $(\hat{y}^1, \hat{y}^2)$ is dual optimal for the matroid intersection problem.

Theorem

The matroid intersection system is totally dual integral.

Proof. Let $w = (w_e : e \in E), w_e \geq 0$ for all $e \in E$ be integral. Find an optimal dual solution and create the split $w^1 + w^2$. Take a dual greedy solution $\overline{y}^i$ for each dual LP i.

So the positive variables in $\overline{y}^1, \overline{y}^2$ correspond to nested families

$$T_1^1 \subset T_2^1 \subset \cdots \subset T_{n_1}^1$$

$$T_1^2 \subset T_2^2 \subset \cdots \subset T_{n_2}^2$$

Consider the matrix with columns for E and rows for the sets T_j^1. We can subtract rows to get 1 in each column.

Now consider the matrix $\hat{A}$ from the primal LP with rows corresponding to the positive dual variables $(\hat{y}^1, \hat{y}^2)$, i.e. delete all variables except that are positive. Let N be a square submatrix of $\hat{A}$. Subtracting the rows to get at most one 1 in each column of each half for M_1 and M_2.

Then $\det(N) \in \{0, +1, -1\}$. So $\hat{A}$ is totally unimodular, so there is an integral dual solution, since by the lemma, we only need to consider dual variables that are positive in $(\hat{y}^1, \hat{y}^2)$. ■

Part VII

Traveling Salesman Problem

Chapter 15

The Traveling Salesman Problem

Definition: Tour

A circuit that passes exactly once through each vertex.

This is also known as a Hamiltonian circuit. We typically represent a tour by its edge set T.

Traveling Salesman Problem (TSP)

Given a finite set of points V and a cost c_{uv} of travel between each pair $u, v \in V$, find a tour of minimal cost.

The TSP can be modeled as a graph problem by considering the complete graph. TSP belongs to class of $\mathcal{NP}$-hard problems. The cost of a tour T is $\sum(c_e : e \in T)$.

Euclidean TSP

Given points $p_1, \ldots, p_n \in \mathbb{R}^2$, let c_{ij} be the Euclidean distance between p_i and p_j. Find an optimal TSP tour.

15.1 Decision Problem

TSP Decision Problem

Given an integer k, does G have a tour of cost $\leq k$?

It is in $\mathcal{NP}$ since we can certify yes by giving a tour of cost $\leq k$. It is not known to be in co-$\mathcal{NP}$. It is $\mathcal{NP}$-hard and $\mathcal{NP}$-complete since it is in $\mathcal{NP}$.

If $\mathcal{P} \neq \mathcal{NP}$, then for any TSP algorithm and any polynomial function $p(n)$, there is an example on $\hat{n}$ vertices such that the running time is greater than $p(\hat{n})$. $\hat{n}$ is finite, but may be large and this does not mean the algorithm cannot solve some large finite example but there may exist large nasty examples for your algorithm.

In theory, we can either prove $\mathcal{P} = \mathcal{NP}$ or $\mathcal{P} \neq \mathcal{NP}$, improve the runtime bound for an exact TSP solver, improve polynomial time approximation, study special cases, or study theory questions related to techniques that have worked well in practice.

In practice, we have

1. Heuristic search methods: find good solutions, but no direct bounds on the performance.
2. Exact solution algorithms: can fail on examples, but aim to get a strong upper and lower bound on the optimal tour cost. It is used to benchmark heuristic methods, and optimal (or near-optimal) bounds are useful in some applications like genetics.

15.2 Asymmetric TSP

Asymmetric TSP

In a complete directed graph where we may have $c_{ij} \neq c_{ji}$, find a TSP tour.

Any instance of the ATSP can be easily modeled as an instance of the TSP.

We can split each vertex v into two vertices v_{in} and v_{out}. The new edge $v_{in}v_{out}$ costs will be set to force it to be in every optimal tour.

For a pair of vertices u, v, the edges uv and vu are replaced as follows (with undirected edges). Set costs of $u_{in}v_{in}$ and $u_{out}v_{out}$ so they cannot be in any optimal tour, i.e. ∞. And set the cost of $u_{in}u_{out}$ and $v_{in}v_{out}$ to be always in the tour, i.e. $-\infty$.

An optimal tour in the new graph corresponds to an optimal directed tour in the original graph.

We have doubled the number of vertices. A bigger difficulty in practice is the unusual cost structure.

15.2.1 Number of TSP Tours

An n-vertex instance of the TSP has $\frac{(n-1)!}{2}$ tours, and an n-vertex instance of the ATSP has $(n-1)!$ tours.

Although it is often cited as being the reason for difficulty to solve, there are more spanning trees than tours in a graph.

Bellman and independently Held and Karp showed that TSP and ATSP can be solved in $n^2 2^n$ steps using dynamic programming.

15.3 Bellman-Held-Karp Algorithm

Algorithm 10 Bellman-Held-Karp Algorithm (TSP/ATSP)

1: Let start vertex $n \in V$
2: $S \subseteq V \setminus \{n\}$
3: $x \in S$
4: $t(S, x) =$ cost of cheapest path starting at n, visiting all vertices in S and ending at x
5: $t(S, x) = \min(t(S \setminus \{x\}, y) + c_{yx} : y \in S \setminus \{x\}$

This is a recursive algorithm to find $t(S, x)$. This works backwards to get the actual optimal tour. It works for TSP and ATSP.

Chapter 16

TSP Approximation Algorithms

It is $\mathcal{NP}$-hard to decide if a graph has a Hamiltonian circuit. There is no polynomial time approximation algorithm unless $\mathcal{P} = \mathcal{NP}$.

16.1 Metric TSP

Metric TSP

Given a complete graph with edge costs $c = (c_e : e \in E)$ satisfying triangle inequality, find a TSP tour.

Triangle inequality: $c_{xy} \leq c_{xz} + c_{zy}$ for all $x, y, z \in V$.

Any TSP instance can be reduced to a metric instance by adding a large constant M to all edge costs. This does not preserve approximation factors.

16.1.1 Simple 2-Approximation

Proposition

If the triangle inequality is satisfied, then $c_{xy} \geq 0$ for all $xy \in E$.

Proof. Suppose $c_{xy} < 0$. Let $z \in V$. We may assume $c_{yz} \leq c_{xz}$. Then $c_{xz} > c_{yz} + c_{xy}$. ■

Proposition

If the triangle inequality is satisfied, then an Euler tour can be a 'shortcut' to a TSP tour.

Proof. Let $v_1, \ldots, v_n$ be the order each vertex if first visited by the Euler tour. This gives a TSP tour of cost $\leq$ Euler tour. ■

These two observations give a 2-approximation for metric TSP.

Algorithm 11 2-Approximation (Metric TSP)

1: Find a minimum spanning tree Q (Since any tour gives a spanning tree by deleting an edge, $\text{cost}(Q) \leq \text{cost}(TSP)$)
2: Double each edge in Q (This gives a graph of even degree)
3: Find an Euler tour in the double tree and shortcut to a TSP tour.

16.1.2 Christofides Algorithm

We can improve the double tree by taking $Q + M$, where M is a min-cost perfect matching on the vertices S that have odd degree in Q.

Algorithm 12 Christofides Algorithm (Metric TSP)

1: Find a minimum spanning tree Q
2: Find a minimum-cost perfect matching M on vertices S having odd degree in Q
3: $Q + M$ is an even degree graph
4: Find an Euler tour in $Q + M$
5: Shortcut to a TSP tour

$|S|$ is even. A TSP tour for the full graph can be shortcut to a TSP tour for S. Since a TSP tour through an even number of points consists of two perfect matchings, we have $\text{cost}(M) \leq \frac{1}{2}\,\text{cost}(TSP)$.

This implies $\text{cost}(Q + M) \leq \frac{3}{2}\,\text{cost}(TSP)$. So the Christofides tour has cost $\leq \frac{3}{2}\,\text{cost}(TSP)$.

This has been improved to $\frac{3}{2} - 10^{-36}$ with a randomized algorithm in 2021 and a deterministic algorithm in 2023 by Karlin, Klein, and Oveis Gharan.

Research problem: Find a 1.49-approximation for metric TSP.

We see that this is connected to T-joins, by letting $T = \{v \in V : v \text{ odd degree in } Q\}$, then a min-cost T-join consists of paths between vertices in T. The triangle inequality implies we can shortcut the paths to a perfect matching.

16.2 Subtour Relaxation

Definition: Subtour Polytope

$$x(\delta(v)) = 2, \ \forall v \in V$$
$$x(\delta(S)) \geq 2, \ \forall S \subsetneq V, |S| \geq 2$$
$$0 \leq x_e \leq 1$$

The first constraint is called the **degree** constraints. The second constraint is called the **subtour-elimination** constraint (or subtour constraint/subtour inequality). An integer solution to the subtour relaxation LP is a tour.

Definition: TSP Subtour LP

$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) = 2, \ \forall v \in V \\ & x(\delta(S)) \geq 2, \ \forall S \subsetneq V, |S| \geq 2 \\ & 0 \leq x_e \leq 1, \ \forall e \in E \end{aligned}$$

Adding $x(\delta(v)) = 2$ for all $v \in S$, we have

$$2\,|S| = 2x(\gamma(S)) + x(\delta(S))$$

So for x satisfying all degree constraints we have

$$\begin{aligned} x(\delta(S)) \geq 2 &\iff 2\,|S| - 2x(\gamma(S)) \geq 2 \\ &\iff |S| - x(\gamma(S)) \geq 2 \\ &\iff x(\gamma(S)) \leq |S| - 1 \end{aligned}$$

This is the "inside" form of the subtour inequality. We can rewrite the subtour polytope as

$$\begin{aligned} & x(\delta(v)) = 2, \ \forall v \in V \\ & x(\gamma(S)) \leq |S| - 1, \ \forall S \subsetneq V, |S| \geq 2 \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Recall that the spanning tree polytope was

$$\begin{aligned} & x(\gamma(S)) \leq |S| - 1, \ \forall S \subsetneq V, |S| \geq 2 \\ & x(E) = |V| - 1 \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Let $n = |V|$. Suppose $x \in$ subtour polytope. Adding the degree constraints, we have $x(E) = n$. Thus, $\frac{n-1}{n}x \in$ spanning tree polytope.

$\frac{n-1}{n}x$ can be written as a convex combination of spanning trees. So there exist spanning trees $F_1, \ldots, F_k$ such that

$$\frac{n-1}{n}x = \sum_{i=1}^{k} \lambda_i x^{F_i}$$

where

$$x^{F_i} = \begin{cases} 1 & \text{if } e \in F_i \\ 0 & \text{if } e \notin F_i \end{cases}$$

Algorithm 13 Best-of-Many Christofides Algorithm

1: x^* be an optimal solution to subtour LP
2: Write x^* as convex combination of spanning trees
3: Run Christofides algorithm on each of the trees (or sample with some distribution)
4: **return** Best TSP tour

16.3 Best-of-Many Christofides

This improved the Christofides bound on several problems related to the TSP, like the Path TSP, but does not directly improve $\frac{3}{2}$-approximation for the TSP.

Definition: SUB

Optimal objective value of the subtour LP.

$$SUB \leq TSP$$

In practice, SUB is a good lower bound.

Theorem (Wolsey 1980)

For the metric TSP, Christofides finds a TSP tour of cost $\leq \frac{3}{2}SUB$.

Proof. Let $x \in$ subtour polytope. Since $\frac{n-1}{n}x$ is in the spanning tree polytope, the cost of a minimum spanning tree is $\leq SUB$. Let T be the odd vertices in the spanning tree. Recall that the min-cost of a T-join is equal to

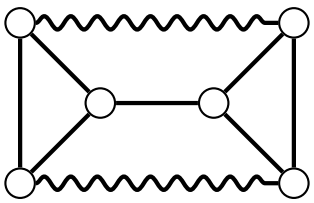
$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(D) \geq 1, \ \forall \ T\text{-cuts } D \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

$\frac{1}{2}x$ is a feasible solution to this LP. So the min-cost perfect matching M for the vertices T has cost $\leq SUB$. ■

So $TSP \leq \frac{3}{2}SUB$, which implies the integrality ratio for the subtour LP is at most $\frac{3}{2}$.

16.4 Approximations to TSP

Example: Consider the graph



Edges in the triangles have cost 2. Paths have q edges of cost 1. Edges uv not drawn have cost equal to the cost of a shortest uv-path.

The tour has cost $4q+6$ since paths have cost q. The subtour $\overline{x}$ has $\overline{x}_e = \frac{1}{2}$ for each edge in the triangle, so the paths have cost q and each triangle has cost 3. So $\overline{x}$ has cost $3q+6$.

$$\frac{4q+6}{3q+6} \to \frac{4}{3} \text{ as } q \to \infty$$

$\frac{4}{3}$-Approximation Conjecture (Goemens)

For metric TSP, $TSP \leq \frac{4}{3} SUB$.

Theorem (Benoit-Boyd 2008)

$\frac{4}{3}$-approximation conjecture is true for $n \leq 12$.

Let $TSP^*(c)$ be the optimal value of TSP with edge costs c and $SUB^*(c)$ be the optimal value of the subtour LP with costs c. The conjecture has

$$\alpha = \max_{\text{metric } c} \frac{TSP^*(c)}{SUB^*(c)} \leq \frac{4}{3}$$

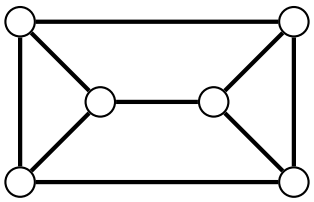
Let SUB_n denote the subtour polytope for $n = |V|$. So

$$\alpha_n = \max_{\text{metric } c} \frac{TSP_n^*(c)}{SUB_n^*(c)}$$

Benoit and Boyd showed there is (for $n \leq 12$) a unique vertex of SUB_n that gives the max α_n.

n	6	7	8	9	10
α_n	$\frac{10}{9}$	$\frac{9}{8}$	$\frac{8}{7}$	$\frac{7}{6}$	$\frac{20}{17}$

Graph with unique vertex ($n = 6$):



Chapter 17

Lower Bounds for TSP

17.1 Held-Karp Bound

Select a vertex $v \in V$.

Definition: 1-Tree

A set $U \subseteq E$ if it consists of two edges meeting v and a spanning tree of $G \setminus \{v\}$.

The minimum cost 1-tree is a lower bound for the TSP.

From the spanning tree polytope, we have that the convex hull of 1-trees is defined by

$$\begin{aligned} &x(\gamma(S)) \leq |S| - 1, \ \forall S \subsetneq V, S \neq \emptyset \\ &x(\delta(v)) = 2 \\ &x(E) = n \end{aligned}$$

Held and Karp improved the 1-tree bound.

Given values $(y_v : v \in V)$, create new edge costs

$$\overline{c}_{uv} = c_{uv} - y_u - y_v$$

For a tour $T \subseteq E$,

$$c(T) = \overline{c}(T) + 2\sum(y_v : v \in V)$$

Definition: Held-Karp Bound

Given $(y_v : v \in V)$ and U be a minimum cost 1-tree using edge cost $\overline{c}$, then every TSP tour has cost

$$\geq \overline{c}(U) + 2\sum(y_v : v \in V)$$

We can find a good choice of vertex values y_v by an ascent algorithm: if the degree of a vertex v in the 1-tree is < 2, increase y_v. If degree > 2, then decrease y_v.

Theorem

The optimal Held-Karp bound (the bound given by the best choice of $(y_v : v \in V)$) is equal to the optimal value of the subtour LP.

The dual variables for the degree constraints in the subtour LP give the $(y_v : v \in V)$.

Algorithm 14 Held-Karp 1-Tree Algorithm

$Q \subseteq E$ be minimum cost 1-tree using edge costs $\bar{c}_e = c_e - y_u - y_v$ for all $uv \in E$
for $v \in V$ **do**
    $d_v = |\delta(v) \cap Q|$ (degree of v in Q)
    if $d_v > 2$ **then**
        Increase cost of edges in $\delta(v)$
    else if $d_v = 1$ **then**
        Decrease cost of edges in $\delta(v)$
    $y_v = y_v + t(2 - d_v)$ for a step size t (gradually decrease t as we iterate)

Theorem

Any point x in the subtour polytope can be written as a convex combination of 1-trees, each of maximum degree 4 such that every vertex has average degree 2.

Proof. Uses matroid intersection. ■

Chapter 18

Subtour Separation

Subtour Separation

Let $x \in \mathbb{R}^E$. Check that $x \geq 0$ and $x(\delta(v)) = 2$ for all $v \in V$. Let $u_e = x_e$ be the edge capacities and find a minimum weight cut in the graph G.

Finds $S \subseteq V(G)$ and if $x(\delta(S)) < 2$, we have a violated inequality, otherwise $x \in$ subtour polytope.

Global Minimum Cut

Karger's random contraction algorithm or $n-1$ max-flow min-cut computations.

$n-1$ max-flow min-cut: Choose a vertex $v_1 \in V$. We may assume the global min cut has $v_1 \in S$ (since S and $V \setminus S$ define the same cut). Find a max flow from v_1 to every other vertex $v \in V \setminus \{v_1\}$. Each gives a minimum v_1v-cut. The global min cut is the minimum of these $n-1$ cuts.

The $n-1$ max-flow approach gives an extended formulation of the subtour polytope with $O(n^3)$ variables and constraints.

An st-max-flow problem is an LP. So for each $t \in V \setminus \{v_1\}$, we write the v_1t max flow constraints using variables f_e^+ for all $e \in E$. We add extra constraints to force the flow to have value 2 (flow leaving v_1). The variables f_e^+ are linked by capacity constraints $f_e^+ \leq x_e$ for all $e \in E$.

We add constraints $x_e \geq 0$ for all $e \in E$ and $x(\delta(v)) = 2$ for all $v \in V$. Now x satisfies all constraints if and only if x is in the subtour polytope.

The subtour bound can be computed as a single polynomial size LP. However, no polynomial time LP solver is a combinatorial algorithm.

Theorem (Planar Subtour Polytope – Riven 1996)

Suppose $G = (V, E)$ is planar and $G^* = (V^*, E^*)$ is the planar dual graph. The subtour polytope for G can be formulated with $3\,|E|$ nonnegative variables and $|E|+|V|+|V^*|$ constraints.

The idea is to use variables that model both a spanning tree in G and a spanning tree in G^*. This allows you to avoid the exponentially many constraints $x(\gamma(S)) \leq |S| - 1$.

Riven found the result while proving an 1832 problem by Steiner: when is a polyhedron combinatorially equivalent to one inscribed in a sphere?

This happens if and only if G is the support graph of a point in the relative interior of the subtour polytope.

18.1 Cutting Planes

Consider the subtour LP

$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) = 2, \ \forall v \in V \\ & x(\delta(S)) \geq 2, \ \forall S \subsetneq V, S \neq \emptyset \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Let $\mathcal{S}$ denote the collection of all sets $S \subsetneq V, S \neq \emptyset$, and consider a subset $\mathcal{S}' \subseteq \mathcal{S}$. If $\overline{x}$ is an optimal solution to the relaxed LP'

$$\begin{aligned} \min \quad & \sum(c_e x_e : e \in E) \\ \text{s.t.} \quad & x(\delta(v)) = 2, \ \forall v \in V \\ & x(\delta(S)) \geq 2, \ \forall S \in \mathcal{S}' \\ & x_e \geq 0, \ \forall e \in E \end{aligned}$$

Then $\sum(c_e \overline{x}_e : e \in E)$ is a lower bound for the subtour LP. More importantly, any dual solution for LP' gives a lower bound for the subtour LP.

If $\overline{x}$ is a feasible solution to the subtour LP, then it is an optimal solution the subtour LP.

We use min-cut to check it $\overline{x}$ satisfies all constraints.

We add subtour inequalities as they are needed. This is very efficient in practice.

18.1.1 Comb Inequalities

The class of inequalities to use in the last step of the cutting plane algorithm.

Algorithm 15 Cutting Plane Algorithm for TSP

1: Initial degree LP: $\min\{\sum(c_e x_e : e \in E) : x(\delta(v)) = 2 \ \forall v \in V, 0 \leq x_e \leq 1 \ \forall e \in E\}$
2: Let x^* be the optimal solution to the LP
3: $E^* = \{e : x_e^* > 0\}, G^* = (V, E^*)$
4: **if** G^* is not connected **then**
5: $\quad$ $S_1, \ldots, S_k$ be the vertex sets of the connected components
6: $\quad$ Add subtour constraints for each S_i
7: Use min-cut to find sets $S \subseteq V$ with $x^*(\delta(S)) < 2$ (let capacities $u_e = x_e^*$ for all $e \in E$)
8: **if** all subtour inequalities are satisfied **then**
9: $\quad$ Look for further valid TSP inequalities

Definition: Comb

A set $H \subseteq V$ is the **handle** and $T_1, \ldots, T_p \subseteq V$ for p odd and $p \geq 3$ are called the **teeth**.
$T_i \cap T_j = \emptyset$ for all i, j, $H \cap T_i \neq \emptyset$ for all i, and $T_i \setminus H \neq \emptyset$ for all i.

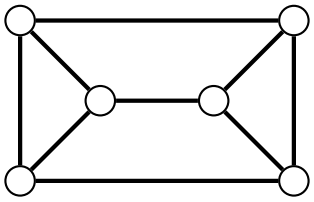
Definition: Comb Inequality

Every tour vector satisfies

$$x(\delta(H)) + \sum_{i=1}^{p} x(\delta(T_i)) \geq 3p + 1$$

For $p = 3$, we have $x(\delta(H)) \geq 2$ and $x(\delta(T_i)) \geq 2$ for $i = 1, 2, 3$. So $x(\delta(H)) + \sum_{i=1}^{3} x(\delta(T_i)) \geq 8$

The comb inequality has ≥ 10, so it is stronger.



For this, we have the left triangle as the handle and the three teeth are the horizontal edges. If each edge in the triangles have $\overline{x}_e = \frac{1}{2}$ and each teeth edge are $\overline{x}_e = 1$, then $\overline{x}(\delta(H)) = 3$, $\overline{x}(\delta(T_i)) = 2$, this is a violated comb.

To see that combs are valid, we start with a domino. Let $A, B \subseteq V, A \cap B \neq \emptyset, A \cup B \neq V$. A and B form a domino shape.

Define $x(\delta(A)) \geq 2, x(\delta(B)) \geq 2, x(\delta(A \cup B)) \geq 2$, then $2x(\delta(A \cup B)) + 2x(E(A : B)) \geq 6$, where $E(A : B)$ are the edges having one end in A and the other end in B. So $x(\delta(A \cup B)) + x(E(A : B)) \geq 3$. $x(\delta(A \cup B)) + x(E(A : B))$ are the edges crossing the border of the domino.

For each tooth T_i, we have a domino $T_i \setminus H$ and $T_i \cap H$. We have $x(\delta(H)) + \sum_{i=1}^{p} x(\delta(T_i)) \geq \sum_{i=1}^{p} T_i \geq 3p$.

In any tour $\overline{x}$, $x(\delta(H)) + \sum_{i=1}^{p} x(\delta(T_i))$ is an even number, so we have the quantity at least $3p + 1$.

Recall that for all $S \subseteq V$ and x satisfying the degree equations, we have

$$2\,|S| = 2x(\gamma(S)) + x(\delta(S)) \implies x(\delta(S)) = 2\,|S| - 2x(\gamma(S))$$

So the comb inequality may be written as

$$\begin{aligned}
2\,|H| - 2x(\gamma(H)) + \sum_{i=1}^{p}(2\,|T_i| - 2x(\gamma(T_i)) &\geq 3p + 1 \\
-2x(\gamma(H)) + \sum_{i=1}^{p} -2x(\gamma(T_i)) &\geq 3p + 1 - 2\,|H| - \sum_{i=1}^{p} 2\,|T_i| \\
2x(\gamma(H)) + \sum_{i=1}^{p} 2x(\gamma(T_i)) &\leq 2\,|H| + \sum_{i=1}^{p} 2\,|T_i| - (3p+1) \\
x(\gamma(H)) + \sum_{i=1}^{p} x(\gamma(T_i)) &\leq |H| + \sum_{i=1}^{p} |T_i| - \frac{3p+1}{2}
\end{aligned}$$

Comb inequalities were first found by Chvátal in 1971. This is called the **Chvátal cut**.

This is a general way to find valid inequalities for the integer solution in a polytope P. If c integer, then $c^T x \leq \lfloor \delta \rfloor$ is valid for $P \cap \mathbb{Z}^n$.

Research question: Given x in the subtour polytope, find a comb inequality violated by x in polytime, if one exists.

- Padberg and Rao (1980): For $|T_i| = 2$ for all tooth T_i.
- Curr (1997): Polytime If k is fixed (runtime is exponential in k).
- Fleischer and Tardos (1999): If G^* is planar, we can find a comb inequality in polytope violated by 1.
- Letchford (2000): If G^* is planar, we can solve the separation problem for domino-parity constraints in polytime.

Research question: Polytime separation for domino-parity constraints.

Part VIII

Additional Topics

Chapter 19

Metaheuristics

These are general ideas/strategies for creating heuristic algorithms for specific classes of problems.

19.1 Local Search

Flood in 1956 developed a 2-approximation for TSP using local search.

If we delete edges e_1 and e_2 from a tour T, we obtain paths P_1 and P_2. We can build a new tour T' by adding two new edges f_1 and f_2. If $c_{e_1} + c_{e_2} > c_{f_1} + c_{f_2}$, then $c(T') < c(T)$. This is called a 2-opt move (or improving 2-swap). T and T' are called **neighbors**.

Local search: Look for a lower-cost tour in the neighborhood of T. If $c(T') < c(T)$, then replace T with T'.

Definition: Locally Optimal

If no tour in the neighborhood of T has lower cost, then T is locally optimal.

Algorithm 16 Local Search

```
1: x = constructed solution (greedy or random)
2: while x is not locally optimal or time permitting do
3:     Examine neighborhood of x
4:     if neighbor solution y has c(y) < c(x) then
5:         x = y
6: return x
```

In practice, we stop the loop after a fixed amount of time, so the produced solution possibly will not be locally optimal. Local search algorithms are hill climbing, as we always move up or down to a better solution.

19.2 Improving Local Search

We can look at more complex/larger neighborhoods. It is important to be able to search the neighborhood efficiently.

3-approximation for TSP: Remove 3 edges and reconnect the 3 paths.

k-approximation for TSP: Remove k edges and reconnect the k paths. This has a large neighborhood, but costly to search when $k \geq 4$.

19.2.1 Variable k-Approximation

Lin-Kernighan (1973) - Variable k-Approximation: If it looks promising to remove an extra edge, then the algorithm will increase k.

This is very effective and remains a key ingredient in best-known TSP heuristics.

19.2.2 Multiple Starting Points

As long as time permits, repeat the local search algorithm from different initial solutions. For TSP, we have nearest-neighbor tours and random tours.

19.2.3 Simulated Annealing

At the start of the search, allow moves that make solutions worse. As the algorithm progresses, gradually decrease the probability of accepting a downhill move.

Idea: Allow the algorithm to initially move away from local optima that may not have good values (not close to the value of the global optimal solution). As the algorithm progresses, we want the solution ot be trapped in the area of a good solution, then switch to hill climbing to reach the local optimum.

At step k, the probability of accepting a worse solution y depends on

$$\Delta = c(y) - c(x)$$

and a parameter t_k called the temperature.

If $\Delta < 0$, then we accept the move (uphill). If $\Delta \geq 0$, then we accept the move with probability

$$\frac{1}{e^{\Delta/t_k}}$$

If Δ is large, then we have only a small probability to accept the move (downhill). If we make t_k smaller, then the probability that we accept a move decreases.

During the algorithm, t_k gradually decreases as k gets larger (called cooling down).

"Optimization by simulated annealing". Kirkpatrick, Gelatt, Vecchi 1983.

Algorithm 17 Simulated Annealing

```
1: x = greedy/random solution
2: x* = x (best solution found so far)
3: t = starting temperature
4: while time permitting do
5:     Choose a random neighbor x' of x
6:     Δ = c(x') − c(x)
7:     if Δ < 0 then
8:         x = x'
9:         if c(x) < c(x*) then
10:            x* = x
11:    else
12:        Choose random number r uniformly in [0, 1]
13:        if r < e^{−Δ/t} then
14:            x = x'
15:    Lower temperature t
16: return x*
```

19.2.4 Chained Local Search

Algorithm 18 Chained Local Search

```
1: while true do
2:     Find a local optimal solution x
3:     Kick x to find a solution x' outside of x's neighborhood
4:     Apply local search to x' to obtain x''
5:     if c(x'') ≤ c(x) then
6:         x = x''
7: return x
```

Gives a more focused search than multiple starts. Local search algorithm for T' will often run faster than from a random solution.

Example: TSP kick (double-bridge move). Remove $e_1 = (v_1, v_2), \ldots, e_4 = (v_7, v_8)$ in the tour (cyclic order) and connect

- v_1, v_6
- v_2, v_5
- v_3, v_8

- v_4, v_7

Chained local search was developed by Martin, Otto, Felten (1992), targeting the TSP.

19.3 Genetic Algorithms

Algorithm 19 Genetic Algorithm

1: Keep multiple solutions $x_1, \ldots, x_k$ (population)
2: Can apply local search to any x_i
3: Mate pairs of solutions x_i and x_j to obtain x_{child}
4: Local search on x_{child} to get x'_{child}
5: Add x'_{child} to population
6: Remove poor solutions from population (fitness)

Many variations of this scheme have been proposed and is widely used since it is easy to implement and can often deliver good solutions.

TSP Mating Example: EAX - Edge assembly crossover. Color edges of T_i red and the edges of T_j blue. Consider the union of T_i and T_j. All vertices have even degree. Find a circuit C that alternates between blue and red edges.

In tour T_i, delete the blue edges in C and add the red edges. This can result in 2 or more subtours. Convert to a tour by a sequence of 2-swaps. This results in a T_{child}.

The combination of local search and genetic algorithms can often produce very good solutions.