Lecture 20

Plan: 1) ~~Finish LCIS algo.~~
2) Matroid intersection polytope
3) Start min-cost arborescence

after today, ~4 more lectures! ~1 matroids, ~3 ellipsoid

Matroid intersection polytope

- Let $M_1 = (E, \mathcal{I}_1)$ & $M_2 = (E, \mathcal{I}_2)$ be two matroids, rank functions r_1, r_2.

- analogously to the matroid polytope, let

$$X = \{ 1_S \in \{0,1\}^E : S \in \mathcal{I}_1 \cap \mathcal{I}_2 \}$$

$$\{0,1\}^E \subseteq \mathbb{R}^E$$

i.e. $X \subseteq \mathbb{R}^E$ is set of indicators of common independent sets.

- Define the matroid intersection polytope $P_{M_1, M_2} := \mathrm{conv}(X)$

(can use to optimize linear functions over X).

- Main result: P_{M_1,M_2} is the intersection $P_{M_1} \cap P_{M_2}$ of the matroid polytopes P_{M_1}, P_{M_2} of M_1, M_2.

$\Rightarrow$ vertices$(P_{M_1}) \cap$ vertices(P_{M_2}) = vertices$(P_{M_1} \cap P_{M_2})$!

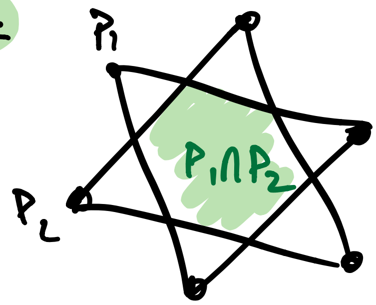
- This is surprising! In general, for polytopes P_1, P_2

verts$(P_1) \cap$ verts$(P_2) \neq$ verts$(P_1 \cap P_2)$

vertices(P_{M_i}) = indicators of $\mathcal{I}_i$ $\Rightarrow$ intersection is common indep sets. = vertices of P_{M_1,M_2}

if $P_{M_1,M_2} = P_{M_1} \cap P_{M_2}$, then same set of vertices.

e.g.



P_1, P_2 share no vertices but $P_1 \cap P_2 \neq \emptyset$ (& hence has vertices).

- In terms of inequalities?
- Recall matroid polytope: for r rank function of M,

$$P_M = \{x \in \mathbb{R}^E : x(S) \leq r(S) \ \forall S \subseteq E$$
$$x_e \geq 0 \ \forall e \in E\}.$$

$x(S) := \sum_{e \in S} x_e.$

- $P_{M_1} \cap P_{M_2}$ has both sets of constraints, so

1981: can efficiently decide membership in P_{M_1, M_2}.

Theorem: Let $P = P_{M_1} \cap P_{M_2}$, i.e.

$$P = \left\{ x \in \mathbb{R}^E : \begin{array}{l} x(S) \le r_1(S) \quad \forall S \subseteq E \\ x(S) \le r_2(S) \quad \forall S \subseteq E \\ x_e \ge 0 \quad \forall e \in E \end{array} \right\}$$

Then

$$\boxed{P_{M_1 M_2} = P}$$ i.e. $P_{M_1 M_2} = P_{M_1} \cap P_{M_2}$.

Proof: Plan: similar to lecture 17, vertex proof for matroid polytope.

- Like second proof for matroid polytope, use vertex integrality

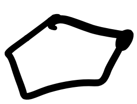
- Integrality of P suffices by the usual logic:

▷ clearly $\mathrm{conv}(X) \subseteq P$, b/c $X \subseteq P_{M_1} \cap P_{M_2} = P$

▷ On the other hand, if P integral then $P \subseteq \mathrm{conv}(X)$ because integral points in P_{M_1}, P_{M_2} are indicators of indep sets in M_1, M_2. $\Rightarrow$ integral points in $P_{M_1} \cap P_{M_2} = P$ are <u>common</u> indep sets.

- Again, if $P = \{x : Ax \le b, x \geq 0\}$, matrix A is <u>not</u> totally unimodular.

- But there are matrices ~~submatrices~~ describing vertices <u>will</u> be T.U. (enough).


x^*

$x^* = (.1, 1, \sqrt{2})$?

$x^* \in \mathbb{R}^E$

<u>Let x^* be an extreme point of P.</u>
↳ i.e. vertex

- We know x^* characterized by which inequalities are tight for it.

- For $i \in \{1,2\}$, let

$$T_i = \{S \subseteq E : x^*(S) = r_i(S)\} \qquad x^*(S) := \sum_{e \in S} x^*_e = x^* \cdot \mathbb{1}_S.$$

i.e. T_i sets of tight rank constraints in M_i.

- Let $J = \{e : x^*_e = 0\}$.

- Then x^* is unique solution to

$$F_1 \left\{ \begin{array}{ll} x(S) = r_1(S) & \forall S \in T_1 \\ F_2 \left\{ \begin{array}{l} x(S) = r_2(S) \\ x_e = 0 \end{array} \right. & \begin{array}{l} \forall S \in T_2 \\ \forall e \in J. \end{array} \end{array} \right.$$

- That is, $\{x^*\}$ is the intersection of two faces F_1, F_2 in P_{M_1}, P_{M_2}.

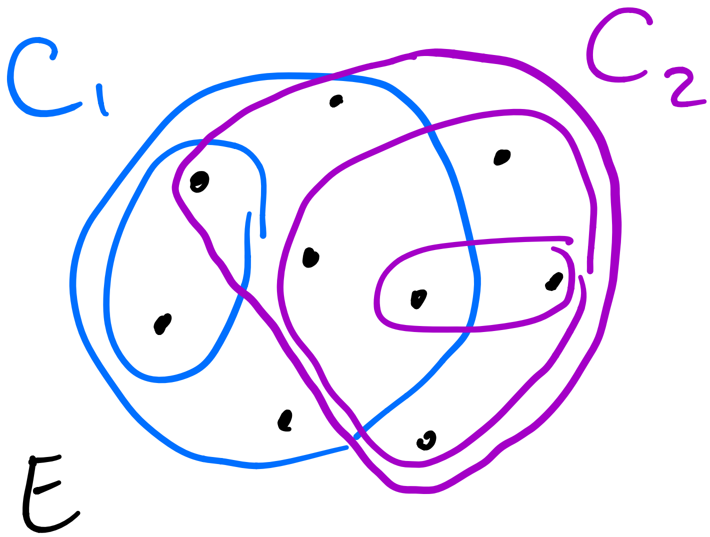
$$F_i = \{x \in P_{M_i} : x(S) = r_i(S) \ \forall S \in T_i$$
$$x_e = 0 \quad \forall e \in J\}.$$

- Recall from lec 17: T_i can be replaced by a chain C_i without changing F_i.

$\exists C_1, C_2$ chains s.t.

$$F_i = \{x \in P_{M_i} : x(S) = r_i(S) \ \forall S \in C_i$$
$$x_e = 0 \quad \forall e \in J\}.$$

e.g.



- Thus, assume x^* is solution to

$$x(S) = r_1(S) \quad \forall S \in C_1$$

$$x(S) = r_2(S) \quad \forall S \in C_2$$

$$x_e = 0 \quad \forall e \in J.$$

- This is $Ax = b$ for $b \in \mathbb{R}$

$\boxed{\text{Claim: } A \text{ T.U.}}$ $\Rightarrow x^*$ integral.

- Why? Rows of A are 1s of S in chain C_1 or C_2.

e.g.

$$A = \begin{array}{c} \left.\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}\right. \end{array} \begin{array}{l} \left.\vphantom{\begin{matrix}1\\1\\1\end{matrix}}\right\} C_1 \\ \left.\vphantom{\begin{matrix}1\\1\\1\end{matrix}}\right\} C_2 \end{array}$$

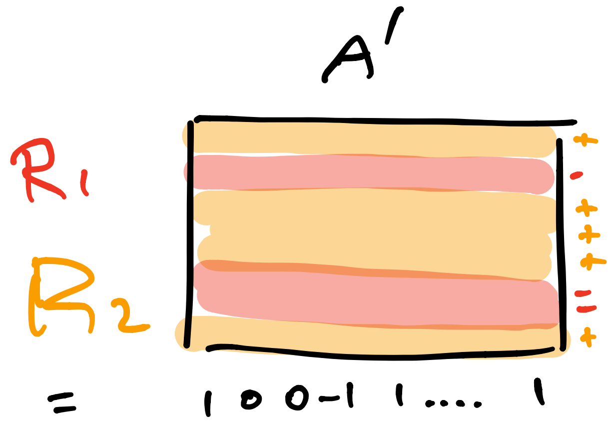
can permute rows of A so it looks like this. doesn't change T.U.

- We use discrepancy to prove.
  Theorem 3.14 in polyhedral notes

- Recall: A T.U. $\Leftrightarrow$ $\forall$ submatrices A' of A, $\exists$ partition R_1, R_2 of rows of A'
  $\uparrow$ a_i

$\sum_{i\in R_1} a_i - \sum_{i\in R_2} a_i$ has $\{-1, 0, +1\}$.

e.g.

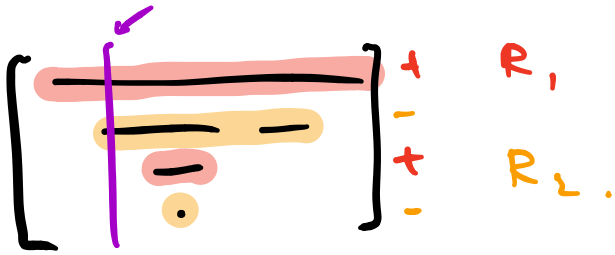


- Consider submatrix A' of A
  corresponds to subchains C_1', C_2'
  (same form as A)

- Assign R_1, R_2 as follows:
  - Assign largest element of C_1' to R_1, then alternately assign remaining elts of C_1' to R_2, R_1

e.g.

C_1'

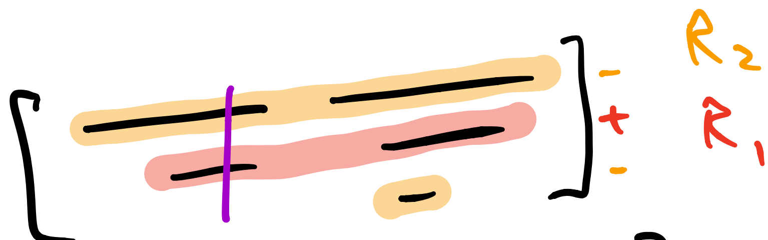


sum has entries in $\{0,1\}$.

▷ For C_2', assign oppositely.

e.g.

C_2'



sum has entries in $\{0,-1\}$.

- Overall, sum has entries in $\{-1,0,1\}$.
- completes the proof. $\square$.

Matroid intersection optimization

- Given a cost function $c : E \to \mathbb{R}$ can we efficiently compute

$$\max_{S \in \mathcal{I}_1 \cap \mathcal{I}_2} c(S) := \sum_{e \in S} c(e) = c \cdot \mathbb{1}_S .$$

equiv: optimize $c^T x$ over $x \in P_{M_1, M_2}$.

- For just one matroid: greedy alg works.
- For $c = \mathbb{1}$: just L.C.I.S.

  $(1, \ldots, 1)$ $|E|$ times
- For perfect matching: Hungarian algo.

  e.g. min-cost p.m.
- can also compute min cost L.C.I.S.

  Exercise: equiv. to max cost indep for $c' = K - c$ for K large.

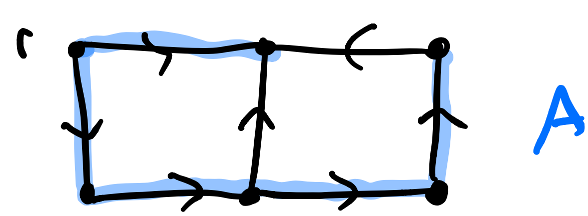
- In general, YES, can efficiently compute.
  - ellipsoid
  - complicated primal-dual algos.
    → strongly poly. time
    # steps indep of c
    if arithmetic is unit cost.

- ~~Today~~! simpler primal dual alg. for

Min-cost arborescence

- **Recall:** given directed graph D & vertex r, arborescence A is a spanning tree in D directed away from r.

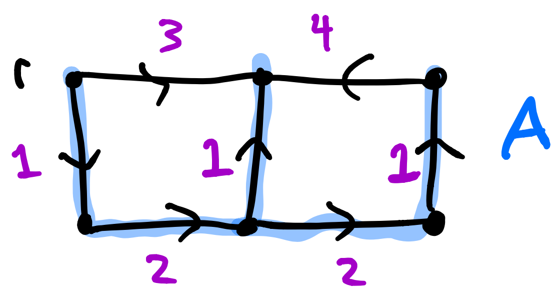
e.g.



- min-cost arborescence:

$$\min_{A \text{ arborescence}} \sum_{e\in A} c(e) = c(A)$$

e.g.



c

- e.g. edges = roads to be fixed
  r = distribution center
  cost = expense of fixing road.

- First, **I.P. formulation:**

assume c nonnegative.

think $x = 1_A$ for A arborescence
$\sum c_e x_e = c(A)$

$$OPT = \min_{x\in\mathbb{R}^E} \sum_{e\in E} c_e x_e$$

subject to $$\sum_{e\in\delta^-(S)} x_e \geq 1 \quad \forall S\subseteq V-r$$

$x = 1_A$, where A has no cuts.

$$\sum_{e\in\delta^-(v)} x_e = 1 \quad \forall v\in V-r$$

indegrees = 1 except r.

$$x_e \in \{0,1\}$$

x is an indicator.

- **Check**: only solutions are 1_A where A is an arborescence.
  in particular, all arbos satisfy constraints.

- **Miraculously**, we'll show even w/out integrality constraint & indegree constraint, there's still an optimal solution that's an arborescence.

- I.e. the following L.P. has optimizer 1_A s.t. A is an arborescence.

$$LP = \min_{x \in \mathbb{R}^E} \sum_{e \in E} c(e) x_e$$

$$\text{subject to } \sum_{e \in \delta(S)} x_e \geq 1 \quad \forall S \subseteq V - r.$$

(primal)

$$x_e \geq 0 \quad \forall e \in E$$

(note $LP \leq OPT$ b/c LP has fewer constraints.)

(typo in pre-lecture!)

- Dual LP is "symmetric version"

$$LP = \max \sum_{S \subseteq V-r} y_S$$

subject to

(dual) $$\sum_{S: e \in \delta^-(S)} y_S \leq c(e) \quad \forall e \in E$$

$$y_S \geq 0 \quad \forall S \subseteq V-r.$$

- **Algorithm sketch**: construct
  - ▷ arb. A,
  - ▷ dual. feas. y

satisfying complementary slackness

Then $c(A) = LP$, but $LP \leq OPT$

$c(A) \leq OPT \Rightarrow \boxed{c(A) = OPT.}$

- Complementary slackness for $x = 1_A$, y says:

  a.) $y_S > 0 \Rightarrow |A \cap \delta^-(S)| = 1$

  b.) $e \in A \Rightarrow \sum_{S : e \in \delta^-(S)} y_S = c(e)$.

- Two phases of algorithm:

  1) Construct
     - dual feas y
     - set F of edges s.t. every vertex of V is reachable from r in F,

  F might not be an arborescence.

  & y, $A = F$ satisfy (b).

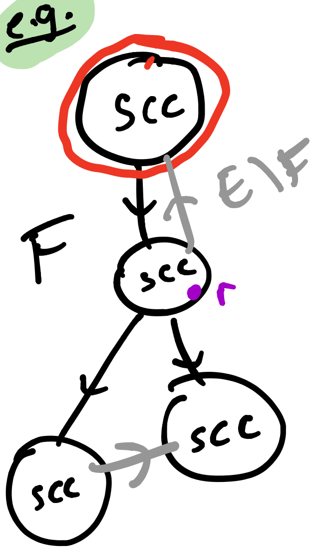
2) Remove unnecessary edges from F, get arborescence which satisfies both (a) & (b).

Phase 1

Initialize $F = \emptyset$, $y = 0$
counter $k = 1$

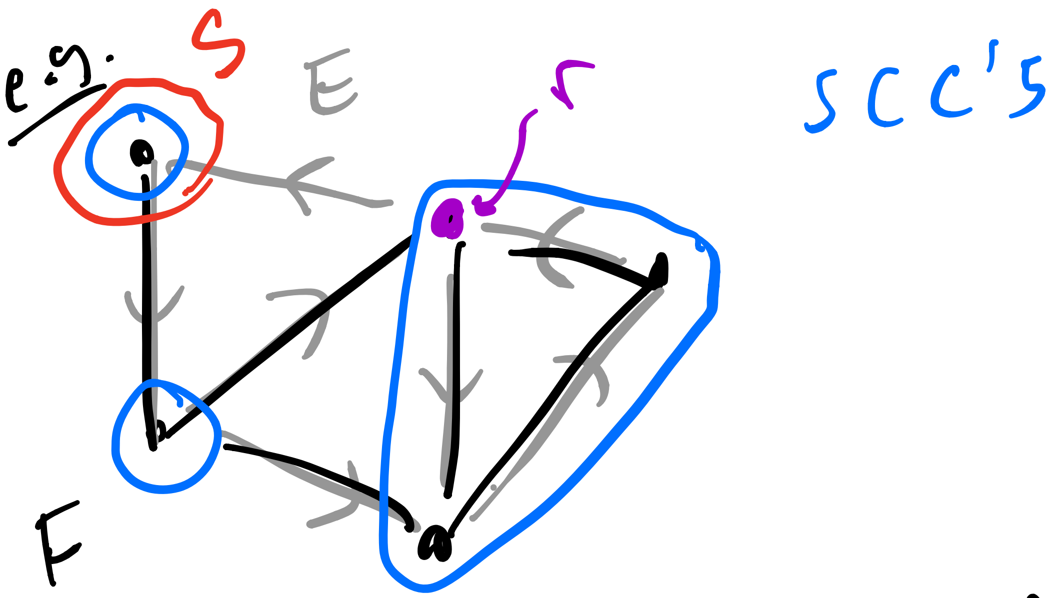
- **While** not everything reachable from r in F
  - select $S \subseteq V - r$
    - i) F strongly connected in S
      (every vertex can reach every other using only edges contained entirely in S)
    - ii) $F \cap \delta^-(S) = \emptyset$
      S is a "source" in decomp. of F into S.C.C's.

e.g.



(digraph has decomp where if S.C.C.'s are contracted, left with DAG).

e.g.



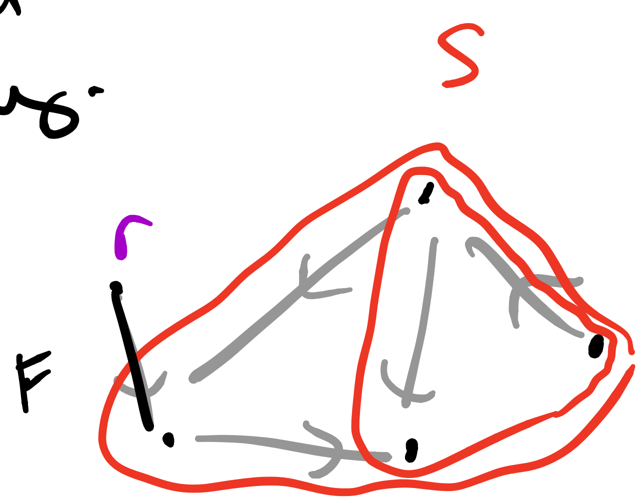
S is a subset of vertices,

F subset of edges.

does S ⊆ vertices "touched by" F?

not necessarily.

initially



▷ increase y_S until new inequality

$$\sum_{S : e_k \in \delta^-(S)} y_S \le c(e_k)$$

becomes an equality.

note $e_k \notin F$ b/c $F \cap \delta^-(S) = \emptyset$.

(y remain dual feas, b/c it was before)

▷ $F \leftarrow F + e_k$, $k \leftarrow k+1$

new F, y don't violate (b) because e_k is tight.

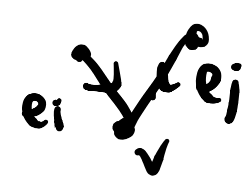
▷ Return F, y satisfying (b), & everything reachable from r in F.

Phase 2: eliminate as many edges as we can in reverse order they were added.

- ▷ **For** $i = k \ldots\ldots 1$:
  - ▷ **If** $F - e_i$ contains a directed path from r to every vertex, $F \leftarrow F - e_i$.
- ▷ **Return** $A = F$

Claim 1: A is an arborescence

Pf:
- We'll show $|A| = |V| - 1$ & $d^-(v) = 1$ $\forall v \in V - r$
- If indegree < 1 for $v \neq r$, contradicts reachability in A
- if $|A| > |V| - 1$ then collision



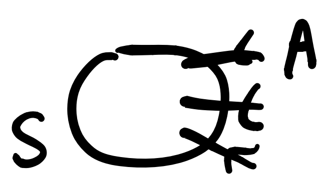
• suppose $i<j \Rightarrow$ in reverse delete, would have removed e_j b/c any vertex reachable through e_j is reachable through e_i. $\square$

finally:

Claim 2: Condition (a) of complementary slackness holds.

a.) $y_S > 0 \Rightarrow |A \cap \delta^-(S)| = 1$.

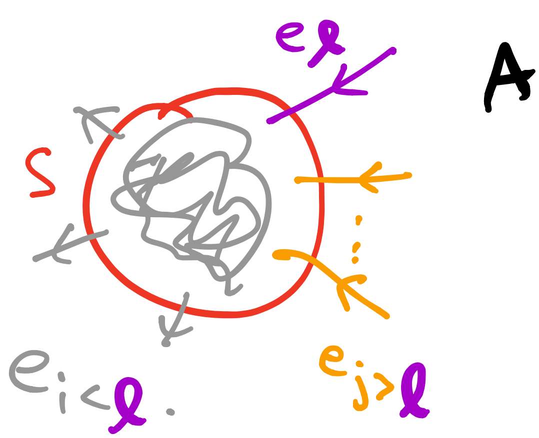
Pf: Assume not $\exists S$ s.t. $y_S > 0$ & $|A \cap \delta^-(S)| > 1$



• S was chosen at some step ℓ of phase 1 when we added e_ℓ to F.

- F had no other edges in $\delta^-(S)$ when e_ℓ was added.
  (by construction)
  $\Rightarrow$ all edges of $A \cap \delta^-(S)$ are e_j for $j > \ell$.

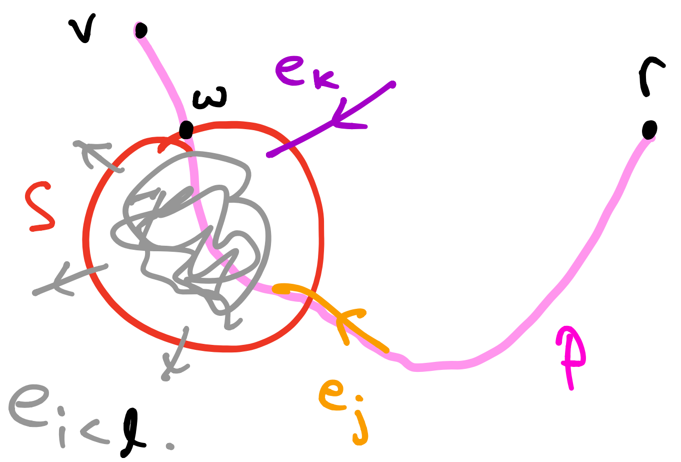
- When S chosen, F strongly connected within S
  $\Rightarrow$ S strongly connected using only e_i $i < \ell$.



- Subclaim: All e_j, $j > \ell$ should have been removed in Phase 2.

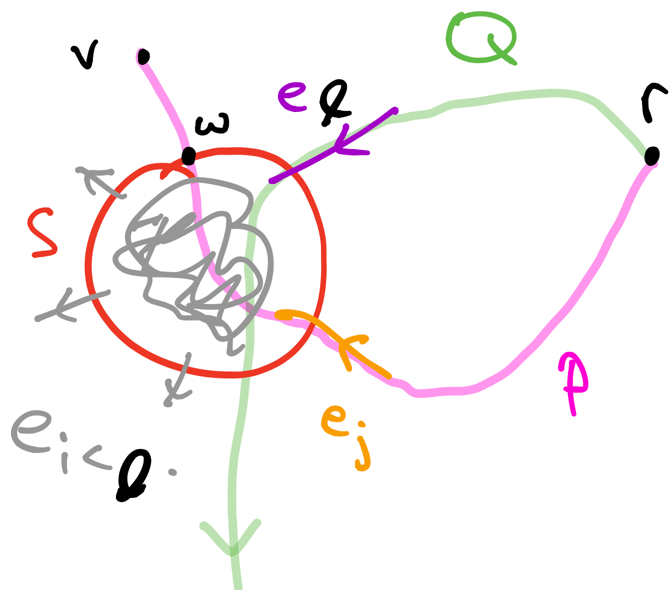
Why? suppose e_j necessary to visit some vertex v

- Let P $r \to v$ path using e_j
- Let w last vertex in S on P



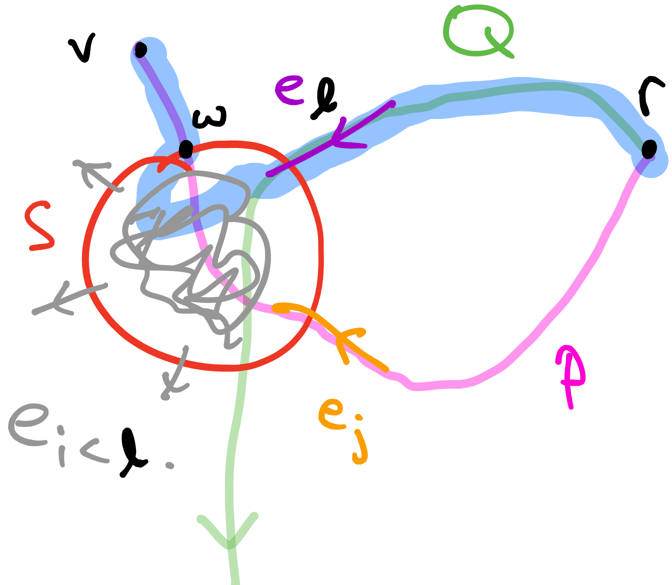
Note: P first enters S through e_j, else could shortcut e_j because S strong conn.

- Because e_ℓ is necessary at Step ℓ of Phase 2, there must be another path Q through e_k.



similarly: Q must enter S first through e_ℓ.

- Can use Q to shortcut e_j; thus e_j not necessary.



v
w
e_ℓ
Q
r
S
p
e_j
$e_{i<\ell}$.

□.