C&O 355
Lecture 12

N. Harvey

Topics

- Polynomial-Time Algorithms
- Ellipsoid Method Solves LPs in Polynomial Time
- Separation Oracles
- Convex Programs
- Minimum s-t Cut Example

Polynomial Time Algorithms

- P = class of problems that can be solved efficiently
  i.e., solved in time $\leq n^c$, for some constant c, where n=input size
- This is a bit vague
  - Consider an LP $\max \{ c^T x : Ax \leq b \}$ where A has size m x d
  - Input is a binary file containing the matrix A, vectors b and c
- Two ways to define "input size"
  A. # of bits used to store the binary input file
  B. # of numbers in input file, i.e., $m \cdot d + m + d$
- Leads to two definitions of "efficient algorithms"
  A. Running time $\leq n^c$ where n = # bits in input file
  B. Running time $\leq n^c$ where $n = m \cdot d + m + d$

"Polynomial Time Algorithm"

"**Strongly** Polynomial Time Algorithm"

Algorithms for Solving LPs

Name	Publication	Running Time	Practical?
Fourier-Motzkin Elimination	Fourier 1827, Motzkin 1936	Exponential	No
Simplex Method	Dantzig '47	Exponential	Yes
Perceptron Method	Agmon '54, Rosenblatt '62	Exponential	Sort of
Ellipsoid Method	Khachiyan '79	Polynomial	No
Interior Point Method	Karmarkar '84	Polynomial	Yes
Analytic Center Cutting Plane Method	Vaidya '89 & '96	Polynomial	No
Random Walk Method	Bertsimas & Vempala '02-'04	Polynomial	Probably not
Boosted Perceptron Method	Dunagan & Vempala '04	Polynomial	Probably not
Random Shadow-Vertex Method	Kelner & Spielman '06	Polynomial	Probably not

- Unsolved Problems:
  - Is there a **strongly polynomial time** algorithm?
  - Does some implementation of **simplex method** run in polynomial time?

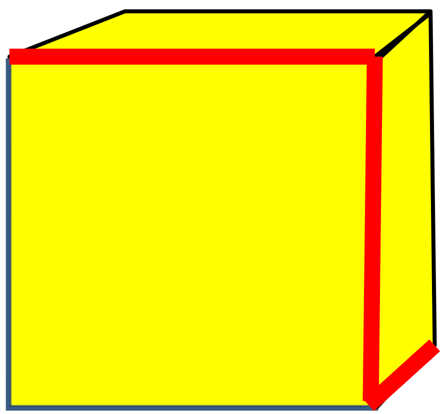
Why is analyzing the simplex method hard?

- Recall how the algorithm works:
  - It starts at a vertex of the polyhedron
  - It moves to a “neighboring vertex” with better objective value
  - It stops when it reaches the optimum
- How many moves can this take?
- For any polyhedron, and for any two vertices, can you move between them with few moves?

Why is analyzing the simplex method hard?

- For any polyhedron, and for any two vertices, can you move between them with few moves?
- **The Hirsch Conjecture** (1957)
  Let P = { x : Ax $\leq$ b } where A has size m x n.
  You can move between any two vertices using only m-n moves.

Example: A cube.
Dimension n=3.
constraints m=6.
Do m-n=3 moves suffice?



Yes!

Why is analyzing the simplex method hard?

- For any polyhedron, and for any two vertices, can you move between them with few moves?
- The Hirsch Conjecture (1957)
  Let $P = \{ x : Ax \leq b \}$ where A has size m x n.
  You can move between any two vertices using only m-n moves.
- We have no idea how to prove this.
- **Theorem:** [Kalai-Kleitman 1992] $m^{\log n+2}$ moves suffice.
- Still the best known result. Proof **amazingly beautiful**! We might prove it later in the course…
- **Want to prove a better bound?** A group of (eminent) mathematicians have a blog organizing a massively collaborative project to do just that.

Ellipsoid Method for Solving LPs

- Ellipsoid method finds feasible point in P = { x : Ax $\leq$ b }
  i.e., it can solve a system of inequalities
- But we want to **optimize**, i.e., solve max { $c^\mathsf{T}x$: x$\in$P }
- **Restatement of Strong Duality Theorem:** (from Lecture 8)
  Primal has optimal solution $\Leftrightarrow$ Dual has optimal solution
  $\Leftrightarrow$ the following system is solvable:

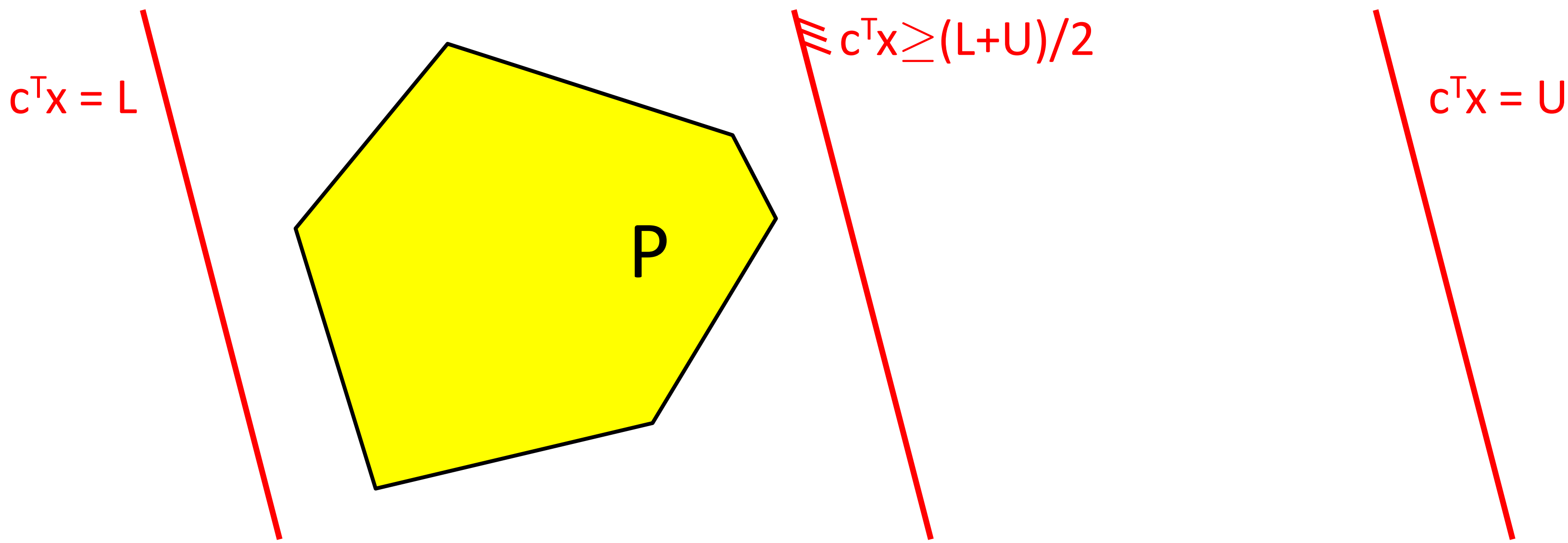
$$Ax \leq b \qquad A^\mathsf{T} y = c \qquad y \geq 0 \qquad c^\mathsf{T} x \geq b^\mathsf{T} y$$

"Solving an LP is equivalent to solving a system of inequalities"

$\Rightarrow$ Ellipsoid method can be used to solve LPs

Ellipsoid Method for Solving LPs

- Ellipsoid method finds feasible point in $P = \{ x : Ax \leq b \}$
  i.e., it can solve a system of inequalities
- But we want to **optimize**, i.e., solve $\max \{ c^T x : x \in P \}$
- **Alternative approach:** Binary search for optimal value
  – Suppose we know optimal value is in interval $[L,U]$
  – Add a new constraint $c^T x \geq (L+U)/2$
  – If LP still feasible, replace L with $(L+U)/2$ and repeat
  – If LP not feasible, replace U with $(L+U)/2$ and repeat



Issues with Ellipsoid Method

1. It needs to compute square roots, so it must work with irrational numbers
   - **Solution:** Approximate irrational numbers by rationals. Approximations proliferate, and it gets messy.
2. Can only work with bounded polyhedra P
   - **Solution:** If P non-empty, there exists a solution x s.t. $|x_i| \leq U \ \forall i$, where U is a bound based on numbers in A and b. So we can assume that $-U \leq x_i \leq U$ for all i.
3. Polyhedron P needs to contain a small ball B(z,k)
   - **Solution:** If $P = \{ x : Ax \leq b \}$ then we can perturb b by a tiny amount. The perturbed polyhedron is feasible iff P is, and if it is feasible, it contains a small ball.

Ellipsoid Method in Polynomial Time

- **Input:** A polyhedron $P = \{ x : Ax \leq b \}$ where A has size m x d. This is given as a binary file containing matrix A and vector b.
- **Input size: n** = # of bits used to store this binary file
- **Output:** A point $x \in P$, or announce "P is empty"
- **Boundedness:** Can add constraints $-U \leq x_i \leq U$, where $U = 16^{d^2 n}$. The new P is contained in a ball $B(0,K)$, where $K < n \cdot U$.
- **Contains ball:** Add ϵ to b_i, for every i, where $\epsilon = 32^{-d^2 n}$. The new P contains a ball of radius $k = \epsilon \cdot 2^{-dn} > 64^{-d^2 n}$.
- **Iterations:** We proved last time that:
  \# iterations $\leq 4d(d+1)\log(K/k)$, and this is $< 40d^6n^2$
- Each iteration does only basic matrix operations and can be implemented in polynomial time.
- **Conclusion:** Overall running time is polynomial in n (and d)!

What Does Ellipsoid Method Need?

- The algorithm uses almost nothing about polyhedra (basic feasible solutions, etc.)
- It just needs to (repeatedly) answer the question:
  Is $z \in P$?
  If not, give me a constraint "$a^T x \leq b$" of P violated by z

Let $E(M,z)$ be an ellipsoid s.t. $P \subseteq E(M,z)$
If vol $E(M,z)$ < vol $B(0,r)$ then Halt: "P is empty"
If $z \in P$, Halt: "$z \in P$"
Else
    Let "$a_i^T x \leq b_i$" be a constraint of P violated by z (i.e., $a_i^T z > b_i$)
    Let $H = \{ x : a_i^T x \leq a_i^T z \}$ (so $P \subseteq E(M,z) \cap H$)
    Let $E(M',z')$ be an ellipsoid covering $E(M,z) \cap H$
    Set $M \leftarrow M'$ and $z \leftarrow z'$ and go back to Start

- **Input:** A polytope $P = \{ Ax \leq b \}$
- **Output:** A point $x \in P$, or announce "P is empty"

The Ellipsoid Method

- The algorithm uses almost nothing about polyhedra (basic feasible solutions, etc.)
- It just needs to (repeatedly) answer the question:

Separation Oracle

Is $z \in P$?

If not, find a vector a s.t. $a^T x < a^T z \;\; \forall x \in P$

- The algorithm works for **any convex set** P, as long as you can give a separation oracle.
  - P still needs to be bounded and contain a small ball.
- **Remarkable Theorem:** [Grotschel-Lovasz-Schijver '81]
  For any convex set $P \subseteq \mathbb{R}^n$ with a separation oracle, you can find a feasible point efficiently.
- **Caveats:**
  - "Efficiently" depends on size of ball containing P and inside P.
  - Errors approximating irrational numbers means we get "approximately feasible point"

Martin Grotschel

Laszlo Lovasz

Alexander Schrijver

The Ellipsoid Method For Convex Sets

Separation Oracle

Is $z \in P$?

If not, find a vector a s.t. $a^T x < a^T z \quad \forall x \in P$

- **Feasibility Theorem:** [Grotschel-Lovasz-Schijver '81]
  For any convex set $P \subseteq \mathbb{R}^n$ with a separation oracle, you can find a feasible point efficiently.
  - Ignoring (many, technical) details, this follows from ellipsoid algorithm
- **Optimization Theorem:** [Grotschel-Lovasz-Schijver '81]
  For any convex set $P \subseteq \mathbb{R}^n$ with a separation oracle, you can solve optimization problem $\max \{ c^T x : x \in P \}$.
  - How?
  - Follows from previous theorem and binary search on objective value.
- This can be generalized to minimizing non-linear (convex) objective functions.

Separation Oracle for Ball

- Let's design a separation oracle for the convex set $P = \{ x : \|x\| \leq 1 \}$ = unit ball $B(0,1)$.

Separation Oracle

Is $z \in P$?

If not, find a vector a s.t. $a^T x < a^T z \ \ \forall x \in P$

- **Input:** a point $z \in \mathbb{R}^n$
- If $\|z\| \leq 1$, return "Yes"
- If $\|z\| > 1$, return $a = z / \|z\|$
  - For all $x \in P$ we have

    $a^T x = z^T x / \|z\| \leq \|x\|$ **Why?** Cauchy-Schwarz
  - For z we have

    $a^T z = z^T z / \|z\| = \|z\| > 1 \geq \|x\| \quad \Rightarrow \ a^T x < a^T z$

Separation Oracle for Ball

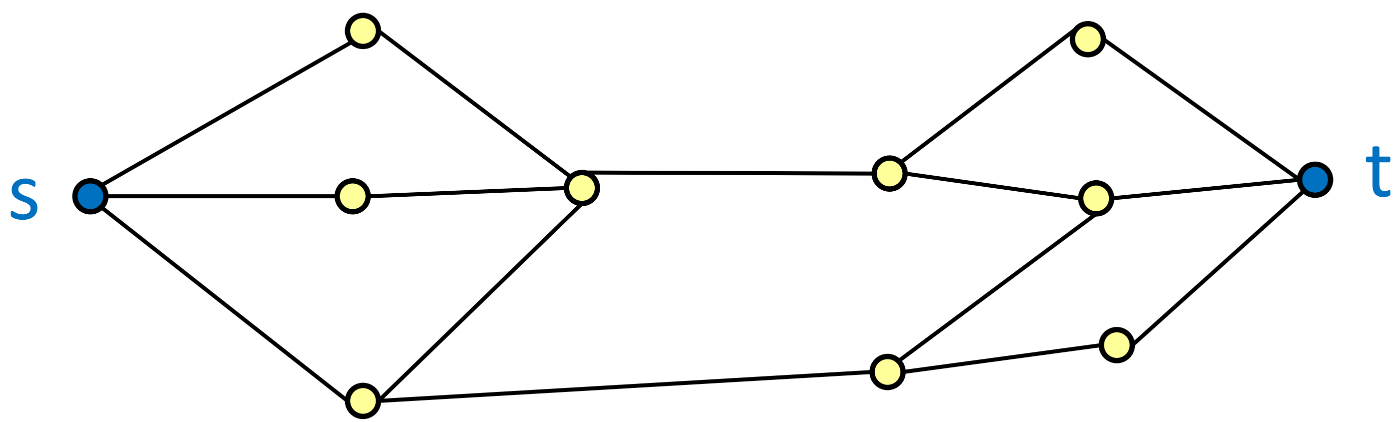
- Let's design a separation oracle for the convex set $P = \{ x : \|x\| \leq 1 \}$ = unit ball B(0,1).

> **Separation Oracle**
> Is $z \in P$?
> If not, find a vector a s.t. $a^T x < a^T z \;\; \forall x \in P$

- **Conclusion:** Since we were able to give a separation oracle for P, we can optimize a linear function over it.
- **Note:** $\max \{ c^T x : x \in P \}$ is a **non-linear program**. (Actually, it's a **convex program**.)
- Our next topic: convex analysis and convex programs!

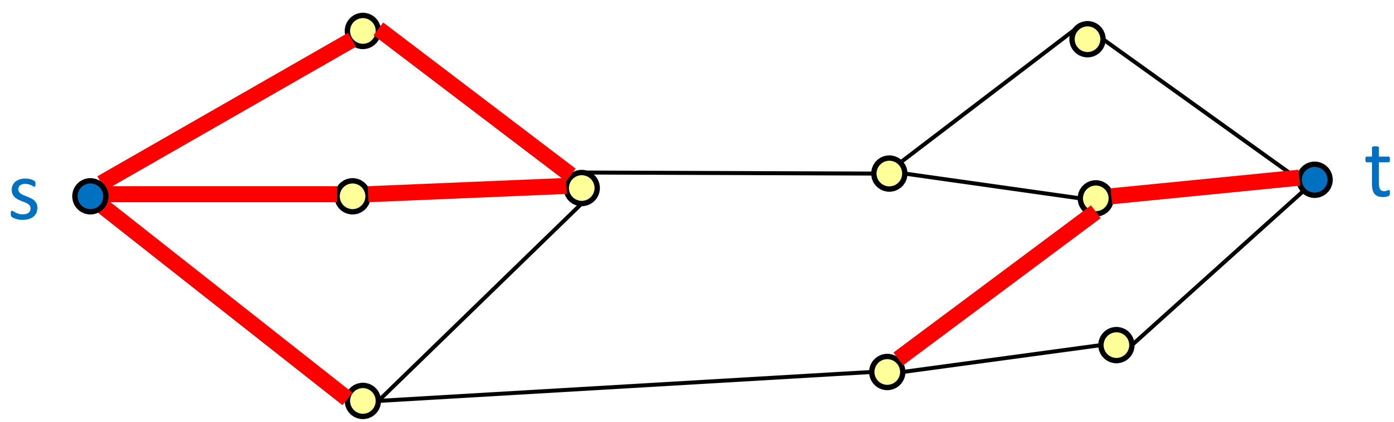
Minimum s-t Cut in a Graph

- Let G=(V,E) be a graph. Fix two vertices s,t$\in$V.
- An **s-t cut** is a set F$\subseteq$E such that, if you delete F, then s and t are disconnected
  i.e., there is no s-t path in G\F = (V,E\F).



Minimum s-t Cut in a Graph

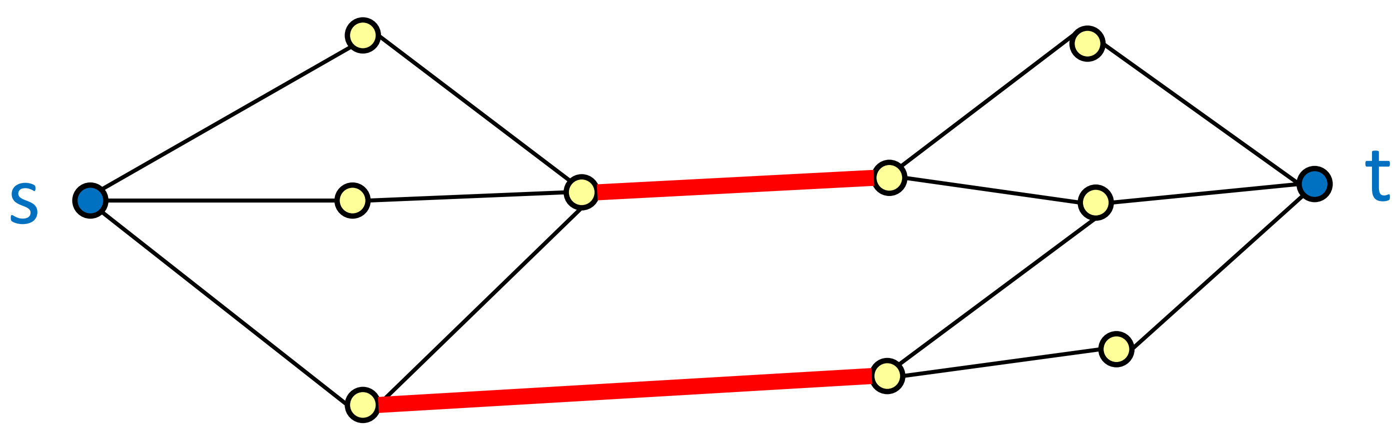
- Let G=(V,E) be a graph. Fix two vertices s,t$\in$V.
- An **s-t cut** is a set F$\subseteq$E such that, if you delete F, then s and t are disconnected
  i.e., there is no s-t path in G\F = (V,E\F).



These edges are an s-t cut

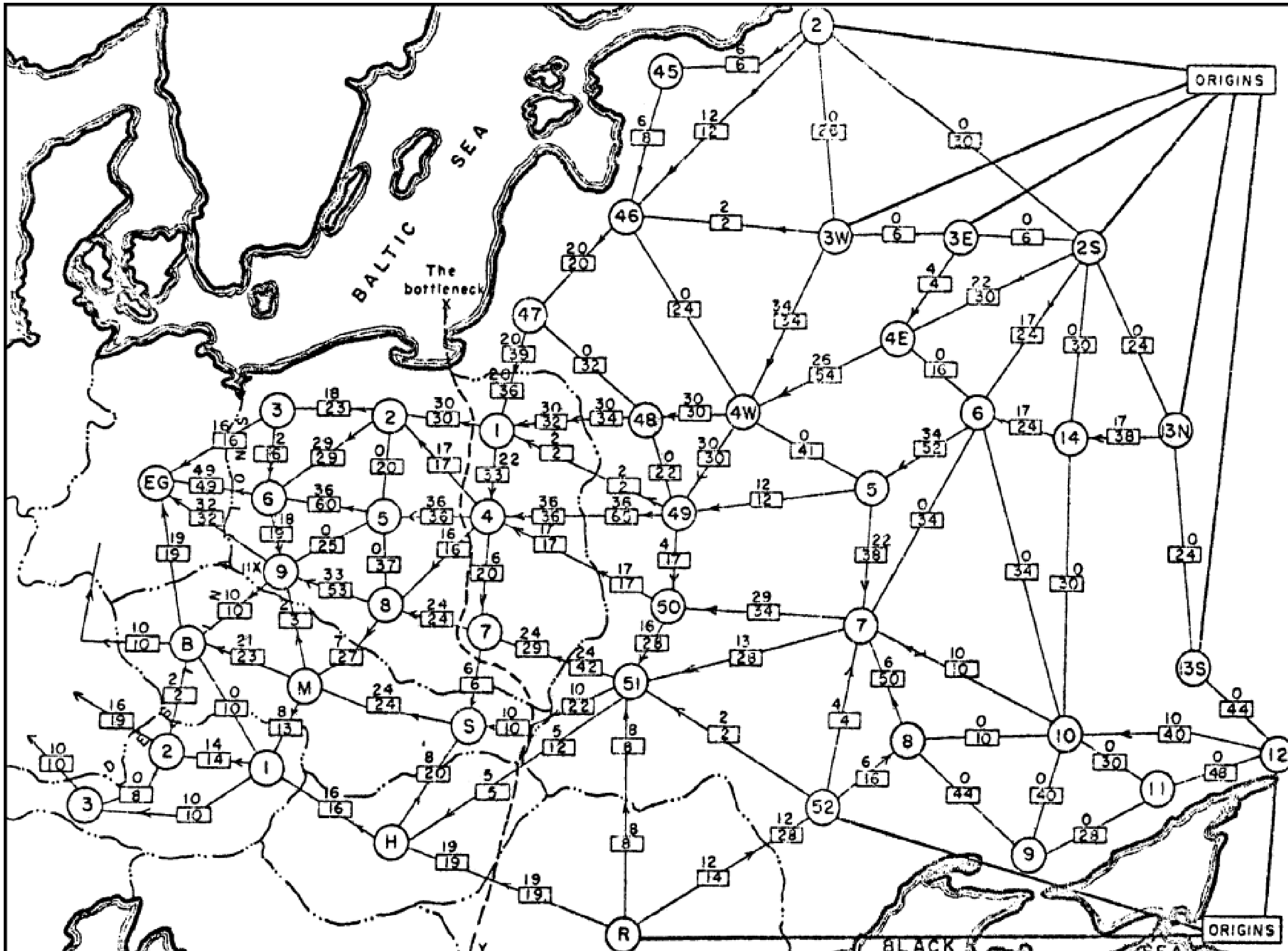
Minimum s-t Cut in a Graph

- Let G=(V,E) be a graph. Fix two vertices s,t$\in$V.
- An **s-t cut** is a set F$\subseteq$E such that, if you delete F, then s and t are disconnected
  i.e., there is no s-t path in G\F = (V,E\F).



These edges are a **minimum** s-t cut

Minimum Cut Example



From Harris and Ross [1955]: Schematic diagram of the railway network of the Western Soviet Union and Eastern European countries, with a maximum flow of value 163,000 tons from Russia to Eastern Europe, and a cut of capacity 163,000 tons indicated as `The bottleneck'.

Minimum s-t Cut in a Graph

- Let G=(V,E) be a graph. Fix two vertices s,t$\in$V.
- An **s-t cut** is a set F$\subseteq$E such that, if you delete F, then s and t are disconnected
  i.e., there is no s-t path in G$\setminus$F = (V,E$\setminus$F).
- Can write this as an **integer program**.
  Make variable x_e for every e $\in$ E.
  Let $\mathcal{P}$ be (huge!) set of all s-t paths.

$$
\begin{array}{lllr}
\min & \sum_{e \in E} x_e & & \\
\text{s.t.} & \sum_{e \in p} x_e & \geq 1 & \forall p \in \mathcal{P} \\
 & x_e & \in \{0,1\} & \forall e \in E
\end{array}
$$

Minimum s-t Cut in a Graph

- Can write this as an integer program.
  Make variable x_e for every $e \in E$.
  Let $\mathcal{P}$ be (huge!) set of all s-t paths.
- We don't know how to deal with integer programs, so relax it to a **linear program**.

$$
\begin{array}{llll}
\min & \sum_{e \in E} x_e & & \\
\text{s.t.} & \sum_{e \in p} x_e & \geq 1 & \forall p \in \mathcal{P} \\
& x_e & \geq 0 & \forall e \in E
\end{array}
$$

- **Theorem:** Every BFS of this LP has $x_e \in \{0,1\}$ $\forall e \in E$.
  (So integer program and linear program are basically the same!)
- **Proof:** Maybe later in the course, maybe in C&O 450.

Minimum s-t Cut in a Graph

$$\begin{array}{llll} \min & \sum_{e \in E} x_e & & \\ \text{s.t.} & \sum_{e \in p} x_e & \geq 1 & \forall p \in \mathcal{P} \\ & x_e & \geq 0 & \forall e \in E \end{array}$$

- How can we solve this LP?
  If graph has |V|=n, then $|\mathcal{P}|$ can be enormous!
  (Exponential in n).
- Our local-search algorithm will take a **very** long time.
- Can use Ellipsoid method, if we can give separation oracle.

Minimum s-t Cut in a Graph

$$\begin{array}{llll} \min & \sum_{e \in E} x_e & & \\ \text{s.t.} & \sum_{e \in p} x_e & \geq 1 & \forall p \in \mathcal{P} \\ & x_e & \geq 0 & \forall e \in E \end{array}$$

Separation Oracle

Is $z \in P$?

If not, find a vector a s.t. $a^T x < a^T z \quad \forall x \in P$

- Can use Ellipsoid method, if we can give separation oracle.
- If I give you z, can you decide if it is feasible?
- Need to test if $\Sigma_{e \in p} z_e \geq 1$ **for every** s-t path p.
- Think of value z_e as giving "**length**" of edge e.
  Need to test if **shortest** s-t path p^* has **length** ≥ 1.
- If so, z is feasible. If not, constraint for p^* is violated by z.

Minimum s-t Cut in a Graph

- If I give you z, can you decide if it is feasible?
- Need to test if $\Sigma_{e \in p} z_e \geq 1$ **for every** s-t path p.
- Think of value z_e as giving "**length**" of edge e.
  Need to test if **shortest** s-t path p^* has **length** ≥ 1.
- If so, z is feasible. If not, constraint for p^* is violated by z.

- **How to efficiently find shortest s-t path in a graph?**
- There are efficient algorithms that **don't** check **every** path.
  e.g., Dijkstra's algorithm. Such topics are discussed in C&O 351.
- **Another way:** Let's use our favorite trick again.
  Write down IP, relax to LP, prove they are equivalent,
  then solve using the Ellipsoid Method!

This can get crazy…

A common Linear Program relaxation of Traveling Salesman Problem

Solve by Ellipsoid Method
Separation oracle uses…

Minimum S-T Cut Problem

Solve by Ellipsoid Method
Separation oracle is…

Shortest Path Problem

Solve by Ellipsoid Method!

Everything runs in polynomial time!

Solving Discrete Optimization Problems

- **How to efficiently find shortest s-t path in a graph?**
- Let's use our favorite trick again.
  Write down IP, relax to LP, prove they are equivalent, then solve using the Ellipsoid Method!
- **Very general & powerful approach for solving discrete optimization problems.**
  Almost every problem discussed in C&O 351 and C&O 450 can be solved this way.
- **Main Ingredient:** Proving that the Integer Program and Linear Program give the same solution.
- We will discuss this topic in last few weeks of CO 355.