# Combinatorial Optimization -Comp 552 -Fall 2005

Professor Adrian Vetta
Scribe Matthew Drescher

January 31, 2006

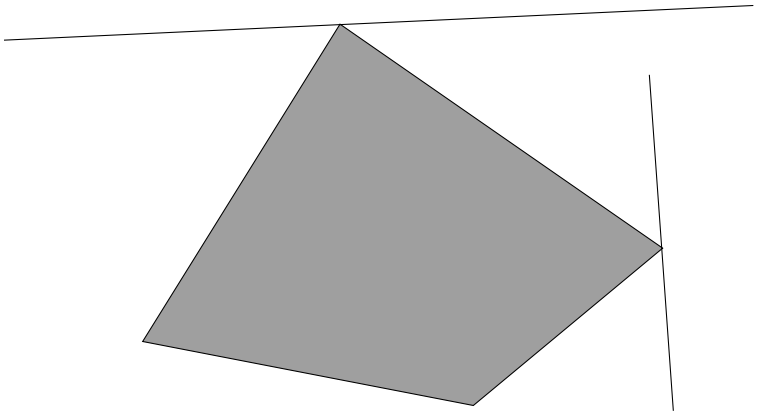

Figure 1: $P := \{x : Ax \leq b\}$

Exam 8 dec 2 pm.
We will be looking at problems of size say n. We have

$$G = (V, E)$$

if our problem is of size $poly(n)$ we are happy. if $exp(n)$ we are screwed. We want efficient = polynomial ways of solving problems.

# 1 Matchings

A matching $M$ in a graph $G$ is a set of vertex disjoint edges. We say that a $v \in V$ is **matched** if it incident to some edge in $M$. Otherwise $v$ is **unmatched**

## 1.1 Maximum matchings

Augmenting path

A path $P$ is augmenting with respect to $M$ if
(i) The endpoints of $P$ are unmatched.
(ii) Edges in $P$ alternate between $E - M$ and in $M$

Note that if $P$ is an augmenting path with respect to $M$, then

$$M' := M \cup P - M \cap P := M \oplus P$$

is also a matching with

$$|M'| = |M| + 1$$

.

**Theorem 1.** *A matching $M$ is of maximum size $iff$ $G$ contains no augmenting path with respect to it.*

*Proof.* $\Rightarrow |M \oplus P| = |M| + 1 \quad \square_{\Rightarrow}$

$\Leftarrow$
Suppose $M$ is not a maximum matching which contains no augmenting path, and let $M^*$ be a maximum matching. Now

$$M \oplus M^* \subseteq M \cup M^*$$

It is easy to see that

$$\forall v \in M \cup M^*, d(v) \leq 2$$

(otherwise at least two edges from the same matching would be incident)

Thus $M \oplus M^*$ is a set of cycles and paths.
i) can not have any odd cycles, since then two edges from the same matching would be incident
ii) cant have an augmenting path, otherwise $M^*$ is not maximum.

Thus we are restricted to even *cycles* and *even* paths, odd paths that are not augmenting with respect to $M^*$.
We cannot have odd paths that are not augmenting for $M^*$, by the assumption that $M$ does not have augmenting paths. Thus $|M| = |M^*|$ so $M$ is maximal.

□

Matching Algorithm

{
1.Start with $M = \emptyset$
2.Find augmenting path with respect to the current matching
3. Augment the current matching.
4. Repeat the above two steps as long as possible.
}

This is finite(exponential time). When it terminates we have a matching with no augmenting paths, which is maximal by Theorem 1.

## 1.2 How to find an augmenting path

Edmonds'65
We look for augmenting *walks* between vertices in the set
$S \subseteq V$ of *unmatched vertices. walk: a not necessarily simple path.*

*Blossom*–a *cycle* of length $2k + 1$ which contains $k$ matching edges.

*Flower* Consists of a *blossom* ,and an *even* length alternating path with unmatched root.

Figure 2: flower

Given a Flower with blossom $B$, let $G'$ be the graph obtained by contracting $B$ to a single vertex. Call the resulting graph and matching $G'$ and $M'$.

## 1.3 Blossom: to reduce the size of our problem

.

**Lemma 1.** *$M$ is maximum in $G$ iff $M'$ is maximum in $G'$. (Note, this is equivalent to not assuming we have a flower, but that $M$ is disjoint from the rest of $B$).*

*Proof.* $\Rightarrow$
Assume that $M'$ is not maximum in $G'$. So there exists an augmenting path $p'$ in $G'$. Then $p'$ contains $B$ as vertex. But routing around $B$ in one of the directions gives an augmenting path $p$ in $G$.

$\Leftarrow$
Suppose that $M$ is not maximal in $G$. Again we have an augmenting path $p$ which again intersects $B$ , or its easy. One of the endpoints of $p$ is not contained in the blossom.(since $B$ has at most one unmatched vertex $w$). We may assume that $w$ is unmatched. (switch on the stem, if not the case). But then $p_0$ to $w$ is an augmenting path in $G'$. Since every edge from $(B, V - B)$ is unmatched. $w$ is the unmatched vertex in $B$.

□

## 1.4 Finding a Blossom

Recall $S$, the set of unmatched vertices.
We find an $S$ to $S$ alternating walk in $G$ using the transformation:

`for every path` $(v_1, v_2)(v_2, v_3)$ `where` $(v_1, v_2)$ `is matched, and` $(v_2, v_3)$ `isn't,add directed edge` $(v_1, v_3)$ `and remove` $(v_1, v_2), (v_2, v_3)$ .
`(Really walk from S to` $N(S)$`).`

This can be done in $O(n)$ time using a *breadth first search*. BFS produces shortest paths. add vertex $r$ adjacent to all of $S$, then add $t$ adjacent to all of $N(S)$ , we can find a shortest path from $r \to t$

So we build this directed graph $\vec{G}$ using BFS.

**Lemma 2.** *shortest $S \to S$ path gives an augmenting path or a blossom.*

*Proof.* Suppose underlying walk (of the directed walk) has cycles. They can not be of even length or it gives a cycle in $\vec{G}$ i.e. not a shortest path.
An odd cycle is a blossom. Every path is augmenting. □

**Theorem 2.** *There exists a polytime algorithm for maximal matching.*

*Proof.* At most $\frac{n}{2}$ stages, since the maximum matching can contain at most $\frac{n}{2}$ edges. It takes $O(m)$ time to find Blossom, or augmenting path. Contracting blossom takes $O(m)$ time. Find at most $O(n)$ blossoms before an augmenting path. As the number of vertices falls with a contraction. Total $O(n^2m)$. □

# 2 Shortest path problem

Given a directed graph, $G = (V, A)$, with arc costs $c_a$. We would like to find a $S.P$ from a specified vertex $s$ to every other vertex. We actually consider *shortest walks*. We have 2 possibilities.
(1) We have negative cycles , then our cost is $-\infty$.
(2) There exists a $s_v$ path $\forall v$ that does not have cycles. (remove the cycles they are of cost 0)

In case (1) *The problem is NP-Hard, includes as a special case, Hamiltonian path*
Can we instead say quickly when there are negative cost cycles?

Case(2): Can be negative cost arcs..

## 2.1 Bellman-Ford

We can test for negative cycles, find SPs using B-Ford algorithm.

$$d(s) = 0, d(v) = \infty \forall v \neq s, pred(v) = \emptyset$$

Bellman-Ford Algorithm
`Repeat n-1 times:`
`for each arc` $(i,j) \in A\{$
`if` $(d_j > d_i + c_{ij})$
`then Set` $d_j = d_i + c_{ij}$; $pred(j) = i$ $\}$ .

Run Time
$m$ arcs, $n$ phases $\Rightarrow O(mn)$

**Theorem 3.** *If there are no negative cost cycles, then B-F gives S.Ps.*

*Proof.* By induction , we show that by phase $t$ , the algorithm has found the SP from $s \to v$ that uses at most $t$ arcs.

For $t = 1$, obviously true. Assume true for $t = k$. Try phase $k + 1$.
Let

$$P = s, v_1, ..., v_{k-1}, v_k = v$$

be an SP $s - v$ path of length $k + 1$. By induction , we considered the sub path $P', s \to v_{k-1}$ in the previous phase, that is $s \to_{P'} v_{k-1} \to v_k$. So we have SP from $s$ to $v_{k-1}$ using at most $k$ arcs with cost at most $c(P')$. Thus

$$d(v_k) \leq c(v_{k-1}, v_k) + c(P') = c(P)$$

□

If there is a negative cycle, then what? B-F gives shortest walks using $t - arcs$.
Run $n$ phases.$d_j^t :=$ *label at phase* $t$
(A)*if* $d_j^n < d_j^n$ *we have a negative cycle* .

*Proof.* Let $W$ be the $s - j$ walk corresponding to $d_j^n$. $d_j^{n-1}$ is the cost of the shortest $s - j$ walk using at most $n - 1$ arcs. Hence $d_j^n < d_j^{n-1}$ thus $W$ has $n$ arcs. Therefore $W$ has a cycle $C$ ,since there are only $n$ vertices with $c(C) \geq 0$. Thus $W - C$ is a walk with less then $n$ arcs and $c(W - C) \leq c(W)$. Therefore $d_j^{n-1} \leq d_j^n$ □

.

(B)Conversely if $d_j = d_j^{n-1} \forall j \in V$ so $d_j^{n+1} = d_j^n = d_j^{n-1}$ *etc...* thus $\nexists$ negative cycles.

If (B) we have SPs.
If (A) we have negative cost cycle.

Observation,non negative cycle The $d_j$ at the end of B-F satisfy

$$d_j \leq d_i + c_{ij} \forall ij \in A, (i,j) \in A$$
$$\Leftrightarrow c_{ij}^d := c_{ij} + d_i - d_j \geq 0 \forall ij \in A$$

Such a vector $d$ is a *potential*. The $c_{ij}$ are called *reduced costs*.

Shortest path $d$ are potentials, but potential $d$ is not necessarily shortest path $d$. For example, if $c_{ij} > 0$, make $\vec{d} = 0$.

SP optimality conditions
If $d_j$ are the lengths of some $s-j$ paths $\forall j$, $d_j$ are $SP-distances\ iff\ c_{ij}^d \geq 0, \forall ij \in A$.

*Note. For a cycle $C$

$$\sum_{ij \in C} c_{ij}^d = \sum_{ij \in C} (c_{ij} + d_i - d_j)$$
$$= \sum_{ij \in C} c_{ij}$$

.

**min cost to time cycle problem(tramp steamer problem)**
We have a profit $\pi_{ij}$ or arc $ij$ anda time $t_{ij}$. Want to maximize $C$

$$\frac{\sum_{ij \in C} \pi_{ij}}{\sum_{ij \in C} t_{ij}}$$

Set $c_{ij} = -\pi_{ij}$ Now we want to minimize $C$

$$\frac{\sum_{ij \in C} \pi_{ij}}{\sum_{ij \in C} t_{ij}} = u^*$$

We can solve the problem in polynomial time.

**Corollary 1.** *Can find min mean cost cycle in polynomial time.*

*Lowest level Algorithm!*
*make guess for* $u^*$
*if* $u$ *is too small guess higher*
" *big guess lower*
*if correct* "*hurrah*"
Can test this in polynomial time. Bisection search. Guess $u$, set $l_{ij} = c_{ij} - ut_{ij} \forall ij \in A$
(1) there exists a negative length cycle, then

$$\sum_{ij \in C} l_{ij} = \sum_{ij \in C} (c_{ij} - ut_{ij}) < 0$$

$$\Rightarrow u > \frac{\sum_{ij \in C} c_{ij}}{\sum_{ij} t_{ij}} \geq u^*$$

(2) All cycles are strictly positive.

$$\sum_{ij \in C} l_{ij} = \sum_{ij \in C} c_{ij} - ut_{ij} > 0 \forall C$$

$$\Rightarrow u < \frac{\sum_{ij \in C} c_{ij}}{\sum_{ij} t_{ij}} = u^*$$

so guess higher.

(3) Min length cycle C has length 0.

$$\sum_{ij \in C} c_{ij} - ut_{ij} = 0$$

$$\Rightarrow u = \frac{\sum_{ij \in C} c_{ij}}{\sum_{ij} t_{ij}} = u^*$$

guess is correct.
Testing for (1) is easy. Run B-F to see if we have negative length cycle. If not B-F gives SP distances $d_j$ from an arbitrary S.
Are we in case (2) or (3)? Set $l^d_{ij} = l_{ij} + d_i - d_j$ By SP opt conditions $l^d_{ij} \geq 0 \forall ij$
Since

$$\sum_{ij \in C} l^d_{ij} = \sum_{ij \in C} l_{ij}$$

we have no negative cycles with respect to $l_{ij}^d$ Case (2) lengths are greater then 0. Case (3) equal to 0 for some $C \Rightarrow$ all $l_{ij}^d = 0 \in C$. Consider $ij$ such that $l_{ij}^d = 0$ Look for cycle. If there is one we are in case (3), if not we are in case (2).

run time Find negative cost cycle in $O(mn)$ B-F.
Look for graph with residual $L_{ij}^d = 0$ and search for cycle in $O(m)$.

Let $deg^+(v) :=$ *# arcs leaving* $v$, $deg^-(v) :=$ " *entering.*
Assume $deg^+(v) \geq 1 \forall v$ Since the graph is finite, we must eventually visit the same vertex twice. $O(m)$

If $\exists u$ such that $deg^+(u) = 0$ we remove $u$. Update $deg^+(v)$ for each neighbor of $u$. repeat until no vertices left or each vertex has $deg^+(v) \geq 1$. We get an acyclic ordering.$O(m)$.

How many guesses do we make? $-u - - - - - - u^* - - - - - - - - u$ use a binary search . $|Avrg\ cost| \leq \bar{c}$, $\bar{c} := max|c_{ij}|$.

$$t_{ij} \in \mathbb{Z}_{++}, \frac{\sum_C c_{ij}}{\sum_C t_{ij}}$$

$$\leq \frac{\sum_C \bar{c}}{|C|} = \bar{c}$$

Let $\bar{t} := max(t_{ij})$

**Theorem 4.** *We make* $O(log(n\bar{t}\bar{c})) = O(log(n) + log(\bar{t}) + log(\bar{c}))$ *guesses by bisection method before finishing.*

*Proof.* Want to show that if internal size is $< \frac{1}{\bar{t}^2 n^2}$ then there is a $\leq 1$ feasible solution. Suppose $\exists C_1, C_2$ giving different ratios in the current interval.

$$|\frac{c(C_1)t(C_1) - c(C_2)}{t(C_2)}| < \frac{1}{\bar{t}^2 n^2}$$

$$= |\frac{c(C_1)t(C_2) - c(C_2)t(C_1)}{t(C_1)t(C_2)}| > \frac{1}{\bar{t}^2 n^2}$$

Start from $2\bar{c}$ (interval size) finish at $\frac{1}{\bar{t}^2 n^2} \Rightarrow O(log(n\bar{t}\bar{c}))$ guesses. □

# 3 Maximum Flows

## 3.1 path packing

Given a digraph $G = (V, A)$ and vertices such that

**Theorem 5.** *max # arc disjoint s-t paths = min # arcs in an s-t cut*

see diagram...

*Proof.* $G = G_0, i = 0$ Repeat until there does not exist an $s - t$ path in $G_i$. Find $P_i$ , an $s - t$ path in $G_i$. Reverse the arcs of $P_i$ to get $G_{i+1}$. $G_{i+1} = G_i \leftarrow P_i$.
Observation 1
for any s.t cut $S$ we have

$$|\delta^+_{G_{i+1}}(S)| = |\delta^+_{G_i}(S)| - 1$$

Observation 2
Let $S^*$ be the set of vertices in $G_{k+1}$ that are reachable from $s$. Then $\delta S^*$ is a min $s - t$ cut in $G$. Follows from observation 1.

So we have an algorithm to find the minimum cut. It remains to show that $P_1P_2...P_k$ "give" $k$ disjoint paths in $G$. Let $H_i :=$ set of arcs that appear in reverse in $G_i$ plus $i$ *copies of* $(t, s)$.

Claim: $H_i$ is Eulerian. (in deg = out deg $\forall v$).

*Proof.* By induction. true for $H_1$. Assume true for $H_r$. Observe that

$$H'_{r+1} := H_r + P_{r+1} + (t, s)$$

$H_{r+1} \subseteq H'_{r+1}$ and is Eulerian by induction. $H_{r+1}$ is obtained from $H'_{r+1}$ by removing an arc $a$ if $a$ and its compliment appear in $H'_{r+1}$. $H_{r+1}$ is Eulerian (just removing cycles). So we can decompose it in to cycles. Remove $k$ copies of $(t, s)$ to get $k$ disjoint $s - t$ paths. □

□

**Corollary 2.** *There exists a $O(m^2)$ algorithm to find a maximum collection of arc-disjoint s-t paths.*

*Proof.* It takes $O(m)$ time to find an s-t path in $G_i$ (by DFS,BFS). We find at most $m$ paths (allow multiple copies of arcs). □

Suppose we have a capacity $u_a$ on arc $a$. An s-t flow is a set of s-t paths $p_1, ..., p_k$, such that arc $a$ is at most $u_a$ of the $p_i$. To solve this , replace each arc $\rightarrow_{u_a}$ by $u_a$ copies.

**Corollary 3.** *Max s-t flow = Min s-t cut.*

How long does this take? $\#arcs \leq \bar{u} \cdot m$ where $\bar{u} = max\ u_a \Rightarrow O(m^2 \cdot \bar{u})$ time. We can do a bit quicker as # of iterations is $\leq min\ s-t\ cut$.

$$min\ s-t\ cut \leq n \cdot \bar{u}$$

$$\Rightarrow O(nm\bar{u})$$

Pseudopolytime, poly(m,n).Homework analyze faster method.

## 3.2 Residual Graphs

Let $P_1, ..., P_k$ be an s-t flow $f$. Let $f_a$ be the number of paths using $a$. $f_z(f_{a1}, f_{a2}, ..., f_{an})$. The residual graph $G^f$ has arcs $(i,j) \in G^f$ if $f_{ij} < u_{ij}$ (residual)capacity $u_{ij} - f_{ij}$. $(j,i) \in G^f$ if $f_{ij} > 0$ (residual)capacity $f_{ij}$.

## 3.3 Blocking Flows

Can we reduce the number of iterations? Must find more paths per iteration. Does this help? Gain in number of iterations, probably lose in time per iteration.
TRICK
`Only augment paths in` $G_i$ `for which every arc is in a shortest s-t path`
see picture "trick"

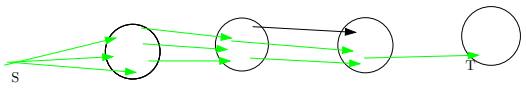


Figure 3: flower

Let $G_i^*$ be the arcs on shortest s-t paths. A blocking flow $F_i := (P_1 P_2 ... P_{ri})$ is a set of disjoint s-t paths in $G_i^*$ such that $G_i^* - F_i$ has no s-t paths.

ALGORITHM
(1) `Find` $G_i^*$--$O(m)$
(2) `Find blocking flow` $F_i$ `in` $G_i^*$--$O(m)$
(3) `Reverse arcs in` $F_i$. `Repeat on` $G_{i+1?} = G_i \leftarrow F_i$
We will show that after step (3) the shortest s-t distance increases.

$$d_{Gi}(s,t) < d_{Gi+1}(s,t) - -n - iterations$$

therefore $O(mn)$ *algorithm.*
(1) Find $G_i^*$ in $O(m)$ time.
Run BFS from s. $d(s,i) - -O(m)$
Run BFS into t. $d(i,t) - - - O(m)$
is $(i,j)$ on a shortest path? $O(1) \rightarrow total\ O(m)$
If $d(s,i) + d(j,t) = d(s,t) - 1$ YES.

(2)Find blocking flow in $G^*$ in $O(m)$ time. Observe $G_i^*$ is acyclic. Run DFS , let $A$ be the arcs we search , and $P_1$ the shortest s-t path we find.

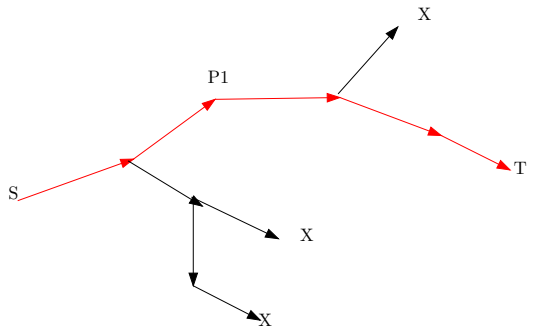


Figure 4: dfs

$A_1$ Repeat on $G_i^* - A_1$ Looking for $A_2, P_2$. Find blocking flow $F_i = P_1 \cup ... \cup P_{ri}$ in $G_i^*$. If not , then there exists $st$ $P$ in $G_i^* - F_i$. Now $P$ must use some arc of some $A_j$. Let $atP$ be the first arc of $P$ we discarded. But we discarded $(i,j)$ because we were stuck. But there is a path still from $j$ to $t$.

This only takes $O(m)$ time. We only consider each arc once. We found a path $a \in P_i$ or we discarded $a$.

(3) $d_{Gi+1}(s,t) > d_{Gi}(s,t)$ Consider $G_i \cup (reversed?)\bar{G}_i^*$ now $a \in \bar{G}_i^*$ is not in any shortest s-t path. We find a blocking flow $F_i \in G_i^*$. But $G_{i+1} \subseteq G_i \cup \bar{G}_i^*$ The shortest s-t path in $G_{i+1}$ is not in $G_i^* - F_i$ So there exists $a \in P - G_i^*$ This arc is only in a path from s to t of length greater then $d_{Gi}(s,t)$. So $d_{Gi+1}(s,t) > d_{Gi}(s,t)$. $\square$

## 3.4 Vertex disjoint paths

take each vertex $v$ replace it with an edge $(v_1, v_2)$, run the algorithm, to find an arc disjoint set, which gives us max number of vertex disjoint paths.
Application max bipartite matching.

**Theorem 6.** *there exists a $O(m\sqrt{n})$ algorithm to find maximum matching in bipartite graph.*

## 3.5 Bipartite Matchings

Suppose $M^*$ is an optimal matching, and we have a matching $M$. Take $M^* \oplus M$ is a set of even cycles and paths.

$$|M^*| = |M| + (\#\ of\ odd\ length\ paths := k)$$

$$|M| = |M^*| - k$$

We have $k$ vertex disjoint s-t paths in $G^f$ where $f$ is a flow for $M$.

**Theorem 7.** *there exists an $O(\sqrt{n}m)$ algorithm for bipartite matching.*

*Proof.* We use the Blocking Flow algorithm.
(i) After $i = \sqrt{n}$ phases, we have $d_{Gi}(s,t) \geq \sqrt{n}$. Some $D_j$ has size $\leq \frac{n}{\sqrt{n}} = \sqrt{n}$. Any $D_j$ is an s-t cut. because any left-right moves from $D_j$ to $D_{j+1}$. i.e. there exists an s-t vertex cut containing at most $\sqrt{n}$ vertices. Therefore we have at most $\sqrt{n}$ vertex disjoint paths from $s$ to $t$. Therefore $|M| \geq |M^*| - \sqrt{n}$ from here I use at most $\sqrt{n}$ more phases. Thus $2\sqrt{n}$ phases. Thus we have $O(\sqrt{n}m)$ algorithm. $\square$

## 3.6 Min cost Flows

Cost $c_{ij}$ is per unit flow.

(*1) $b_v = flow_{out}(v) - flow_{in}(v)$
(*2)

$$\sum_{u \in \delta^+(v)} f_{vu} - \sum_{u \in \delta^-(v)} f_{uv} = b_v$$

.

$G = (V, A)$ , arc cost $c_{ij}$, arc capacity $u_{ij}$, node supply / demand $b_v$.
eg, $b_s = 1, b_t = -1, b_v = 0, v \in V - \{s, t\}$.

Find min cost:= $\sum_{ij \in A} c_{ij} f_{ij}$, $f$ satisfying (*).

Generalizes max flow problem. maximize $val(f)$ subject to:
flow $b_s = val(f)$, $b_t = -val(f)$, $b_v = 0$

We can assume only $b_s, b_t \neq 0$.

$$b_s = \sum_{v: b_v > 0} b_v$$

$$b_t = \sum_{v: b_v < 0} b_v$$

Observe, if max flow = $b_s$ then there is a solution. If max flow is less then $b_s$ there is no solution. Residual graphs help us here too. Given flow $f$ of value $b_s$, then $G^f$ is $(i, j) \in G^f$ if $f_{ij} < u_{ij}$ capacity: $u_{ij} - f_{ij}$ cost: $c_{ij}$
$(j, i) \in G^f$ if $f_{ij} > 0$ capacity: $f_{ij}$ cost : $-c_{ij}$

Suppose we find a cycle $C$ in $G^f$. Pushing flow along $C$ still gives a flow.
$Flow\ out - Flow\ in = b_v$
If cost $C < 0$ in $G_f$ then pushing flow around(augment) $C$ saves money.

$$f \to f',\ cost(f') = cost(f) + cost(c)$$

Keep augmenting on negative cycles

$$f_1 \to^{c1} f_2 \to^{c2} f_3 \to^{c3} ... f_{*optimal}$$

If $f_i$ is not optimal is there a negative cycle $C_i$ in $G^{fi}$?

**Theorem 8.** *$f$ is a min cost flow iff $G^f$ has no negative cost cycle.*

*Proof.* If negative cost cycle , then $f$ is not optimal. Suppose $G^f$ has no negative cost cycle. Take the optimal flow $f^*$ and consider $f^* - f . f^* - f$ Decomposes into positive and negative arcs. Suppose
$(u_{ij} \leq) f^*_{ij} > f_{ij} \Rightarrow ij \in G^f \; cost \; c_{ij}$ ,
$0 \leq f^*_{ij} < f_{ij} \Rightarrow ji \in G^f \; cost \; -c_{ij}$.
It follows that $f^* - f$ decomposes into cycles in $G^f$. Therefore

$$cost(f^*) = cost(f) + \sum cost(cycles)$$

thus there exists a negative cycle in $G^f$ or $f$ is optimal. □

This gives us an algorithm, repeatedly augment on negative cycles. Cost of initial flow is at most $\bar{c}\bar{u}$ i.e. at most $\bar{c} := max \; c_{ij} \cdot \bar{u} := max \; u_{ij}$ iterations. Find min cost cycle is NP-hard.

Find minimal mean cost cycle in poly-time.

## 3.7 min-flow cont

Start with $f = 0$
Find min mean cost cycle in $G^f$. Augment $C$ as much as possible $f' \leftarrow f$. Note tight(bottle neck) arc is not in $G^f$. Let $\mu^f$ be the min mean cost of a cycle in $G^f$. The algorithm terminates when $\mu^f \geq 0$.
The following claims will help us analyse the algorithm:

**Claim 1.** *if $\mu^f > \frac{-1}{n}$ then $f$ is optimal.*

*Proof.* Any negative cost cycle has cost at most $-1$, and at most $n$ arcs. □

**Claim 2.** *If $f'$ is obtained from $f$ by augmenting $C$ then $\mu^{f'} \geq \mu^f$.*

*Proof.* Let $C$ be the min mean cost cycle in $G^f$, $C'$ in $G^{f'}$. Suppose

$$\frac{c(C')}{|C'|} < \frac{c(C)}{|C|}$$

$f'$ differs from $f$ only on arcs in $C$. Now $C'$ uses some arc not in $G^f$. Otherwise $C'$ was min mean cost cycle in $G^f$. Let

$$B := \{a \in C' : a \notin G^f\} \neq \emptyset$$

$$\bar{B} := \{\bar{a}_{reverse} : a \in B\}$$

Observe that $\bar{B} \subseteq C$ ( (*) did something to arcs in $\bar{B}$ for their reverses to appear in $G^f$ ) Consider the circuit(set of arcs where in deg = out deg) $C \cup C'$. The mean cost of these is

$$\frac{c(C) + c(C')}{|C| + |C'|} < max(\frac{c(C)}{|C|}, \frac{c(C')}{|C'|}).$$

note
$min\frac{a_i}{A_i} \leq \frac{\sum_{i=1}^k a_i}{\sum_{i=1}^k A_i} \leq max_i \frac{a_i}{A_i} = \mu^f$ now $C \cup C'$ can be decomposed into cycles. These cycles are $C_1 C_2 ... C_k$ all in $G^f$ plus 2-cycles with one edge in $B$, the other in $\bar{B}$ by (*). Cost of *digons* $= 0$ so $c(C) + c(C') = \sum_{i=1}^k c(C_i) + 0$. Hence

$$0 \geq \mu f > \frac{c(C) + c(C')}{|C| + |C'|} \geq \frac{\sum_{i=1}^k c(C_i)}{\sum_{i=1}^k |C_i|}$$

So

$$\mu^f > \frac{\sum c(C_i)}{|C_i|} \geq min_i \frac{c(C_i)}{|C_i|}$$

but $C_i \in G_f ... \Rightarrow \Leftarrow$. □

**Claim 3.** *After $m$ augmentations the $f'$ we obtain has $\mu^{f'} > (1 - \frac{1}{n})\mu^f$*

*Proof.* Set $L_{ij=c_{ij}-\mu^f}$. There are no negative mean length cycles, therefore no negative cycles. So there exist shortest paths with respect to $l_{ij}$ ( from $S$) plus shortest path distances $d$. So $d_j \leq d_i + l_{ij} = d_i + (c_{ij} - \mu^f)$. Observation: Set $C_{ij}^d = c_{ij} + d_i - d_j$ then $C_{ij}^d \geq \mu^f$. $C_{ij}^d \geq c_{ij} + d_v - (c_{ij} + d_i - \mu_f)$.
$c_{ij}^d$ "reduced cost". Recall

$$\sum_{ij \in C} c_{ij} = \sum_{ij \in C} c_{ij}^d$$

Each time we augment we "saturate" an arc and it gets removed from the residual graph.

$$-C_{ij}^d = -C_{ji}^d$$

If every edge has reduced cost $\geq 0$ in $G^f$ then $G^f$ has no negative cycles. Suppose we augment on $C$ and $C$ only has arcs with $c_{ij}^d < 0$. Suppose this happens $2m$ times in a row. We remove at least 1 arc and add arcs with

$c_{ij}^d > 0$. After $m$ or all arcs have $c_{ij}^d > 0$ and $f$ is optimal. Once every $m$ steps $\geq 1$ arc in $C$ has $c_{ij}^d > 0$.

$$\frac{c(C)}{|C|} = \frac{\sum_{ij \in C} c_{ij}}{|C|} = \frac{\sum c_{ij}^d}{|C|} \geq \frac{(|C|-1)\mu f + 1}{|C|}$$

$$= \frac{|C|-1}{|C|} \cdot \mu^f$$

$$\geq (1 - 1/n)\mu^f$$

□

We can find min mean cost cycle in time

$$O(mn(logn + log\bar{c}))$$

Every $m$ steps we improve $\mu^f$ by factor $(1 - \frac{1}{n})$
" $m \cdot n$ " ....." $\approx e^{-1}$
i.e. done in $log(n\bar{C}\bar{U})$ of these. i.e. Run time $O(m^2n^2log(n\bar{c}\bar{u}))$ (weakly polynomial time). In fact the alg is strongly polynomial

$$O(m^3n^3logn)$$

see Schrijver.

# 4 Polyhedral Methods

$$min\ cx$$

such that

$$Ax \leq b$$

$$x \geq 0$$

A polytope is

$$\{x : Ax \leq b\}$$

is integral. Assume its bounded. If each "vertex" of the polytope is integral. A vertex $x$ is a point of $P$ such that it cant be expressed as a convex combination of points in $P$.

It is the solution to $N$ linearly independent equations $A^*x = b^*$ where $x \in \mathbb{R}^N$. where $A^* \subseteq A, b^* \subseteq b$.

Observe for integral $P$ an optimal solution to $max(cx : x \in P)$ occurs at an integral point(i.e. a vertex). We want to solve $max(cx : Ax \leq b, x \in \mathbb{Z}_+)$ we can relax to obtain $max(cx : Ax \leq b, x \geq 0)$ by LP in poly time if $P$ is integral, we have solved our original problem.

## 4.1 The Matching problem

The incidence vector $\chi^n$ of a matching $m$ is defined as

$$\chi_e^m = 1 \; if \; e \in m, \; 0 \; if \notin m$$

*The perfect matching polyope* $PM(G)$ is the convex hull of incident vectors of perfect matchings in $G$. Matching polytope $M(G)$ is the convex hull of $^m$ for all matchings $m$.

Let

$$P(G) := \{x : x_e \geq 0, \sum_{e \in \delta(v)} x_e = 1\}$$

**Theorem 9.** *For bipartite Graphs,* $P(G) = PM(G)$

*Proof.* $PM(G) \subseteq P(G)$ since its vertices are all the perfect matchings. $^M \in P(G) \forall p.m \; M$ . Want to show that

$$P(G) \subseteq PM(G)$$

The smallest counter example $G$ such that $P(G) \nsubseteq PM(G)$ has a vertex of $P(G), x$, such that $x \notin PM(G)$.
i) $x_e > 0$ otherwise throw $e$ away and $G - e$ is a counter example.

$$x \in P(G - e)$$

but

$$PM(G - e) \subseteq PM(G)$$

ii) $x_e < 1 \forall e$
if $x_e = 1$ remove $u, v$ to have a smaller counter example. $x$ restricted to $G - u - v$ is a convex combination of $PMs$ in $G - u - v$ therefore $x$ is a convex combination of $PMs$ in $G$. So $x_e > 0$ is fractional. Therefore

$deg(v) \geq 2 \forall v \in G$ Therefore $G$ has a cycle(M1,M2) which is even because its bipartite. There exists an $\epsilon$ such that

$$x^1 = x + \epsilon(^{M1} - ^{M2}) \in P(G)$$

$$x^2 = x - \epsilon(^{M1} - ^{M2}) \in P(G)$$

But

$$x = \frac{1}{2}x_1 + \frac{1}{2}x_2$$

Which contradicts the assumption that $x$ is a vertex. □

So using $LPs$ we find a max weight PM. We can $max(wx|\sum_{e\in\delta(v)} x_e > 1, x \geq 0)$ .
Let

$$Q(x) = \{x|x_e \geq 0 \forall e, \sum_{e\in\delta(v)} x_e \leq 1, \forall v\}$$

**Theorem 10.** *For bipartite graphs $M(G) = Q(G)$*

*Proof.* Clearly $M(G) \subseteq Q(G)$. We want to show that any vertex $x$ of $Q(G)$ is a convex combination of matchings. let $G'$ be the auxilery graph $G'$ obtained by taking a copy of $G$ and edges connecting $v'$ and $v$ with weight $1 - \sum_{e\in\delta(v)} x_e$. Note $\sum_{e\in\delta(v)} x'_e = 1 \forall v \in G'$.

$$x' \in P(G') = PM(G')$$

therefore $x'$ is a convex combination of perfect matchings in $G'$ . Restrict $x'$ to $G$ to see that $x'$ is a convex combination of perfect matchings in $G$. □

## 4.2 Matching Polytope: General Graphs

PM(G)

$$P(G) = \{x : x_e \geq 0 \forall e, \sum_{e\in\delta(v)} = 1 \forall v, \sum_{e\in\delta(S)} \geq 1 \forall S, |S| \ odd \ \geq 3\}$$

**Theorem 11.** *$PM(G) = P(G)$ for non-bipartite graphs.*

*Proof.* $PM(G) \subseteq P(G)$ by A.C . Meta theorem...

Take smallest counter example with $P(G) \nsubseteq PM(G)$ so

$$\exists e.p.\ x \in P(X) - PM(G)$$

i) $x_e > 0 \forall e$, as before, by minimality.
ii) $x_e < 1 \forall e$ as before, by minimality.
Therefore every edge has degree at least 2. Suppose $deg(v) = 2 \forall v$. Then $G$ is the union of disjoint cycles , all of which are even of (*) is violated. Even cycles are bipartite, so $x \in PM(G)$ by the previous theorem.

suppose $\exists v, deg(v) \geq 3$ then

$$2m = \sum_v deg(v) \geq 2n + 1$$

iii) $m > n$
iiii) $n$ is even or we violate the definition of $P(G)$ for $S = V$.
Thus we can rewrite

$$P(G) = \{x : x_e \geq 0 \forall e, \sum_{e \in \delta(v)} = 1 \forall v, (*) \sum_{e \in \delta(S)} \geq 1 \forall S, |S||V - S| \geq 3 \; for \; |V| \; even\}$$

$x$ is a vertex so $x$ satisfies $m$ linearly independent constraints with equality. Thus there exists an odd $S$, $|S| \geq 3, |V - S| \geq 3, s.t, \sum_{e \in \delta(S)} x_e = 1$. Consider $G_1 \to x^1$ obtained by contracting $V - S$,$G_2 \to x^2$ obtained by contracting $S$. $x^1$ satisfies constraints for $P(G^1)$. So $x^1$ is a convex combination of PMs in $G^1$ as $|G^1| < |G|$. Similarly $x^2$ is a convex combination of PMs in $G^2$.

$$x^1 = \frac{1}{k_1} \sum_{i=1}^{k1} m_i^1$$

$m_i^1$ is pm in $G^1$

$$x^1 = \frac{1}{k_2} \sum_{i=1}^{k2} m_i^2$$

we can assume $k_1 = k_2$. Let $e \in \delta(S)$ . Then the # of matchings $M_i^1$ containing

$$e = \#\ of\ matchings\ M_i^2\ containing\ e$$

Only one edge in $\delta(S)$ appears in any $M_i^1$ *or* $M_i^2$.

$e_1 \in \delta(S), e_2 \in \delta(S), e_3$ set $M_i = M_i^1 \cup M_i^2$. This is a perfect matching in $G$ and $x = \frac{1}{k}\sum^{Mi}$. thus $x$ is a convex combination of PMs in $G$. □

## 4.3 Matching polytope

$$Q(G) = \{x : x_e \geq 0 \forall e, \sum_{e \in \delta(v)} \leq 1 \forall v, \sum_{e \in \delta(S)} \leq 1 \forall S \ odd\}$$

$$\sum_{e \in E(S)} x_e \leq \frac{|S|-1}{2} \forall S \ odd$$

**Theorem 12.** $Q(G) = M(G)$

*Proof.*

$$x^M \in Q(G) \forall matchings \ min \ G$$

Want $Q(G) \subseteq M(G)$ pick vertex $x \in Q(G)$, make 2 copies of $G$ as before, get $x'$ as before. $G, x \to G', x'$ So $\forall v \in V, \sum_{e \in \delta(v)} x'_e = 1, x'_e$ Want to show that $x'$ is a convex combination of PMs in $G'$. If so then , as before , $x$ is a convex combination of matchings. Need $x'$ to satisfy

$$\sum_{e \in \delta(s)} x'_e \geq 1 \forall odd \ S \in G'$$

**Claim 4.**

$$x'(\delta(S)) \geq x'(\delta(S_1 - S_2)) + x'(d(S_2 - S_1))$$

*Proof.* $x'(W) = \sum_{e \in W} x'_e$, $S = S_1 \cup S_2$. $S_1 - S_2$ is the set of vertices in $S_1$ minus the copies that are in $S_2$. We have 4 types of vertex. $v \in S_1 - S_2, v \in S_1 \cap S_2, v \in S_2 - S_1, v \notin S_1 cup S_2$. $S', G'$ mirrors of $S, G$ so each edge on RHS contributes same to LHS. □

$$|S| = 2k + 1 = |S_1 - S_2| + |S_2 - S_1| + 2|S_1 \cap S_2|$$

one of the first to summands must be odd. Assume WLOG that $T = |S_1 - S_2|$ is odd.

$$|T| = \sum_{v \in T} x'(\delta(v))$$

$$= x'(\delta(T)) + 2x'(E(T))$$
$$= x'(\delta(T)) + 2x(E(T)) \leq x'(\delta(T)) + |T| - 1$$
$$\Rightarrow x'(\delta(T)) \geq 1 \Rightarrow x'(\delta(S)) \geq 1$$

□

$max(cx|x \in Q(G))$ has an exponential number of constraints! This makes it a bit tricky to solve by conventional methods.

# 5 Weighted Matching in Bipartite Graphs

We are dealing now with Bipartite graphs $(A, B)$ with weights on the edges.

## 5.1 Hungarian Method

This is a general method that includes the augmenting path approaches we have seen. $A_u, B_u$ are unmatched vertices in $A, B$ respectively . Multiply cost of edges that arent in the matching by $-1$, and direct the other edges towards $A$, call the resulting graph $G'$. Find a shortest path from $A_u$ to $B_u$ in $G'$. Let $M' = M \oplus P$.

Algorithm
`Start with` $M_0 = \emptyset$
`Repeat until` $A_u - B_u$ `path in` $G'$
`Find shortest` $A_u - B_u$ `path` $P_i$
$M_{i+1} = M_i \oplus P_i$

Let $M_0, M_1, ..., M_{n/2}$ be the matchings we obtain. We claim that the max weight of these is a max weight matching. Towards this goal, we say $M$ is *extreme* if $M$ has max weight amongst all matchings of size $|M|$.

**Theorem 13.** *Each $M_i$ is extreme.*

*Proof.* True for $M_0, M_1$ Let $M_{i+1} := M_i \oplus P_i$ i.e. ($M_i$ *is extreme*). Take $M$ of size $|M_{i+1}| = i + 1$.

$$|M| = |M_i| + 1$$

thus

$$M_i \oplus M$$

has an augmenting path $P$ .

$$M_i = M \oplus P$$

Since $P_i$ is shortest path

$$L(P_i) \leq L(P)$$

Now $M \oplus P$ is a matching of size $i$ Therefore

$$w(M \oplus P) \leq w(M_i)$$

Putting together

$$\begin{aligned} w(M_{i+1}) &= w(M_i) - L(P_i) \\ &\geq w(M_i) - L(P) \\ &\geq w(M \oplus P) - L(P) = w(M) \end{aligned}$$

Hence $M_{i+1}$ is extreme. □

Is the algorithm polynomial time? $n/2$ iterations, shortest paths, do we have negative cycles in $G'$. A negative cycle is even alternating $C$. $M_i \oplus C$, is a new matching, $M$, with $|M_i| = i$ *edges* of weight $w(M_i) - L(C) > w(M_i)$ but $M_i$ is extreme.

$$Run\ time \Rightarrow O(n^2 m)$$

Use data structures to get $O(nm + n^2 log(n))$.

# 6 Linear Programming

An $LP$ is a problem of the following form:

$$max(cx | Ax \leq b, x \geq 0)$$

A vast number of other problems can be formulated as $LPs$. An $LP$ can be solved in polynomial time in $m, n$.
Lower bound: Any feasible $x$ gives $opt \geq c^t x$

Upper bound: add constraints so that their coefficients are greater then $c$. Thus we are

$$min(yb | yA \geq c, y \geq 0)$$

This gives us
**Weak Duality**

*Proof.*

$$cx \leq yAx \leq yb$$

□

**Farkas Lemma 1.** *Take an $m \times n$ matrix $A$, and an $m-$vector $b$.* $(x : Ax = b, x \geq 0)$ *has a solution iff* $\forall w, wA \geq 0 \Rightarrow b^t w \geq 0$.

*Proof.* Geometrically, if $Ax \neq b \forall x \geq 0$ then $b$ is not in the cone generated by the columns $a_1, ..., a_n$ of $A$.

$$cone(A) := cone(a_1, ..., a_n) := \{\lambda_1 a_1 + \lambda_2 a_2 + ... + \lambda_n a_n, \lambda_i \geq 0 \forall i\}$$

if $b$ is not in $cone(A)$ then we must be able to find a hyperplane which separates $b$ from $cone(A)$.i.e. a *separating hyperplane* $h$ such that $b$ and $cone(A)$ are on different sides of $h$.

Let $w$ be the normal to $h$. So $w^t b < 0$ and $w^t a_i \geq 0 \forall i$. □

**Farkas Lemma 2.** $(x : Ax \leq b, x \geq 0)$ *has a solution* $iff \forall w \geq 0, wA \geq 0 \Rightarrow b^t w \geq 0$.

*Proof.* Let $A' := [A, I], x' = [x, z]^t$ Apply Farkas Lemma (1).

$$(x : A'x' = b, x \geq 0) \text{ has solution } iff \forall w (A')^t w \geq 0 \Rightarrow b^t w \geq 0$$

this is saying

$$[A, I][x, z]^T = b = Ax + z \leq b \Leftrightarrow Ax \leq b, x \geq 0$$

Therefore

$$[A^T, I]w \geq 0 \Rightarrow b^T w \geq 0$$
$$A^T w \geq 0, w \geq 0 \Rightarrow b^T w \geq 0$$

**Strong Duality 1.** *If there exists an optimal solution*

$$cx^* = y^* b$$

*Proof.* Want feasible $x, y$ such that $cx \geq yb$.

$$\exists \text{ solution to } (x : Ax, x \geq 0) \; iff \; \forall w \geq 0 wA \geq 0 \rightarrow wb \geq 0$$

We want $x, y \geq 0$ $s.t.(*)$ :

$$[A, 0;\ 0 - A^T;\ -c^T\ b^T]^T [x\ y]^T \leq [b\ -c\ 0]^T$$

(*) is true iff (A)

$$[A^T\ 0\ -c;\ 0\ -A\ b;]^T[w_1\ w_2\ x]^T \geq 0, \forall w_1, w_2, x \geq 0$$

this implies

$$\hat{b}^T w \geq 0$$
$$\hat{b}^T w := [b^T\ -c^T\ 0][w_1,\ w_2\ \lambda]^T \geq 0$$
$$b^T w_1 - c^T w_2 \geq 0$$
$$b^T w_1 \geq c^T w_2$$

(B)

$(A) \Rightarrow (B)$ then we are done.

$$A^T w_1 - c \geq 0$$
$$-Aw_2 + b \geq 0$$
$$w_1, w_2, \lambda \geq 0$$

we know

$$A^T w_1 \leq \lambda c$$
$$Aw_2 \leq \lambda b$$

Want to show

$$b^T w \geq c^T w_2$$

Case 1:

$$\lambda > 0$$
$$b^T w_1 = \frac{1}{\lambda}\lambda b^T w_1$$
$$\geq \frac{1}{\lambda}(Aw_2)^T w_1 = \frac{1}{\lambda} w_2^T A^T w_1$$

$$\geq \frac{1}{\lambda} w_2^T \lambda c = w_2^T c$$

Case 2:

$\lambda = 0$

Take the feasible $x_0, y_0$ (exist by assumption)

$$Ax_0 \leq b, x_0 \geq 0$$

$$A^T y_0 \geq c, y_0 \geq 0$$
$$w_1^T b \geq w_1^T A x_0 = (A^T w_1)^T x_0 \geq \lambda c^T x_0 ..?? erasedlastpart$$
$$w_2^T c \leq w_2^T A^T y_0 = (A w_2)^T y_0$$
$$\leq \lambda b y_0 = 0 \leq w_1^T b$$

□

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

# 7 Applications of LP duality

## 7.1 Shortest s-t paths

Consider the following relaxation:

$$min \sum_{e \in E} c_e x_e$$

such that

$$\sum_{e \in \delta^-(v)} x_e - \sum_{e \in \delta^+(v)} x_e$$
$$x_e \geq 0$$

$$= -1, for\ v = s,\ = 0\ for\ v \neq s, = 1\ for\ v = t$$

We claim that this "relaxation" is in fact modeling the entire problem, and not adding new feasible area. This can be seen since the vertex of the polytopes are integral. Any solution to primal is a fractional s.t. flow. Any flow decomposes into $\leq m$ paths ( and cycles in general). So we have flow $= P_1, P_2, ..., P_k$, with weights $\lambda_1, \lambda_2, ..., \lambda_k \geq 0$. Thus $\sum_{\lambda_i} = 1$. So

$$Flow = \sum_i \lambda_i P_i\ \lambda_i \geq 0, \sum_i \lambda_i = 1$$

$$\sum_{e \in E} c_e x_e = \sum_i \lambda_i c(P_i)$$

so there exist path $P_j$ such that $c(P_j) \leq \sum_{e \in E} c_e x_e$ So solving LP gives optimal solution.

Consider the dual of LP.

$$min\ c^T x$$

$$Ax = b$$

$$x \geq 0$$

$A$ is the vertex, edge incidence matrix. the dual is

$$max \sum_{v \in V} b_v y_v$$

$$A^T y \leq c$$

$$y\ unrestricted$$

we have constraints

$$\forall (i,j) \in E, -y_i + y_j$$

and we are maximizing $y_t - y_s$ . Add constant to each $y_v$ has no affect. Assume $y_s = 0$. Now we have got

$$max\ d_t$$

$$s.t.\ d_j \leq d_i + c_{ij}\ \forall e = (i,j)$$

$$d_s = 0$$

We can see that we are actually maximizing the shortest path distances. $d_t \leq$ min cost s-t path = distance of $t$ from $s$.

$$d_i = distance\ of\ i\ from\ s$$

$$\Rightarrow max\ d_t = min\ cost\ path\ s - t$$

$\Rightarrow$ Shortest path from $s - t$ has length equal to distance $t\ from\ s$. This is nonsense! well in fact this information gives us some insight into algorithms...primal dual algorithms. $P - D$ gives us something else here. Given $d$ set

$$s_p = \{v : d_v \leq p\} \forall p = 0, 1, ..., dist(t-1)$$

Consider the cuts

$$\delta^+(S_0), \delta^+(S_1), \delta^+(S_2)$$

each arc $e = (i, j)$ is in at most $c_e$ of these cuts.

$$d_j \leq d_i + c_{ij}$$

otherwise

$$d_j > d_i + c_{ij}$$

So

**Lemma 3.** *Length of shortest $s - t$ path equals the max packing of s-t cuts $\delta^+(c_0), ..., \delta^+(c_{12})$ such that e is in at most $c_e$ cuts.*

this is not nonsence!.rather interesting.

## 7.2 Max Flow

we have

$$max\ s - t\ flow\ problem$$

$$max\ f = \sum_{e \in \delta^-(t)} x_e - \sum_{e \in \delta^+(t)} x_e$$

such that

$$\sum_{e \in \delta^-(v)} x_e - \sum_{e \in \delta^+(v)} x_e = -f\ for\ v = s, \bar{0}\ for\ v \neq s, t, = t,\ for\ v = t$$

$$x_{ij} \leq v_{ij} \forall ij \in A, x_{ij} \geq 0\ \forall ij \in A$$

find dual. This is integral (opt solutions given by our combinatorial algorithms are integral). Find dual $A =$ the edge incidence matrix on top of the edge identity matrix. we dont need some constraints we can just have

$$\sum_{e \in \delta^-(v)} x_e - \sum_{e \in \delta^+(v)} x_e = 0\ for\ v \neq s, t$$

find dual $y = [z, w]$

$$min\ b^T y = \sum_{ij \in A} u_{ij} \cdot w_{ij}$$

$$z_i\ unrestricted\ , w_{ij} \geq 0$$

$$A^T y \geq c$$

$c_e = 1\ if\ e \in \delta^-(t), 0\ otherwise$. Dont need arcs coming out of $t$ or going into $s$. So we never have $-1$ in $c$. So

$$min \sum_{e \in E} u_e w_e$$

$$s.t. \ -Z_i + Z_j + w_{ij} \geq 0$$

$$Z_j + W_{sj} \geq 0$$

$$-Z_j + w_{jt} \geq 1$$

$$w_{ij} \geq 0$$

Now $Z_s, Z_t$ dont appear but we could add them back in as shown. Set $Z_s = 0, Z_t = -1$ Now add 1 to all $Z's$

$$min_e u_e w_e$$

$$s.t. Z_i \leq Z_j + w_{ij} \forall i, j \in E$$

$$Z_t = 0$$

$$Z_s = 1$$

# 8 Dual of Max Flow

$$min \sum_{e \in E} u_e w_e$$

$$s.t.\ z_i \leq z_j + w_{ij}\ \forall ij \in E$$

$$z_t = 0, z_s = 1, w_{ij} \geq 0 \forall ij$$

What do solutions to $D$ look like? Note that $z, w$ have the following property? Given $w_{ij}$ the$z_i$ are shortest path distances to $t$. The distance from $s$ to $t$ is 1. From last time( shortest path-cut packing theorem)

$$min\ S.P = Max\ packing\ of\ cuts(s-t)\ s.t.\ (i,j)\ in\ atmost\ w_{ij}\ cuts$$

(weighted version) i.e. cuts $\delta^+(S_1), \delta^+(S_2)$.... weights $\lambda_1, \lambda_2 ... \sum(\lambda_i) = 1, \lambda_i \geq 0$.

$$min \sum_{ij \in E} u_{ij} w_{ij} \geq \sum_{ij \in E} u_{ij} ( \sum_{S: ij \in \delta^+(S)} \lambda_s)$$

$$= \sum_s \lambda_s \sum_{ij \in \delta^+(s) u_{ij}}$$

$$= \sum_s \lambda_S Capacity(+(s))$$

convex combination of cut capacities.

$$Max\ Flow = Min\ dual \geq \sum_s \lambda_s capcity(s)$$

but $max\ flow \leq capacity\ of\ any\ s-t\ cut$. So $max\ flow = min\ cut$. Our dual variables are $z = \chi_s$, $w = \chi_{\delta^+(s)}$, $w_{ij} = 1$ if $ij \in \delta^+(S)$, 0 otherwise.

# 9 Ellipsoid Method

Suppose we wish to optimize a linear function $cx$ over some convex body $K$. It suffices to find a feasible point in $K$. Basic idea, impose $cx \geq \alpha$, if there exists a point in $K \cap \{x : cx \geq \alpha\}$ choose larger $\alpha$. Else choose smaller $\alpha$. Bisection search then gives solution in polytime. (To as small $\epsilon$ as you want $poly(n, m, 1/\epsilon)$). We know this already for $LP's$ i.e. $feasibility = optimality$. Find $(x, y)$ such that

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

feasible for this, is optimal for $P$ by duality. The ellipsoid can find feasible points in $K$ (under some conditions on $K$). Here we show it works for polyhedra of the form $Ax \leq b$. This leads to polytime algorithm for $LP$. (Assume here $P$ is bounded)

$$P := \{x | Ax \leq b\}$$

Suppose $P \subset E_1$ an ellipsoid with center $z_1$. if $z_1 \in P$ then we are done, if not in $P$ then we can separate $z_1$ from $P$, so $a_i z_1 > b_i$ for some constraint $a_i x \leq b$. Let $H_1 := \{x : c_i x \leq a_i z_1\}$ So $P \subseteq E_1 \cap H_1$. Now find $E_2$ the smallest(volume) ellipsoid containing $E_1 \cap H_1$. See if $z_2$ center of $E_2$ is in $P$. Repeat...

For this to work quickly we need
(1) find $E_1$.

(2) in polytime check feasibility of give violated constraint(separating hyperplane).
(3) find $E_2$ given $H_1, E_1$.
(4) Use polynomial number of iterations. ($E_i$ shrink quickly).

Note $P \subseteq E_i \forall i$ as $E_i$ shrink we terminate with feasible point if $P > 0$.

## 9.1 Background(proof method is not tested material)

An ellipsoid $E(A, a)$ is defined as

$$E(A, a) = \{x : (x - a)^T A^{-1}(x - a) \leq 1\}$$

where $a$ is center and $A$ is positive definite. The following are equivalent:
(1) $A$ is p.d.
(2) $A$ is symmetric and $x^T Ax > 0 \forall x \neq 0$
(3) $A^{-1}$ is p.d.
(4) $A = BB^T$ for non singular $B$ (write $B = \sqrt{A}$).

Unit ball is $E(I, 0)$. We need

**Lemma 4.** $E(A, a) = \sqrt{A}E(I, 0) + a$

*Proof.* Suffices to show $E(A, 0) = \sqrt{A}E(I, 0)$

$$\begin{aligned} x^T x &= (\sqrt{A}^{-1}\sqrt{A}x)^T(\sqrt{A}^{-1}\sqrt{A}x) \\ &= x^T\sqrt{A^T}\sqrt{A^{-1}}^T\sqrt{A^{-1}}\sqrt{A}x \\ &= x^T\sqrt{A^T}(\sqrt{A}\sqrt{A}^T)^{-1}\sqrt{A}x \\ &= (x\sqrt{A}^T)A^{-1}(\sqrt{A}x) \\ &= y^T A^{-1} y \end{aligned}$$

where $y = \sqrt{A}x$. □

We now prove the algorithm works. First the ellipsoids shrink "quickly".

**Theorem 14.** *if $E'$ is smallest ellipsoid containing $E(A, a) \cap \{x : c^T x \leq c^T a\}$ then*

$$\frac{Vol(E')}{Vol(E)} \leq e^{\frac{-1}{2(n+1)}}$$

*Proof.* Using linear transformations we may assume that $E(A,a) = E(I,\alpha)$(volume ratios preserved under linear transformations). By symmetry assume $c = (1,0,0,0,0,...,0)$. It is well known that $E(A',a')$

$$a' = \frac{-1}{n+1}\frac{c}{\sqrt{c^T 2}} = (-\frac{1}{n+1}, 0, 0, ..., 0)$$

$$A' = \frac{n^2}{n^2-1}(I - \frac{2}{n+1}\frac{cc^T}{\sqrt{c^T c}}) = \frac{n^2}{n^2-1}M$$

$$M := \begin{matrix} 1-\frac{2}{n+1} & \text{a0} \\ 0 & \text{a...} \end{matrix}$$

now

$$\frac{Vol(E')}{Vol(E)} = \frac{|det\sqrt{A'}|V_n}{V_n} = |det\sqrt{A'}| = \sqrt{det(A')}$$

$$det(A') = \frac{n^2}{n^2-1}^{n}(1 - \frac{2}{n+1}) = \frac{n^2}{n(\frac{n-1}{n+1})}$$

$$= \frac{n^2}{(n+1)(n-1)}^{n-1}\frac{n}{n+1}^{2} = (1+\frac{1}{n^2-1})^{n-1}(1-\frac{1}{n+1})^2$$

$$\leq e^{\frac{n-1}{n^2-1}} \cdot e^{\frac{-2}{n+1}} = e^{\frac{1}{n+1}}e^{-2} \cdot something = e^{-\frac{1}{n+1}}$$

□

(1) Know ellipsoids shrink by $e^{\frac{-1}{2(n+1)}}$
$P$ has "large" volume, initial ellipsoid,$E_0$ is not too big.

(i) Let $\mu$ be the number of bits used to store an $n \times n$ sub-matrix of $A$ (and corresponding part of $b$). So $E_0 = E(2^\mu I, 0)$ contains $P$. (ii)

**Lemma 5.** $Vol(P) \geq 2^{-n^3\mu}$

*Proof.* Assume $P$ is full dimensional so it contains $n+1$ affinely independent vertices, $x_0, ..., x_n$. $Vol(P)$ is bigger than the volume of the simplex $con.hull(x_0, ..., x_n)$. the formula for the volume of the simplex is $\frac{1}{n!}det(M)$ where $M$ is $n \times n$ **1** with the row $x_0, ..., x_n$.

What is a vertex? $x_i$ is a solution to subset of $n$ rows of $[A : b]$, $A_i x_i = b_i$ . So by Crammers rule , $x_{ij} = \frac{|A_{ij}|}{|A_i|}$

We can substitute this into $M$ to obtain

$$\frac{1}{n!}\frac{1}{|A_0||A_1|...|A_n|}det(M)$$

$det(M) \geq 1$ due to affine independence, and integrality.

$$\geq \frac{1}{n!}\frac{1}{|A_0||A_1|...|A_n|}$$

Now $|A_i| \leq n!2^{n\mu}$. So

$$Vol(P) \geq \frac{1}{(n!)^{n+1}}\frac{1}{2^{n+1}} \geq \frac{1}{n^{\mu n+1}}\frac{1}{2^{\mu n+1}}$$

$$\geq \frac{1}{2^{\mu n^3}}$$

□

Count the number of iterations...

$$Vol(E_t) \leq 4^{\mu n} e^{\frac{-t}{2n}}$$

$$\leq 2^{2\mu n - \frac{t}{2n}} \leq e^{-\mu n^3}$$

Set

$$t = O(n^4\mu)$$

Now $\mu = n^2 L$ where $L$ is the number of bits/entry. It takes $O(mn)$ time to test feasibility (linear time to find ellipsoids).$O(mn^7 L)$ this is more like $O(mn^3 L)$ if we are more careful (problem: used $\sqrt{s}$ in finding ellipsoids. Get around it by rounding the center point and slight blowing up the ellipsoid(still contains $P$)).

## 9.2 Applications

Note, to use the ellipsoid method we only need to
(i) state yes $x \in P$.
(ii) state no $x \notin P$ and give a violated constraint, in polytime.

It may be possible to do (i) even if the number of constraints is exponential.

**Weighted Matching**

(P)

$$max\ cx$$

$$(1)\ x_e \geq 0$$

$$(2)\ \sum_{e \in \delta(v)} x_e = 1$$

$$(3)\ \sum_{e \in \delta(S)} x_e \geq 1\ \forall S, |S|\ odd$$

(P) is integral. Solve $P$ to get max $PM$. But $P$ has exponential number of constraints. (1),(2) are easy to check. What about (3)? Given $x$ see if $x$ satisfies (3) or give a violated constraint. (3) relates to min cuts... Let $x_e$ be a capacity on $e$. Find *min cut* $\delta(S^*)$ using max flow algorithm. Try for all pairs $(s,t)$.

$x(\delta(S^*)) \geq 1$ for all odd cuts $\geq 1$. $\Rightarrow$ (3) is satisfied. So $x(\delta(S^*)) < 1$ if $|S^*|$ is odd we have a violated constraint. What if $|S^*|$ is even?

**Lemma 6.** *let $S^*$ be a min cut. Then there exists a min odd cut $S$ such that $S \subseteq S^*$ or $S \subseteq V - S^*$.*

*Proof.* There are 4 types of vertices. In $S^* \cap S$, $V - S^* \cap S$ ,$S^* \notin S$, $V - S^* \notin S$ WLOG assume $|S \cap S^*| = even$, $|S \cap V - S^*| = odd$.

$$x(\delta(S^*)) + x(\delta(S)) \geq x(\delta(S \cap S^*)) + x(\delta(V - S - S^*))$$

by minimality

$$x(\delta(S^*)) \leq x(\delta(V - S - S^*))$$

So $x(\delta(S)) \geq x(\delta(S \cap S^*))$. So $S \cap S^*$ is a min odd cut. □

So find min cut(even) $S^*$ and split into 2 problems, $S^* \rightarrow G_1, V - S^* \rightarrow G_2$ For $G_1$ we contract $S^*$, for $G_2$ we contract $V - S^*$ $|G| > |G_1|$, $|G| > |G_2|$ as $S^*, V - S^*$ are both even. Suppose min odd cute $\subseteq S^*$. know $S$ contains at least one vertex of $S^*$, $S$ contains at least 1 vertex of $S^*$. Look for min $s - t$ cut where $s, t \in S^*, \forall pairsinS^*$ (So dont find cut $(V - S^*, S^*)$) min cut we find has value $\leq$ min odd cut. If odd cut then done. If even cut, recurse. Repeat until we find odd cut. Can't miss $S$ as it is subset of even cuts.

Use recursion.

$$f(n) = n_{max\ flows} p(n)_{time\ for\ max\ flow} + f(|S^*| + 1) + f(n + 1 - |S^*|)$$

this is polynomial. So we have a separation oracle. (what if we find a fractional optimal solution? see hmw).

# 10 Shortest r-arborescences

Given a directed graph $G = (V, A)$ and a root $r$, an $r - arborescence$ is a directed spanning out tree rooted at $r$. Costs $c_a$ on each arc. Find min cost $r - arb$. For undirected graphs finding $MST$ is easy. Here things are a little harder.

**Min-Max relation**

We say $\delta^-(S)$ is an $r - cut$ if $r \notin S$. Assume integral costs. Assume $c_a \geq 0 \forall a$ (if not add a constant to every arc).

**Theorem 15.** *Min cost of $r - a - b$ = Max packing of r-cuts. (arc is in at most $c_a$ of the r-cuts).*

*Proof.* Show $Max \leq Min$. Let $T$ be an $r - arb$. Make $c_a$ copies of each arc. Take packing $S = \{S_1, ..., S_T\}$. $|\delta^-(S_i) \cap T| \geq 1$ as there exists a path from $r$ to each vertex in $T$. $Cost(T)$ = number of arcs in picture. Remove one arc copy for each arc in $\delta^-(S_i) \cap T \forall i$ Then we remove atleast 1 arc per cut so $\#Cuts \leq \#arcs = cost(T)$.

Show $Max \geq Min$

by induction $\sum_{a \in A} c_a$. Let $A_0$ be the zero cost arcs. If $A_0$ contains an $r - arb$, the $|packing| = 0 \geq cost(T^*) = 0$. So assume $A_0$ has no $r - arb$. Consider the strongly connected components of $G[A_0]$. There is at least 2 strongly connected components or there exists a 0 cost $r - arb$.

$r$ is in a strongly connected component that doesnt have a path to all other strongly connected components. There exists a strongly connected component $S$ that we can not reach from $r$. So $\delta^-(S)$ contains no arcs from $A_0$. So every arc $\delta(S)$ has cost $\geq 1$.

Let $c'_a = c_a - 1 \geq 0,\ if\ a \in \delta^-(S),\ and\ c_a\ otherwise.$ So by induction there exists an $r - arb\ T$ with $c'(T) = R$, and there exists $S' = \{S_1, ..., S_k\}$ of r-cuts such that $a$ is in at most $c_a$ of the cuts in $S'$. Suppose $|T \cap \delta^-(S)| \geq 2$. Then for any $a \in \delta^-(S)$ we have that $(T - a) \cup A_0$ contains an $r - arb\ T'$ , this has cost $c'(T') = c'(T) - c'_a \leq c'(T)$ So $|T \cap \delta^-(S)| = 1$. $c(T) = c'(T) + 1 =$

$k+1$. $(S' = \{S_1, ..., S_k\}) \cup S$ is a packing of size $k+1$. This is a valid packing as any $a \in \delta^-(S)$ it was in at most $c_a - 1$ of the cuts in $S'$. Any arc $a \notin \delta^-(S)$ is not in $\delta^-(S)$. □

## 10.1 r-a- polytope

$(P_G)$

$$min \sum_{a \in A} c_a x_a$$

$$\sum_{a \in \delta^-(S)} x_a \geq 1 \forall r - cuts\ S$$

$$x_a \geq 0$$

Given $r - arb\ T$. Clearly $\chi^T$ is a feasible solution to $P(G)$. **Claim** Any minimal vertex of $P(G)$ is an $r - a - b$.

$$P(G) = conv\{\chi^T : T\ is\ r - arb\} + \mathbb{R}^m_+$$

*Proof.* Suppose $P(G)$ is not integral. So there exists some non integral solution $x$. So there exists integral cost function $c$ (scale if neccessary) So opt with respect to $c$ only occurs at fractional solution $x$.

$$min(c^T x : x \in P(G))$$

$$c^T x < min_{T\ is \mathring{-} arb}\ c^T \chi^T = max_{S\ packing w.r.t\ c} |S|$$

$$c^T x = \sum_{a \in A} c_a x_a$$

$$\geq \sum_{a \in A} ( \sum_{S \in' S' : a \in \delta^-(S)} 1) x_a$$

$$= \sum_{S \in' S'} \sum_{a : a \in \delta^-(S)} x_a$$

$$\geq \sum_{S \notin' S'} 1 = |' S'|$$

which is a contradiction. □

So $P(G)$ is integral $(= conv(\chi^T + \mathbb{R}_+^M))$.
**polytime algo**
Seperate $\sum_{a\in\delta^-(S)} x_a \geq 1$ $(*)$
We can solve this by ellipsoid method. Given $x$ test $(*)$ in polytime? Use $x_a$ as capacities. Find max flow from $r$ to $v$ $\forall v \neq r$, i.e. run $n-1$ times. Get min $r-v$ cut if all $min(s) \geq 1$ then feasible. If not $r-v$ cut of value 1, value of cut $= \sum_{a\in\delta^-(S)} x_a < 1$ $\square$.

# 11 Max cut problem

Given $G = (V, E)$ with weights $w_e$ on edge $e$, find $S \subseteq V$ such that $\sum_{e\in\delta(S)} w_e$ is maximized. Consider the following (IP)

$$max \sum_{e\in\delta(S)} w_e x_e$$

$$s.t.(1) \sum_{e\in F} x_e - \sum_{e\in C-F} x_e \leq |F| - 1, \forall circuits\ C\ and\ all\ odd\ F \subseteq E(C)$$

$$(2)\ x_e \in \{0,1\} \forall e \in E$$

Claim that solutions to this are cuts(or subsets of cuts), (i.e. edge sets are bipartite).
i) let $\chi_{\delta(S)}$ be incidence vector for cut $S$. So take any circuit $C$ and add $F \subseteq E(C)$ if at most $|F|-1$ edges of $|F|$ are in $\delta(S)$ then (1) is satisfied.

So suppose $F \subseteq \delta(S)$ , $|C \cap \delta(S)|$ is even, $|F|$ is odd, thus there exists an edge in $(C-F) \cap \delta(S)$ so $\sum_{e\in C-F} x_e \geq 1$. So (1) holds.
ii) Take an integral vector $E'$ satisfying (1). Show that $E'$ is a subset of a cut. i.e. show $E'$ is bipartite If not then $E'$ has an odd cycle $C'$, consider $C'$ and $F' = C'$ then

$$\sum_{e\in F'} x_e - \sum_{e\in F'-C} x_e = \sum_{e\in F'} x_e = |F'|$$

So $E'$ is bipartite. The (maximal) vertices of the polytope are cuts. Relax to $0 \leq x_e \leq 1$ and solve LP. Can we separate these constraints in polytime? Given $x$, $0 \leq x \leq 1$ is easy to check. How do we check (1)? Take two copies of the graph $G, G'$ where $x_{u'v'} = x_{uv}$ but now add cross edges $\forall (u,v) \in$

$E$, *add* $(u, v'), (v, u')$ and $d_{uv} = x_{uv}$, $d_{u'v} = 1 - x_{uv}$ A circuit is a walk from $S$ to $S'$. For each $s$, find a shortest path $P_s$ from $s$ to $s'$. $P_s$ corresponds to a circuit $C$ in the original graph $G$. Let $F$ be the set of edges in $P_s$ from $G$ to $G'$. now

$$d(P_s) = \sum_{e \in F}(1 - x_e) + \sum_{e \in C-F} x_e = |F| + \sum_{e \in C-F} - \sum_{e \in F} x_e$$

$$|F| + \sum_{e \in C-F} - \sum_{e \in C} x_e \geq 1$$

if $d(P_s) < 1$ we have a violated constraint. Moreover if there is an $F, C$ that violates (1) then there exists a violating $P_s$ path. So we can solve the relaxation in polytime.

Can we show (LP) is integral? No. Max cut is NP-hard. But what if we restrict the class of graphs? for example planar graphs

**Theorem 16.** *CutPolytope(G) = LP relaxation iff G contains no $K_5$ minor.*

**Corollary 4.** *CutPolytope(G)= LP relaxation if G is planar.*

**Corollary 5.** *max cut is polysolvable in planar graphs*

# 12 Graph Connectivity

We say a digraph $G = (V, A)$ is $k-arc$ connected if there exists $k$ arc disjoint paths from $i$ to $j$ $\forall i, j \in V$. We have seen that

**Theorem 17.** *$G$ is $k - arc$ connected iff $Min - Cut = k$.*

Similarly $G$ is $k - vertex\ connected$ if there are $k$ vertex disjoint paths from $i \to j \forall i, j$. using the transformation $a \to v \to d \Rightarrow a' \to v \to v' \to d$ for all such $a, d$.

**Menger 1.** *$G$ is $k - vertex$ connected iff min seperator $= k =$ vertex cut*

We will consider undirected graphs here. We have corresponding definitions of $k - edge$ connectivity and $k - vertex$ connectivity. using the transformaion $(u, v)_{undirected} \Rightarrow (u, v)_{directed}, (v, u)_{directed}$, our min max results also hold.

How quickly can we find a mincut?
-All pairs:max flow $O(n^3 m)$

-$s$ to every $t \in V$, $O(n^2 m)$
as $s$ is separated from some $t$ in the min-cut.
Can wedo better? (allow multiple edges). Let $r_{ij}$ =min size of $i-j$ cut.
$mincut = min_{ij} r_{ij}$, $d(i,u) = \#edges\ from\ i\ to\ u$.
Order vertices $v_1, v_2, ..., v_n$ such that $v_j$ has max number of edges tp $\{v_1, ..., v_{j-1}\}$ amongst $V - \{v_1, ..., v_{j-1}\}$ i.e. $v_j$ maximizes $d(v, \{v_1, ..., v_{j-1}\})$
Claim:

$$r_{v_{n-1}, v_n} = deg(v_n) = d(v_n, \{v_1, ..., v_{n-1}\})$$

*Proof.* Take a $v_{n-1} - v_n$ cut $S$. Let $x_0 = v_1$, $x_i = v_{f(i)}$ where $f(i)$ is the smallest index $> i$ such that $S$ separates $v_i$ and $v_{f(i)}$ and $v_{f(i)}$. Now $d(x_i, \{v_1, ..., v_{i-1}\}) \leq d(x_{i-1}, \{v_1, ..., v_{i-1}\})$. Either $x_i = x_{i-1}$ or $x_{i-1} = v_i$. $|\delta(S)| \geq \sum_{i=1}^{n-1} d(x_i, v_i)$

$$\begin{aligned} &= \sum d(x_i, \{v_1, ..., v_i\}) - d(x_i, \{v_1, ..., v_{i-1}\}) \\ &\geq \sum_i d(x_i\{v_1, ..., v_i\} - d(x_{i-1}\{v_1, ..., v_{i-1})\} \\ &= d(x_n, (v_1, ..., v_{n-1))-d(x_0, \emptyset)} \\ &= d(v_n, \{v_1, ..., v_{n-1}\}) = deg(v_n) \end{aligned}$$

□

if $v_{n-1}, v_n$ are separated by min cut $S^*$ then $|\delta(S^*)| = r_{v_{n-1}, v_n} = deg(v_n)$ So $S^* = \{v_n\}$ is a min cut. Otherwise $v_n, v_{n-1}$ are on the same side of the min cut. Contract them together. Repeat on new graph $G'$. Repeat until 2 vertices left. If we don't find min cut in steps 1...$i$ then $S^*$ is still consistent with mergings by step $i+1$.

# 13 Graph Connectivity-Gomory-Hu-Trees

So we can find a min cut in time $O(mn)$ in an undirected graph $= min_{i,j}(r_{ij})$. If $r_{ij}$ is edge connectivity from $i$ to $j$, how quickly can we find $r_{ij}$ and a min $i-j$ cut, for all pairs $i, j$. Clearly this can be done in time $O(n^2 f(n))$ where $f(n)$ is time for min-cut algorithm. However, we can do better...

A Gomory-Hu tree is a tree $T$ on $V$ such that $\forall (i,j) \in E(T)$, $\delta(T_{ij})$ is a min $i-j$ cut. $T_{ij} = component\ of\ T - (i,j), containing\ i$

**Theorem 18.** *For all graphs and edge capacities, $u_e$, there exists $G-H$ tree.*

Before proving the theorem, observe that infact a $G - H$ tree tells us about all pairs $i, j$ not just $(i, j) \in E(T)$.

**Lemma 7.** *Let $T$ be a G-H tree and take any $s, t \in G$. If $s = v_0, v_1, ...., v_r = t$ is the $s - t$ path in $P$ and if $(v_i, v_{i+1})$ minimizes $r_{v_i,v_i+1} = r_{i,i+1}$ on the path then $r_{s,t} = r_{i,i+1}$ and $\delta(T_{i,i+1})$ is a min s-t cut.*

*Proof.* Take any set of vertices $s = u_0, u_1, ..., u_p = t$, (not necessarily a path) we have the "triangle inequality"

$$(*) r_{st} \geq min_i r_{u,u+1}$$

Any s-t cut separates some $u_i$ from $u_{i+1}$ so $(*)$ follows. For $G-H$ take $v_i, v_{i+1}$ as in Lemma statement. $\delta(T_{i,i+1})$ is an s-t cut so $r_{s,t} \leq r_{i,i+1} = min_p r_{j,j+1}$ By (*) $r_{st} \geq min_p r_{j,j+1}$. □

Pick any pair $s, t \in V$ and find *min $s - t$ cut* $S$. we have got a partial tree

$$T := (S, V - S)$$

Label edge $(S, V - S)$ by $r_{st}$. Given a partial tree plus edge labels, pick any node $Z$ with at least 2 vertices in it. Contract each $C_i$ into a vertex. Call this graph $G'$. Find min $x - y$ cut in $G'$ say $\delta(S)$. Let $X = Z \cap S$ $Y = Z \cap (V - S)$, $x \in X, y \in Y$. Replace $Z$ by $X$ and $Y$. Label $(X, Y)$ by $r_{xy}$. Place $C_i$ adjacent to $X$ if $C_i \subseteq S$ or adjacent to $y$ if $C_i \subseteq V - S$. Repeat until all nodes of $T$ contain 1 vertex of $V$.

To show this gives a $G - H$ tree we need the following lemma:

**Lemma 8.** *let $\delta(S)$ be a min $s - t$ cut. For any $u, v \in S$ there exists min $u - v$ cut $\delta(U)$ with $U \subseteq S$.*

*Proof.* Assume $s \in U$, otherwise switch labels $u$ and $v$.
we have 2 cases: (i) $t \in U$, (ii) $t \notin U$. let

$$|\delta(A)| = \sum_{e \in \delta(A)} u_e$$

There are 6 types of arcs $|\delta(U)|, |\delta(S)|, |\delta(U \cap S)|, |\delta(U \cup S)|, |\delta(U - S)|, |\delta(S - U)|$
pictoral... (i) $t \in U$ as $S$ is a min s-t cut.

$$|\delta(U - S)| \geq |\delta(S)|$$

therefore $|\delta(U)| \geq |\delta(S-U)|$ $\delta(S-U)$ is a $u-v$ cut so it is a min $u-v$ cut.

case (ii) $t \in U$

$$|\delta(U \cup S)| \geq |\delta(S)| \Rightarrow |\delta(U)| \geq |\delta(S \cap U)|$$

so $\delta(S \cap U)$ is a min $u-v$ cut. □

Now we prove the theorem.

*Proof.* We show alg gives a $G-H$ tree. We say $x, y$ are representatives of edge $(X, Y) \in T$, if label of $(X, Y)$ is $r_{xy}$. Claim is true for $(S) \rightarrow_{r_{st}} (V'-S)$. Proceed by induction: look at when $Z$ splits into $X$ and $Y$. around an $x-y$ cut. Why is a min cut in $G' = r_{xy}$? By the above lemma a min $x-y$ cut in $G$. If we contract $C_1$ we still have some min $x-y$ cut available then if we contract $C_2$ there is still an $x-y$ cut available etc... $\Rightarrow x, y$ are representatives for $(X, Y)$
What about the edges adjacent to $Z$ ?

Previously $w, v$ were representatives for edge $(B, Z)$ If $v$ is in $X$ then $w, v$ is still a rep for $(B, X)$ Claim $x, w$ are reps for $(B, X)$ i.e. $r_{x,w} = r_{wv}$. $(B, Z)$ originally separates $x$ and $w$ so

$$r_{xw} \leq r_{vw}$$

Want $r_{xw} \geq r_{vw}$.. Let $G^y$ be the graph obtained by contracting $y$ to a vertex $y'$. Then $r_{xw} \geq r^y_{xw}$ by lemma 3, where $r^y$ is connectivity in $G^y$ by triangle inequality (*)

$$r^y_{xw} \geq min r^y_{xy'}, r^y_{y',w}$$

want $\geq r_{vw}$. But $v \in Y$ so $r^y_{y'w} \geq r_{vw}$
Finally

$$r^y_{xy'} \geq r_{xy} \geq r_{vw}$$

as min $x-y$ cut separates $V$ and $W$. So $r_{xw} = r_{vw}$ □

We only use$n-1$ min cut algorithms so run time is $O(nf(n))$.

# 14 T-Joins

Given an undirected graph $G = (V, E)$ and an even sized subset $T \subseteq V$ we say that $J \subseteq E$ is a $T - Join$ if (*) $T$ is exactly the odd(odd degree) set of vertices in $G' := (V, J)$.

Now given $T$ and edge costs $c_e$ how do we find a min cost $T - Join$ ?

**Lemma 9.** *Any minimal $T - Join$ is the union of $\frac{|T|}{2}$ edge paths pairing the nodes in $T$.*

*Proof.* $J$ is acyclic as removing a cycle keeps the parities the same, so $J$ is a forest. Any tree has at least 2 leaves (as any tree induced by $T$ has at least 2 vertices), say $u, v$. All odd vertices in the tree $R$ are also in $T$. So we have an even number $\geq 2$. Take the path $P_{uv}$ in $R$ from $u$ to $v$. Consider $J' := J - P_{uv}$ is a $T' - join$ where $T' := T - u - v$(as removing $P_{uv}$ changes only the parities of $u$ and $v$). By induction $J'$ is a collection of disjoint paths between pairs in $T'$. Adding $P_{uv}$ gives disjoint paths for $T$. □

***Remark***
*Any set of $\frac{|T|}{2}$ paths pairing vertices in $T$ is a $T - join$.*

*Now if $c_e \geq 0 \forall e$ then a min cost $T - Join$ is minimal.*

**Theorem 19.** *there exists a polytime algorithm for $T - Join$ problem if $c \geq 0$.*

*Proof.* Form a complete graph $H$ on $T$. For $i, j \in T$, the weight $w_{ij}$ of $(i, j) \in H$ is the shortest path distance from $i$ to $j$ in $G$. Find a min cost perfect matching in $H$. This gives us a collection of $\frac{|T|}{2}$ paths. Removing cycles(including 2-cycles), we still have a solution of less cost. Every cycle must have cost of 0 otherwise we end up with a solution that gives a cheaper $P.M$ □

*What if some of the $c_e$ are negative? Convert to a problem with $c_e \geq 0 \forall e$ Pick all negative cost edges (call them $E^-$) to be in $J$. If $E^-$ is a $T - Join$ we are done, if not $E^-$ has some parities wrong. So we have a new problem in which the parities have changed. Let $T' := T \oplus \triangle$. We have a $T' - join$ problem with $c'_e = |c_e|$. Choosing an edge in $E^-$ now corresponds to removing it. i.e. $e \notin J$, so cost is $|c_e|$. Let $J'$ be a $T' - Join$, then $J \oplus E^-$ is a $T - Join$ ($J' \cup E^-$ is $T - join$-remove-digons).*
*Want: $J' \oplus E^-$ to be min cost $T - Joint$.*

*Let $J$ be a $T-Join$. Then $J \oplus E^-$ is a $T'-Join$. So*

$$c(J \oplus E^-) = c'(J') + c(E^-) \leq c'(J \oplus E^-) + c(E^-)$$

*As $J'$ is mincost $T'-join$,*

$$= c(J)$$

*So $J' \oplus E^-$ is a min cost $T-Join$* □

So we can solve $T-Join$ problem for any $c$.

**Corollary 6.** *There exist a polytime algorithm for finding the max cost $T-Join$*

*Proof.* set $c_e = -c_e$ □

## 14.1 Applcations

(1) Testing for negative cost cycles in an undirected graph. There is a negative cost cycle in $G$ if and only if there exists an $\emptyset - Join$ with cost $< 0$.(i.e. $T = \emptyset$).

(2) Shortest path problem in undirected graphs. Finding SPs in directed graphs (with non-neg cycles) is easy. For undirected graphs us the transformation $(u,v)_{undirected} \Rightarrow (u,v)_{directed}, (v,u)_{directed}$, but obviously this wont work if we have negative edge weights. But, if we have negative edges we can solve it using $T-Joins$. Set $T := \{s,t\}$ min cost $T$ join is $s-t$ path plus cycles. If there are no negative cycles then we have the min cost $s-t$ path. If there are negative cycles then we cant say anything about the cost of the path. (if we could solve ham path problem which is NP-hard).

i.e. Can solve in polytime Shortest $s-t$ path if there does not exist negative cost cycles(we can have negative edges).

# 15 T-Joins and T-Cuts

We say $\delta(S)$ is a $T-Cut$ if $|S \cap T|$ is odd. Clearly every $T-Cut$ intersects every $T-join$.

**Theorem 20.** *If $G$ is bipartite then min size of $T-join$ is equal to the maximum packing of disjoint $T-Cuts$.*

*Proof.* max $\leq$ *min*, each $T - Cut$ hits a different edge in min T-Join. -Show min $\leq$ max. (The max packing occurs for a cross -free family of cuts $(S_1, S_2, ..., S_k)$). cross free: one must be true
(1) $S_i \cap S_j = \emptyset$, (2) $S_i \subseteq S_j$ $S_j \subseteq S_i$, $S_i \cup S_j = V$.

Let $J$ be a min size $T - join$, define a length function

$$l_e = 1 \; if \; e \notin J(add \; e \; t \; J), = -1 \; if \; e \in J(remove \; e \; from \; J)$$

If there is a negative length cycle then $J \oplus C$ is a T-Join with size $|J| + l(C) < |J|$. Take a min length walk(path) $p$ with the minimal number of edges. -the end edges have $l = -1$
-$v$ is adjacent to one $-1$ edge $f$, otherwise we get shorter walk.

**Claim**: any cycle $C$ using $v$ but not $f$ has length $> 0$. -if $C \cap p = v$ then $l(C) > 0$ Let $u$ be the last vertex of $p$ in $C$ (before we get back to $v$). -consider subpath $p[u \rightarrow v]$, $l(p[u \rightarrow v]) < 0$ or we get a path of shorter *length* and less edges.

So $l(C[u \rightarrow v]) > 0$, $l(C[u \leftarrow v]) > 0$ or we get a negative cycle, therefore $l(C) > 0$. This proves the claim.

contract $v \cup \Gamma(v)$ to get $G'$. (note v $\in T$).$T' := T \cup v_0 - (v \cup \Gamma(v) \; if | T \cap (V \cup \Gamma(u) | isodd, \; 0 \; otherwise$

Set $J' := J - f$ Claim: $J'$ is a min sized $T' - join$.

*Proof.* Suppose not. Then there is a circuit $C'$ in $G'$such that $|C' - J'| < |C' \cap J'|$ , $(J' \oplus C')$ is smaller then $J'$.

$C'$ must correspond to a circuit $C$ in $G$ that uses $v$, otherwise $J \oplus C$ gives smaller $T - join$.

If $C$ uses $f$ then

$$|C' - J'| - |C - J| - |C \cap J| \geq 0$$

(ii) $C$ does not use $f$. By claim $l(C) > 0$ so $l(C) \geq 2$ as $C$ is bipartite.

$$|C' - J'| = |C - J| - 2 \geq_{2 edges in C \cap (v \cup \Gamma(v))} |C \cap J| = |C' \cap J'|$$

this proves the claim.

By induction on $n + |T|$, $G'$ has $|J'| \uparrow T' - cuts, S_1, ..., S_{|J'|}$, $|J| = |T'| + 1$ But $\{v\}$ is a $T - cut$. It is not in $G'$ so is disjoint from $S_1, ..., S_{|J'|}$ $\square$

Consider the following polyhedron $P(G)$

$$\sum_{e \in \delta(S)} x_e \geq 1 \; \forall T - cuts \; S$$

$$x_e \geq 0$$

$min \sum_{e \in E} c_e x_e$ *over* $P(G)$.

**Theorem 21.** $P(G) = conv.hull(\chi_J : J \ is \ T-join) + \mathbb{R}^m$

*Proof.* $\chi_J$ is in $P(G)$ as $T-joins$ intersect all $T-joins$.
Suppose $P(G) \nsubseteq conv(T-join) + \mathbb{R}^m_+$
So $\exists w \geq 0 \ s.t. \ min \ w^t x \ over P(G)$ is smaller then the min weight$= \alpha \ T-Join$. We may assume (by scaling up) that $w_e$ is even $\forall e \in E$. Create new graph $G'$ with edge $e$. Therefore $G'$ is bipartite (*no odd cycles*). Min *#edges* size $T-join$ in $G' = \alpha$ (edge $e$ of weight $w_e$ replaced by $w_e$ edges). So there exist packing of $\alpha$ T-cuts Each $T-cut$ in $G'$ corresponds to a $T-cut$ in $G$. and each edge $e \in G$ is in at most $w_e$ of these $T-cuts$.

As $G'$ basically has many copies of $e$, so a $T-cut$ in $G$ may correspond to many $T-cuts$ in $G'$. Let $y_S$ be the number of times $S$(in G) appears in the packing of $G'$.

$$max \sum_{S:T-cut} y_S = \alpha$$

$$\sum_{S:e \in \delta(S)} y_S \leq w_e$$

$$y_S \geq 0$$

Choice of $y_S$ give optimal dual variables, i.e. min $w^T x \geq \alpha$

# 16 Totally Unimodular matrices

We have seen many examples where we model a combinatorial problem by an LP that has integral vertices. In these cases we could solve the problem in polytime e.g. by the ellipsoid method. Are there other examples of this? Here we will try to be more systematic.

A matrix $A$ is totally unimodular if every sub-determinant has value $0, 1, -1$.

**Theorem 22.** *If $A$ is T.U and $b$ is integral, then the polyhedron $P := \{x : Ax \leq b\}$ is integral.*

*Proof.* Let $F := \{x : A'x \leq b'\}$ be a face of $P$. Where $A'x \leq b'$ is a subsystem of $Ax \leq b$ and $A'$ is full rank(can assume). Since $A'$ is full rank we can re-order the columns so that $A' = [U, V]$ where $det(U) = \pm 1$. So $x = \binom{U^{-1}b'}{0}$ is integral. □

**Corollary 7.** *If $A$ is $TU$ and$b, c$ are integral then both primal and dual problems. $max\{c^T x : Ax \leq b, x \geq 0\} = min\{b^T y : A^T y \geq c, y \geq 0\}$ have integral optimal solutions.*

*Proof.* Follows from the facts that $\binom{A}{I}$ is $TU$, and transpose of $TU$ matrix is $TU$. □

Are there any interesting $TU$ matrices? Yes e.g. let $C$ be the vertex-arc incidence matrix of a directed graph $G$. Let $C$ be the vertex arc incidence matrix.

**Theorem 23.** *Any $0, \pm 1$ matrix with at most one $-1$ and $1$ in any column is $TU$.(e.g. $C$)*

*Proof.* Take a $k \times k$ submatrix $B$ of $C$. If $k = 1$ then $det(B) = 0, \pm 1$. If $k \geq 2$ then if some column of $B$ that has only one non-zero entry then expanding along that column gives the result. If not , then each column of $B$ has one 1 and one $-1$, so summing all rows gives a $0 - vector$, i.e. $det(B) = 0$. □

# 17 Applications

-Max Flow min cut theorem.

**Theorem 24.** *Given directed graph $G$ and vertices $s, t$ and some integer $k$ the $P :=$ collection of $k$ disjoint $s - t$ paths $(P_1, ..., P_k)$*
*$C =$ collection of $s - t$ cuts (not necessarily disjoint).*

$$min \sum_{i=1}^{k} |P_i| = max| \cup_{s \in C} \delta(S)| - \sum_{S \in C} (|\delta(S)| - k)$$

*Proof.* Want

$$min \sum_{a \in A} x_a$$

$$\sum_{a \in \delta^-(v)} x_a - \sum_{a \in \delta^+(v)} x_a = (-k \; if \; v = s, k \; if \; v = t, 0 \; otherwise)$$

$$0 \leq x_a \leq 1$$

$$min\ 1 - x =: \lambda$$
$$s.t.\ [C, -I]^T x \leq_{\geq}^{=} [-k, k, 0, 0, ..., -1, -1, ..., -1]^T$$

The dual is

$$max - \sum_{a \in A} y_a - kp_s + kp_t$$
$$[C^T, -I][p, y]^T \leq 1$$
$$\forall v, p_v\ unconstrained\ \forall a\ y_a \geq 0$$

the dual constraints are

$$-p_a + p_v - y_a \leq 1 \forall a \leq (u, v)$$

$$y_a \geq 0 p_s = 0$$

both have integral solutions as $C$ is $T.U.$ Now $\lambda \geq 0$ so $P_t \geq 0$, so $P_t = r \in \mathbb{Z}^+$ For each $j \in \{1, 2, ..., r\}$ let $S_j = \{v : p_r < j\}$ So we have $r$ $s-t$ cuts $(p_s = 0, p_t = r)$, $S_1, ..., S_r$.

$$\sum_{j=1}^{r} \delta^+(S_j) \leq \sum_{a=(u,v) \in A, p_u < p_v < r} (p_v - p_u)$$

$$\leq \sum_{a \in A, p_v > p_u} (1 + y_a) \leq |\cup_{j=1}^{r} \delta^+(S_j)| + \sum_{a \in A} y_a$$
$$= |\cup_{j=1}^{r} \delta^+(S_j)| + kp_t - \lambda$$

Hence

$$\lambda \leq |\cup_{j=1}^{r} \delta^+(S_j)| - (\sum_{j=1}^{r} (|\delta^+(S_j)| - k))$$
$$\leq max\ C$$

We know

$$min \geq\geq max|\cup \delta(s)| \geq |\cup \delta^+(s)| - (\sum_{s \in C} \delta^+(s) - k)$$

$\geq 0$ as we have $k$ disjoint $s-t$ paths.

□

We have a similar result if we reverse the roles of cuts and paths.

We would like to extend these methods. We will need the following result about $C$: Let $C^{-i} = C$ *minus the ith row.*

**Lemma 10.** *An* $(n-1) \times (n-1)$ *sub-matrix* $B^{-i}$ *of* $C^{-i}$ *is a basis of* $C^{-i}$ *iff the columns of* $B^{-i}$ *form a spanning tree on* $G$*.*

*Proof.* So we will show

$$|B^{-i}| = (\pm 1 \; if B^{-i} \; is \; spanning \; tree, 0 \; otherwise)$$

Assume edges of $B^{-i}$ dont give a $S.T$, assume $i = 1$. If its not a spanning tree, then there exists at least 2 components. $\exists component \; S \; where \; v_i \notin S$. Summing the rows in $S$ gives 0. So $|B^{-i}| = 0$.

So assume $B^{-i}$ gives a spanning tree. Renumber the vertices such that $w_1 \neq v_i$ is a degree 1 vertex with respect to $B^{-i}$. Let $w_t \neq v_i$ be degree 1 vertex in $B^{-i} - w_1$. $w_1 = v_4, w_2 = v_3, w_3 = v_2$ Let $e_1, e_2$ be the edges incident to $w_i$ when they are removed. So $e_i = (w_i, w_{i'})$ where $i' > i$ So $B^{-i}$ is lower triangular. Since $v_i$ has end point $e_i$ we have $|B^{-i}| = \pm 1$.

□

**Theorem 25.** $P := \{x : Ax \leq b, x \geq 0\}$ *is integral* $\forall b (integral)$ *iff* $A$ *is* $T.U$

*Proof.* Exersize. □

Now we look at a broader class of $TU$ matrices.

## 17.1 Network Matrices

Given $G := (V, A)$ and a tree $T := (V^0, A^0), V \subseteq V^0, \; but \; A^0 \nsubseteq A$ Then the network matrix $M$ is defined as :
(1) rows of $M$ indexed by arcs in $A^0$
(2) columns of $M$ indexed by arcs in $A$
(3) $M_{a^0, a=(u,v)} =$
$1 \; if \; (u \rightsquigarrow v) \; in \; T \; use \; a^0 \; forwards$
$-1 \; if \; (u \rightsquigarrow v) \; in \; T \; use \; a^0 \; backwards$
$0 \; does \; not \; use \; a^0$

Consider $C^{-i}$ and a basis $B^{-i}$ of $C^{-i}$
note $B^{-i}M = C^{-i}$, and $B \cdot M = C$ $[B^{-i} \cdot C^{-i}]$ and also $[B^{-i}, C^{-i}]$ is TU as

there are at most one 1 and one $-1$ in each col. since $|B^{-i}| = \pm 1$ we have that $B^{-i}$ is a basis of $[B^{-i} \cdot C^{-i}]$ and $[B^{-i}, C^{-i}]$. Now ** if $V$ is TU and $W$ is a basis of $V$ then $W^{-1}V$ is TU. So $[I(B^{-i}C^{-i}] = [IM]$ is TU. So $M$ is $TU$.

**Examples of Network matrices:**
(1) Node - arc adjacency matrix $C$. Take $G$ and add a vertex $s$ with directed arc to each vertex in $G$, these arcs correspond to the vertices in $G$ at their end points. $s$ and its edges are $T$, and $M(T) = C$.

(2) Interval matrix(consecutive ones matrix) $T = path$ $G$ corresponds to arcs which jump intervals.

$$a_1^0 \to a_2^0 \to a_3^0 ...$$

(3) Vertex-edge incidence matrix of bipartite graph. (hmwk)
(want to set up so that there are 2 ones in each column).

$$max\ c^T x$$
$$Ax \leq b$$
$$x \geq 0$$

=

$$min\ b^T y$$
$$A^T y \geq c$$
$$y \geq 0$$

So if $A$ is TU, then $\forall b$ both primal and dual are integral.
e.g. $c = 1 = b$ then we have

$$max \sum_{e \in E} x_e = min \sum_{v \in V} y_v$$

s.t.

$$sum_{e \in \delta(v)} x_e \leq 1\ \forall v \in V, y_u + y_v \geq 1$$
$$x_e \geq 0,\ y_v\ \geq 0$$

primal integral solution is a matching. Dual integral solution is a vertex cover. If $G$ is bipartite then $A$ is totally unimodular so primal and dual have integral optimal solutions.

**Theorem 26.** *In a bipartite graph max matching = min size of vertex cover.*

*Proof.* □

i.e. Vertex cover is polytime in bipartite graphs. However Vertex Cover is NP hard in general.

**Auction**

Sell plots of land on a waterfront. Assume bidder wants to buy one "piece" of land. Bidder $i$ bids $\$b_i$ for a "piece" of land.

want to

$$max \sum_{i=1}^{n} b_i x_i$$

s.t. $A = Interval\ matrix.$

$$Ax \leq 1$$

says, accept at most one bid containing plot $i$. $A$ is TU so we can maximize revenue. (this is extremely rare for auctions)

So network matrices are $TU$. The reverse is *almost* true. Every $TU$ matrix $A$ can be *composed* from Network matrices and two special $S \times S$ TU matrices(look this up and insert).

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ & 1 & & \\ -1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ & 1 & & \\ 0 & -1 & 1 & -1 \\ -1 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

# 18 Total Dual Integrality

Recall $TU$ matrices are those that have

$$P := \{x : Ax \leq b\}$$

is integral for all integral $b$.

Here we look at something a bit weaker. Consider LP duality:

$$max(cx : Ax \leq b) = min(yb : yA = c, y \geq 0)$$

We say that the system $Ax \leq b$ is TDI(fixed rational $A, b$) if the dual has integral optimum $y$ , $\forall$ integral $c$. This may be of interest as the dual may have combinatorial meaning, maybe this info helps us solve the primal as well, e.g. primal-dual algorithms. It also says a lot about the primal...

**Theorem 27.** *If $Ax \leq b$ is TDI, then for rational $A$, and integral $b$,*

$$P := \{x : Ax \leq b\}$$

*is integral.*

*Proof.* If $b$ is integral, min $b^T y$ is integral by TDI. So max $c^T x$ is integral $\forall c \in \mathbb{Z}$. Claim: A rational polytope $P := \{x : Ax \leq b\}$ is integral if and only if $\forall c \in \mathbb{Z}$, max $c^T x$ is integral. Necessity($\Rightarrow$) is clear.
Sufficiency($\Leftarrow$): Suppose $\forall$ integral $c$, max $c^T x$ is integral. Take vertex $v$ of $P$. Want to show that $v$ is integral. So there exists $c$ such that $v$ maximizes $c^T x$ over $P$ (unique optimum), can assume $c$ is integral by scaling if necessary. Can also assume

$$c^T v > c^T u + u_1 - v_1 \forall \ vertices \ u \neq v$$

since by uniqueness we have $c^T v > c^T u$, and then we can scale up. So $v$ is optimal for $c' := (c_1 + 1, c_2, c_3, ..., c_n)$ but $c'v = cv + v_1$ is integral, so $v_1$ is integral. We can repeat this for the other coordinates. □

For a combinatorial problem $A, b$ are integral. TDI property for $Ax \leq b \Rightarrow P = (x : Ax \leq b)$ is integral. (in fact if $b$ is not integral the TDI tells us nothing).(see assignmet)

A $TUM$ matrix is TDI as transpose is $TUM$

## 18.1 Orientations of Graphs

**Theorem 28.** NASH-WILLIAMS
*A $2k-$ edge $-$ connected indirected graph can be oriented to be $k-$ arc connected.*

*Proof.* Take $G$ and pick an arbitrary orientation. call this $D$. Primal:

$$max \sum_{a \in A} c_a x_a$$

such that

$$\sum_{a \in \delta^-(S)} x_a - \sum_{a \in \delta^+(S)} x_a \leq |\delta^-(S)| - k \forall S \subset V$$

$$0 \leq x_a \leq 1$$

(*)

$$|\delta^-(S)| + \sum_{a \in \delta^+(S)} x_a - \sum_{a \in \delta^-(S)} x_a \geq k$$

$x_a = 1$–switch orientation, $x_a = 0$– keep orientation.

If (*) is true for integral solution $\forall S$ we are done. We want to show that the primal is integral. Putting $x = \frac{1}{2}$ gives feasible solution is

$$\frac{1}{2}(|\delta^+(S)| + |\delta^-(S)|) \geq \frac{1}{2} 2k = k$$

So the LP is non-empty.

Show system is TDI:

$$[A, I]x \leq [\delta^-(S_i) - k ... 1]$$

where $a_{ij}$ is 1 if coming out, 0 not in $-1$ if leaving the cut $S_j$.

$$min \sum_S z_S(|\delta^-(S)| - k) + \sum_{a \in A} y_a$$

such that

$$[A, I][z, y] \geq c$$

$$z, y \geq 0$$

$\forall a \sum_{s: a \in \delta^-(S)} z_S - \sum_{S: a \in \delta^+(S)} z_S + y_a \geq c_a.$

Look at dual variables $z_S$. with $z_S > 0$. We want to show that these cuts are a cross - free family: $S_i \cap S_j = \emptyset$ or $S_i \subseteq S_j$ or $S_j \subseteq S_i$ or $S_i \cup S_j = V$
Suppose $S, T$ have $z_S, z_T > 0$ and they cross. Set $z_S = z_S - \epsilon$
$z_T = z_T - \epsilon$
$z_{S \cap T} = Z_{S \cap T} + \epsilon$
$z_{S \cup T} = z_{S \cup T} + \epsilon$

Claim - New $z$ is feasible. New $z$ gives better objective value.
(*)

$$\sum_{s: a \in \delta^-(S)} z_S - \sum_{S: a \in \delta^+(S)} z_S + y_a \geq c_a$$

case analysis: each case cancels out. So star holds. Observe

$$|\delta^-(S)| + |\delta^-(T)| \geq |\delta^-(S \cap T)| + |\delta^-(S \cup T)|$$

(seen before)

So objective falls when we change $z$ values for these 4 cuts. So repeatedly un-crossing gives cross-free optimal solution. How does this help? So this means the primal (complementary slackness say the i constraints for $z_S > 0$ must be tight in primal). The primal only needs constraints such that $z_{S_i} > 0$ where $S_i$ are cross-free family.

Claim: $A$ is a network matrix. Take the cross free family, and construct a tree with the outer face as the root and add a directed edge from outer face to inner faces. The number of arcs in $T$ equals the number of cuts. $A$ is $TUM$. So the primal is integral.

remark: the original matrix is not $TUM$, since we removed lots of rows.
excersize : Show how to separate these constraints (in primal). □

# 19 Directed Cuts and Dijoins

A directed cut $\delta^+(S)$ is directed if $\delta^-(S) = \emptyset$. A set of arcs $T$ is a dijoin (or directed cut cover/traversal) if $T$ intersects every directed cut(dicut).

If $T$ is a dijoin then (1) $G \cup T'$ is strongly connected.
(2) $G/T$ is strongly connected.

So dijoins are fundamental objects in network design.

**Theorem 29.** *min size of dijoin = max size of packing of disjoint directed cuts.*

*Proof.* Clearly $\tau(G) \geq p(G)$. Take a minimal counter example $G$. (obviously $p(G) \geq 1$) So $\tau(G) = p(G) + 1$
We use the following transformation. Replace each arc in $B \subseteq A$ by a path of 2 arcs. Observe that $p(G_A) = 2p(G)$ . So there exists a maximal $B \subseteq A$ such that $p(G_B) = p(G)$ as $p(G_\emptyset) = p(G)$. So take any $a \in A - B$ then $p(G_{B\cup\{a\}}) = p(G) + 1$ by the maximality of $B$.

Moreover, since $G$ is a minimal counter example:

$$p(G/a) = \tau(G/a)$$

$$\geq \tau(G) - 1 = p(G)$$

We take maximum packings of dicuts in both $G/a$ and $G_{B\cup a}$ (note these are all cuts in $G_B$). Call this set $W$. $|W| = 2p(G) + 1$ Each arc is in at most 2 cuts. We show the "shores" of the cuts ($S$ is shore of $\delta^+(S)$, form a cross free family. If not, replace $S, T$ by $S \cap T, S \cup T$. which are *dicuts*. Also ($\chi_S$ is incidence vector of $\delta^+(S)$) $\chi_S + \chi_T \geq \chi_{S\in S'\cap T} + \chi_{S\cup T}$ . This process terminates

$$\sum_S |S| \cdot |V(G_B) - S|$$

calls each time. Now then $S' - (family\ of\ cuts)$ contains two types of cut.
(1)-$S_1$ those cuts that appear exactly once.
(2)-$S_2$ those cuts that appear exactly twice.

Note: No arc appearing in a cut in $S_1$ appears in a cut in $S_2$(otherwise it appears 3 times). Now consider the cuts in $S_1$, let $S \in S_1$ and set

$$\alpha(S) := \{T : T \in S', T \subseteq S \text{ or } T \cap S = \emptyset\}$$

Let

$$S_1^{odd} = \{S \in S_1 : |\alpha(S) \text{ is odd}\}$$
$$S_1^{even} = \{S \in S_1 : |\alpha(S) \text{ is even}\}$$

Claim: sets in $S_1^{odd}$(resp $S_1^{even}$) are arc disjoint.

*Proof.* Take $S, T \in S_1^{odd}$ with $a \in \delta^+(S) \cap \delta^+(T)$ Since $S'$ is cross-free we may assume $S \subseteq T$.
(i) Suppose $|\alpha(T)| \leq |\alpha(S)|$ . But $T \in \alpha(T) - \alpha(S)$ so there is -$R \in \alpha(S) - \alpha(T)$. Now $R \cap S = \emptyset$ otherwise $R \subset S \subset T$.
-$R \cap T\emptyset$ *or* $R \in \alpha(T)$
-$R \nsubseteq T$. So $R \cup T = V(G_B)$. Otherwise not-cross free family. Then $a \in \delta^-(R)$ so $R$ is not a directed cut. $\Rightarrow\Leftarrow$.

(ii) $|\alpha(T)| \geq |\alpha(S)| + 2$ So there is an $R \neq T$ such that $R \in \alpha(T) - \alpha(S)$ So $S \subseteq R \subseteq T$. But then $a$ is in 3 cuts $\Rightarrow\Leftarrow$
(iii) $|\alpha(T)| = |\alpha(S)| + 1$ $\Rightarrow\Leftarrow --$ *must have same parity.*

So cuts in $S_1^{odd}$ are disjoint. WLOG $|S_1^{odd}| > |S_1^{even}|$ So $S_2 \cup S_1^{odd}$ gives disjoint set of $p(G) + 1$ cuts in $G_B$. So $p(G_B) \geq p(G) + 1$, $\Rightarrow\Leftarrow$, as $p(G_B) - p(G) \Rightarrow \tau(G) \leq p(G)$ □

□

**Corollary 8.** *Given $G := (V, A)$ and $w : A \to \mathbb{Z}_+$ min weight dijoin = max number of dicuts , such that $a$ is in at most $w(a)$ cuts $\forall a \in A$*

*Proof.* Replace $a$ by path of length $w(a)$. If $w(a) \geq 1$. If $w(a) = 0$ contract $a$. Apply previous theorem. □

This means the following system is totally dual integral.
(P)

$$min \sum_{a \in A} w_a x_a$$

such that

$$\sum_{a \in \delta^+(S)} x_a \geq 1 \forall S \; is \; directed \; cut$$

$$x_a \geq 0$$

(D)

$$max \sum_{S \; dicut} y_S$$

such that

$$\sum_{S \; dicut = a \in \delta^+(S)} y_S \leq w_a \; \forall a \in A$$

$$y_S \geq 0$$

# 20 Packing Arborescences

**Theorem 30.** *Max number of arc-disjoint r-arborescences = min size of an $r-cut$*

We will prove something stronger. Given a set $R \subseteq V$, a branching with root set $R$ is a set of disjoint arborescences that span $V$. (eg if $R = \{r\}$ the branching = r-arb.) Let $R_1, R_2, R_3, ..., R_k \subseteq V$. We refer to the $i$ as colors. The $R_i$ may intersect so a vertex may have many colors. Let $\Gamma(U) := \#of\ colors\ missing\ U \subseteq V$.

**Lemma 11.** *There exists arc disjoint branchings rooted at $R_1, ..., R_k$ if and only if*

$$|\delta^-(U)| \geq \Gamma(U) \forall U \subseteq V$$

*Proof.* Clearly $|\delta^-(U)| \geq \Gamma(U)$ or there does not exist disjoint branchings. For the other direction:
We use induction on $\sum_{i=1}^{k} |V - R_i|$ (backward induction of sizes of $R_1, ..., R_k$) , eg $R_i = V \forall i$ we are done. So assume $R_1 \subset V$ Let $W$ be a minimal set such that

$$(1)\ W \cap R_1 \neq \emptyset\ i.e. color\ 1\ hits\ some\ vertex\ in\ W$$

$$(2) W - R_1 \neq \emptyset\ i.e.\ some\ vertex\ in\ W\ does\ not\ have\ color\ 1$$

$$(3) |\delta^-(W)| = \Gamma(W)\ i.e. \#in\ comming\ arcs = \#colors\ missing$$

$V$ satisfies properties so $W$ exists.
**Claim**: there exists an arc $(u, v)$ from $W \cap R_1 \to W - R_1$

*Proof.*

$$|\delta^-(W - R_1)| \geq \Gamma(W - R_1)$$

by assumption.

$$\geq \Gamma(W) + 1$$

(color 1 hits $W$ but misses $W - R_1$) by (1).

$$> \Gamma(W)$$

by (3):

$$= |\delta^-(W)|$$

Therefore $(u, v)$ exists. □

$G' := G - (u, v)$
Consider $A - (u, v)$,
$R'_1 := R_1 \cup \{v\}, R'_i := R_i \forall i \neq 2$
Suppose there exist disjoint branchings from $R'_1, ..., R'_k$ *in* $G'$ . For $i > 1$ we have the same branching , for $i = 1$ take the branching and add $(u, v)$.

Show $G', R'_i$ satisfy initial conditions and apply induction. If not

$$\exists U \ st \ |\delta^-(U)| < \Gamma(U) \ with \ respect \ to \ G', R'_i$$

But condition held in $G, R_i$, $|\delta^-(U)| \geq \Gamma(U)$. Therefore must fall by 1, i.e. $(u, v) \in \delta^-(U)$, $\Gamma(U) = \Gamma'(U) \ if \ it \ falls \ we \ still \ have \geq$. i.e. $R_1 \cap V \neq \emptyset$ or it falls as $v \in R'_1$ we have that (*) $|\delta^-(U)| = \Gamma(U)$

Back in $G$...

$$\delta^-(U \cap W)| \leq |\delta^-(U)| + |\delta^-(W)| - |\delta^-(U \cup W)|$$

by (*) and (3) we have

$$\delta^-(U \cap W)| \leq \Gamma(U) + \Gamma(W) - |\delta^-(U \cup W)|$$

$$= \#\ of \ colors \ missing$$

U *or* W

$$(.) \leq \#colors \ missing \ U \cap W = \Gamma(U \cap W)$$

. i.e.

$$\delta^-(U \cap W)| \leq \Gamma(U \cap W)$$

So $|\delta^-(U \cap W)| = \Gamma(U \cap W)$ by initial assumption. Therefore all equalities above. We know that 1 hits $W$ by (1). We know that 1 hits $U$ .

**Claim**: 1 hits $U \cap W$: Suppose not, then we dont have equality in (.) So $(U \cap W) \cap R_1 \neq \emptyset$.
$(1')$

$(2')$ $(U \cap W) - R_1 \neq \emptyset$ so $v \in U \cap W$ but $v \notin R_1$, $(v \in R'_1)$. Want to show that $U \cap W \subset W$ and we can by simply noting that $u \in W - U$. $\square$

**Theorem 31.** *Max number of arc-disjoint r-arborescences = min size of an* $r - cut$

*Proof.* The theorem follows from this lemma by setting $R_1 = R_2 = ... = R_k = \{r\}$ and applying the result. $\Gamma(u) = 0 \; if \; r \in U$ So $\Gamma(U) > 0$ only for $r - cuts$ □

Consider the LP:

$$min \sum_a c_a x_a$$

$$s.t.$$

$$\sum_{a \in \delta^-} x_a \geq k \; \forall r - cuts \; S$$

$$0 \leq x_a \leq 1$$

This is $TDI$ by results on $r - arbs$. So we have integral $LP$ . So by packing result these contain $k$ disjoint arboresonces. Therefore min $LP$ we get min cost packing of $k - arbs$ .

Given $G = (V, E)$ with $c_e \geq 0$ Find min cost $k - edge - connected$ subgraph. This is $NP$ hard. $k = 2, c_a = 1$(Hamilton cycle).

**Approximation Algorithm:**-Polytime, -gives feasible solution, $cost \leq \alpha \cdot Optimal$. There exists $2 - approx - algorithm$

*Proof.* Bidirect each edge $(u, v)_{c_e} \Rightarrow (u, v)_{c_e}^{\leftarrow} + (u, v)_{c_e}^{\rightarrow}$
pick any root $r$. Now $\delta^-(u) \geq k \forall r - cuts$ because $G$ is $k - connected$.

So in polytime can find min cost $k - packing$ of $r - arbs : T_1, T_2, ..., T_k$

$$\sum_{i=1}^{k} c(T_i)$$

Pick undirected $\cup T_i$ in $G$. $cost(\cup T_i) \leq \sum c(T_i)$
$\cup T_i$ is $k - edge$ connected. i.e. min cut is $k$. $\geq 1$ arc in $\delta^-(S) \forall T_i$ therefore $k$ $edges$.

Take $H \subseteq G$ $min$ $k$ $edge$ $connected$ $subgraph$ Thus if we take both copies this gives a feasible solution to LP so $\cup T_i$ in $G$. $cost(\cup T_i) \leq \sum c(T_i) \leq 2cost(H) = 2optimal$

# 21 Information Theory

We have a set of symbols $V$, that can be used to create messages (words) for transmission. Some pairs of symbols may be confused during transmission. Represent this by "'confusion Graph"'. Two words can be confused if every symbol in them can be confused(in order). e.g. if words have size 1 then max # words $= \alpha(G) :=$ Largest stable set.

e.g. suppose $\alpha(C_5) = 2$. If words have size 2 $x_1x_2$ and $y_1y_2$ can be confused iff
(i) $x_1 = y_1$ *and* $(x_2, y_2) \in E$
(ii)$(x, y) \in E, x_2 = y_2$
(iii) $(x_1, y_1), (x_2, y_2) \in E$

This can be viewed as the product of graphs.

$$G_1 \times G_2$$

$$V := \{v_1v_2 : v_1 \in V_1, v_2 \in V_2\}$$

$$E := u_1u_2 \; adjacent \; to \; v_1v_2 \; iff \; u_i = v_i \; or \; (u_i, v_i \in E_i \forall i$$

It follows easily that max # 2 symbol words is $\alpha(G \times G) = \alpha(G^2)$ in general for $n - symbol$ words max # is $\alpha(G^n)$. The information rate of words of length $n$ is

$$\frac{bg_z \; \frac{\alpha(G^n)}{}}{n = log(\alpha(G^n)^{\frac{1}{2^n}})}$$

Shannon asked what is the maximum info rate of $G$. i.e. max $\theta(G) = max_n \alpha(G^n)^{\frac{1}{2^n}})$

Observe : Stable set $\{x_1, x_2, ..., x_k\}$ in $G$. Consider $G^n$ vertices:$y_1, y_2, ..., y_n$ $y_i \in V$
if $y_i = x_r, \; r = 1, ..., k$ certainly have a stable set of size $k^n$. i.e. $\alpha(G^n)^{\frac{1}{2^n}}) \geq \alpha(G)$.

We are interested in

$$\theta(G) = sup_{n \geq 1} \alpha(G^n)^{\frac{1}{2^n}})$$

**Lemma 12.** $\theta(C_5) \geq \sqrt{}(5)$.

*Proof.* arrange the vertices in a $5 \times 5$ grid. Each row is a $5-cycle$, and each column is a $5-cycle$. There are other edges as well $(2,2) \leftrightarrow (3,2),(1,2),(2,1),(2,3)$ *plus diagonals* : $(1,1),(3,1),(1,3),(3,3)$
We can have a stable set of size 5, but not of size 6 since then we would have 2 in the same row . □

Shannon calculated $\theta(G)$ for all $5-node$ graphs except $C_5$.

## 21.1 Upperbounds

We can upperbound $\alpha(G^n)^{\frac{1}{2^n}}$ by L.P duality. Towards this goal, let $W$ be the set of cliques in $G$. For any probability distribution $x$, $\sum x_i = 1, x_i \geq 0$ on vertices let

$$\lambda(x) = max_{Q \in W} \sum_{v \in Q} x_v$$

$$\lambda(G) = min_x \lambda(x)$$

Take a stable set $S$. Set $x_v = \frac{1}{|S|}\ if\ v \in S,\ 0\ if\ v \notin S$

as a clique intersects $S$ in at most one vertex. So $\lambda(G) \leq \frac{1}{\alpha(G)}$ let $S =$ max stable set. in fact $\lambda(G) \leq \frac{1}{\alpha(G^n)^{\frac{1}{2^n}}}$ we will show

$$\lambda(G \times H) = \lambda(G) \cdot \lambda(H)$$

$$\lambda(G^n) = (\lambda(G))^n \leq \frac{1}{\alpha(G^n)}$$

**Claim:** $\lambda(G \times H) = \lambda(G)\lambda(H)$
want

$$min\ max_{Q \in W} \sum_{v \in Q} x_v$$

$$\sum_{v \in V} x_v = 1$$

$$x_v \geq 0\ \forall v$$

we can write this as

$$min_x \beta$$

$$\beta - \sum_{v \in Q} x_v \geq 0\ \forall Q \in Q$$

$$\sum_{v \in V} x_v \geq 1 * *(optimal\ solution\ must\ have\ \sum x_v = 1\ anyway)$$

$$\beta, x_v \geq 0$$

(Dual)

$$max\Gamma$$

such that

$$\Gamma - \sum_{Q \in W : v \in Q} y_Q \leq 0\ \forall v$$

$$\sum_{Q : Q \in W} y_Q \leq 1$$

$$\Gamma, y_Q \geq 0\ \forall Q \in W$$

get a probability distribution on $Q$.

=

$$max_y\ min_{v \in V} \sum_{Q : v \in Q} y_Q$$

$$\sum y_Q = 1$$

$$y_Q \geq 0 \forall Q$$

So

$$min_x max_Q \sum_{v \in Q} x_v = max_y min_v \sum_{Q; v \in Q} y_Q$$

Consider $G \times H$ and let $x, x'$ achieve mins on $G, H$ respectively. For each vertex in $G \times H$ let

$$Z_{uv} = x_u \cdot x'_v$$

$$\sum_{u,v} z_{uv} = \sum_{u \in G} \sum_{v \in H} x_u x'_v = \sum_{u \in G} x_u \sum_{v \in H} x_v = 1 * 1 = 1$$

**Claim:** $\lambda(G \times H) = \lambda(G)\lambda(H)$

i.e. $z$ is a probability distribution. The maximal cliques in $G \times H$ are of the form $Q_G \times Q_H$ where $Q_G, Q_H$ are cliques in $G, H$ respectively. Hence

$$\lambda(G \times H) \leq \lambda(z) = max_{Q_G \times Q_H} \sum_{u,v \in Q_G \times Q_H} z_{uv}$$

$$= max_{Q_G \times Q_H} (\sum_{u \in Q_G})(\sum_{v \in Q_H} x'v)$$

$$= max_{Q_G} \sum_{u \in Q_G} x_u \cdot max_{Q_H} \sum_{v \in Q_H} x'_v = \lambda_G(x)\lambda_H(x')$$
$$= \lambda(G)\lambda(H)$$

Now

$$\lambda(G \times H) \geq \lambda(G)\lambda(H)$$

using same idea applied to the dual. □

e.g. consider cycles $C_k, k \geq 4$. Setting $x = (\frac{1}{k}, \frac{1}{k}, ..., \frac{1}{k})$ So $\lambda(C_k) \leq \frac{2}{k}$ as edge are max cliques.

Consider dual:
let $y_Q = \frac{1}{k}$ for each clique(edge) . So $\lambda(C_k) \geq \frac{2}{k}$(each vertex has 2 cliques). therefore $\lambda(C_k) = \frac{2}{k}$
$\theta(G) \leq \frac{1}{\lambda(G)} = \frac{k}{2}$. But $\theta(G) \geq \alpha(G) = k/2 \; if \; k \; is \; even, (k-1)/2 \; for \; k \; odd$ i.e.

$$\theta(G) = k/2 \; even \; cycles$$
$$\frac{k}{2} \geq \theta(G) \geq \frac{k-1}{2} \; odd \; cycles$$
$$\frac{5}{2} \geq \theta(C_5) \geq \sqrt{(}5)$$