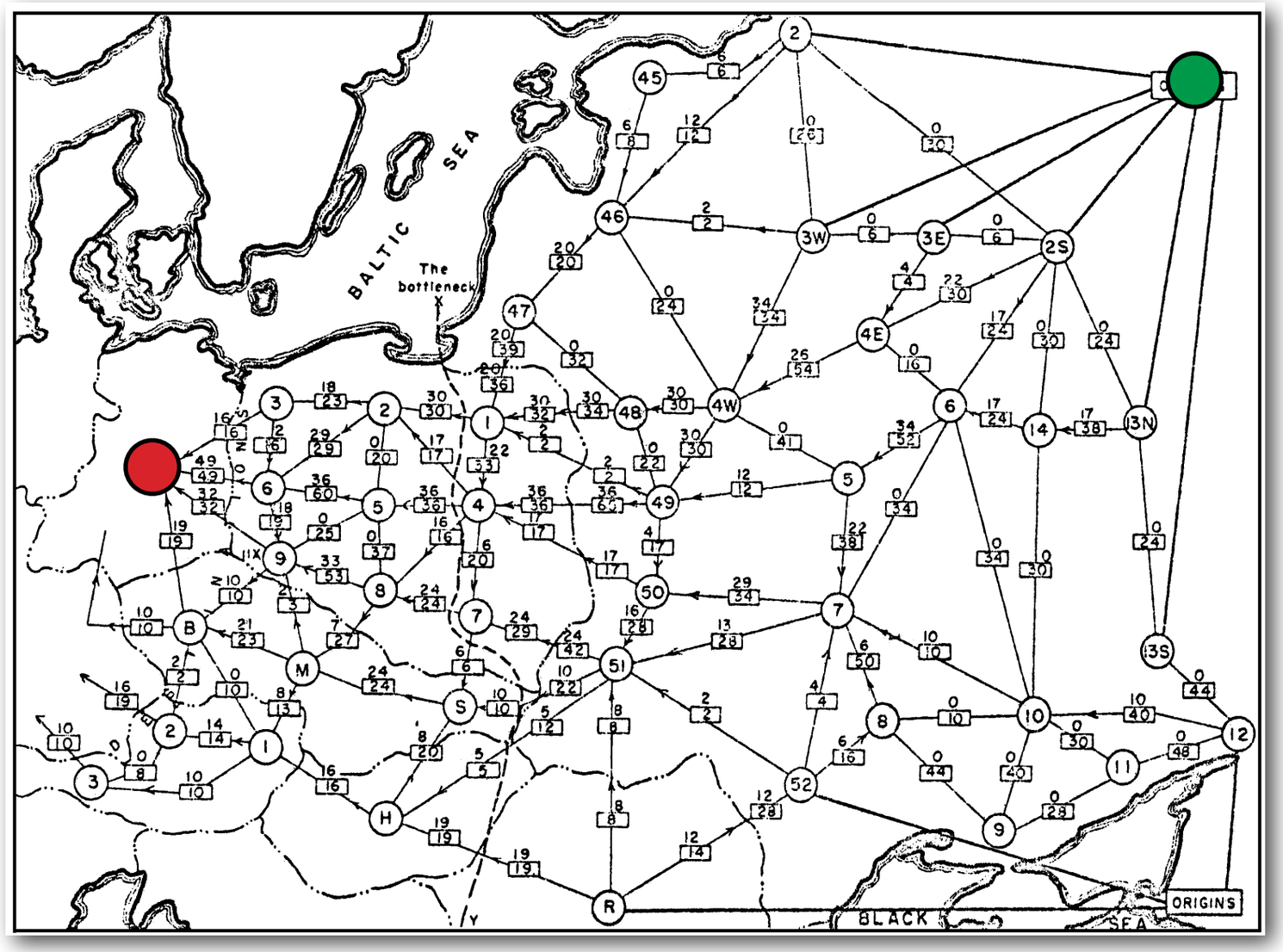
One-Dimensional Computational Topology

IV. Minimum cuts and maximum flows

Jeff Erickson
University of Illinois, Urbana-Champaign

Maximum flows

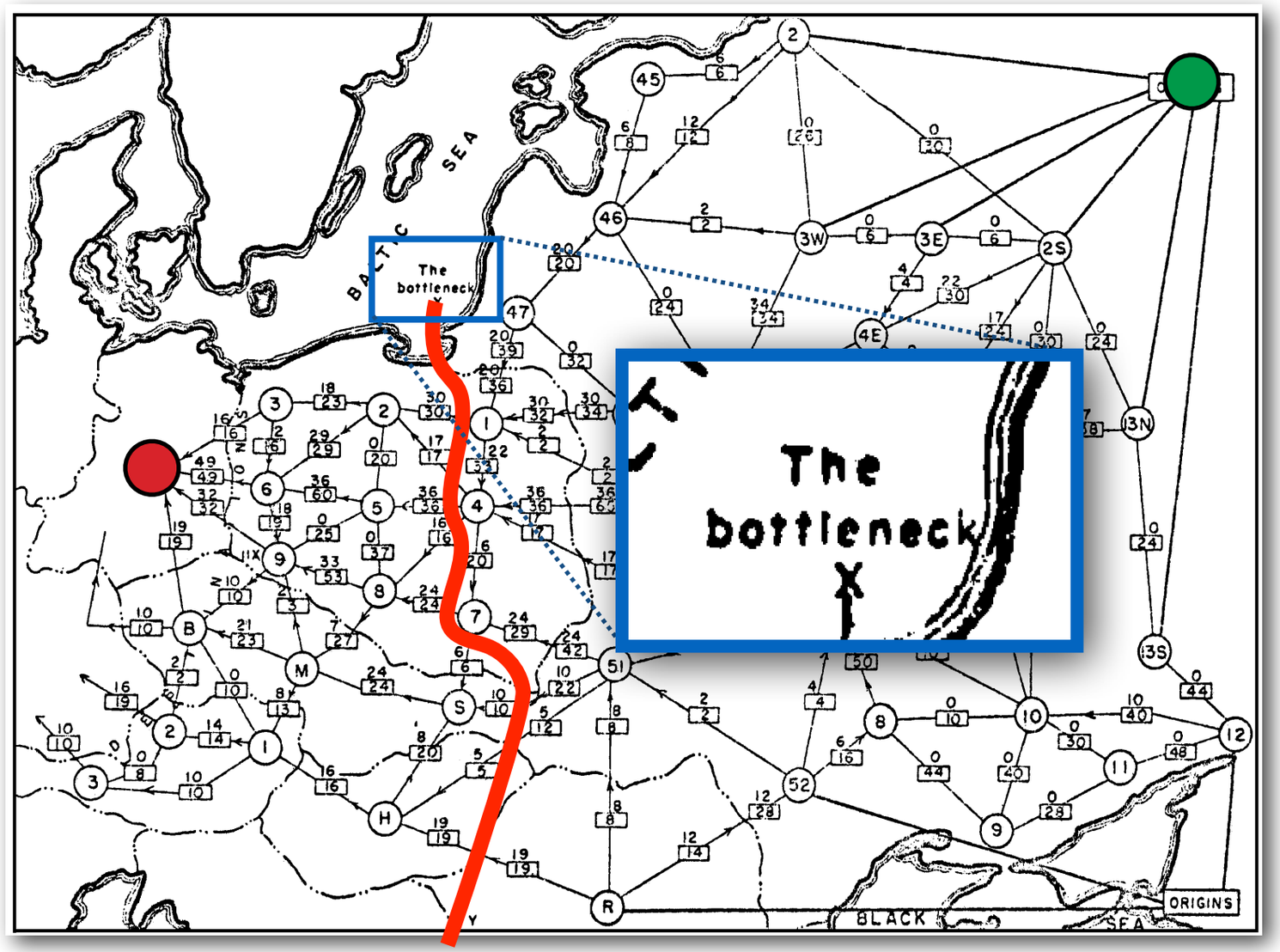
How quickly can we move soldiers from Moscow to Berlin?



[Harris Ross ’55, Ford Fulkerson ‘56]

Minimum cuts

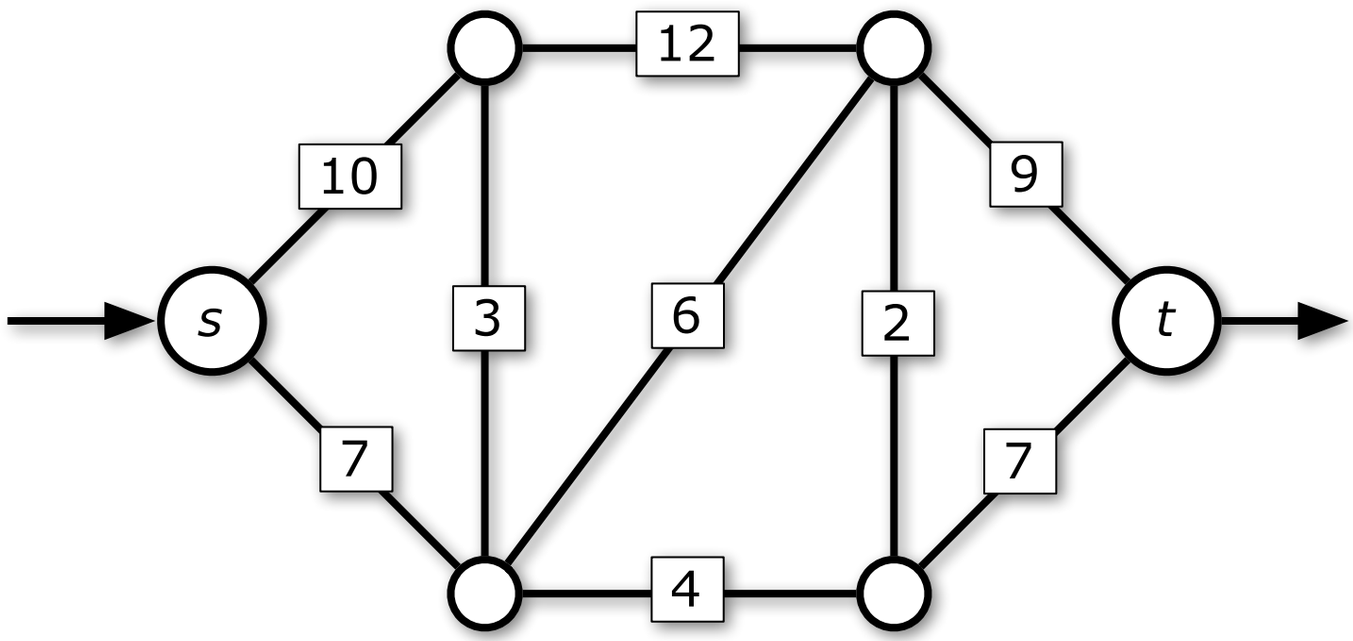
How cheaply can we separate Moscow from Berlin?



[Harris Ross ’55, Ford Fulkerson ‘56]

Flow network

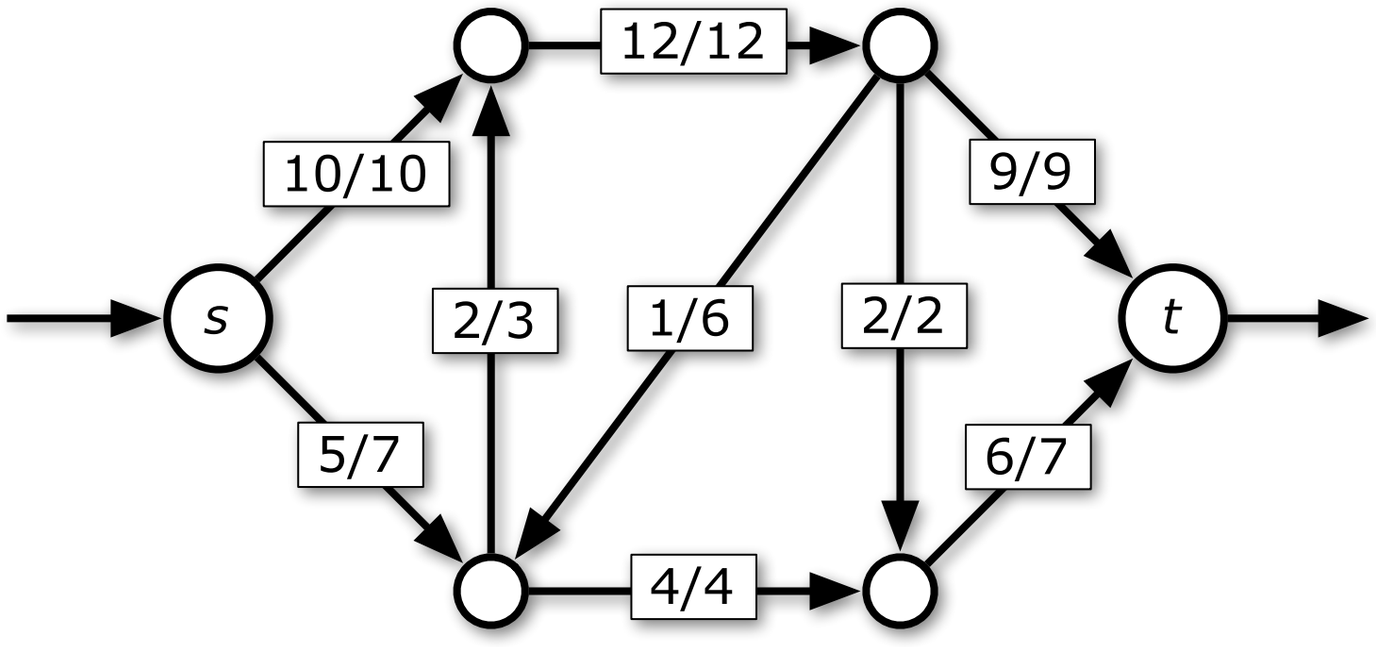
- An undirected graph $G = (V,E)$
- A capacity function $c : E \rightarrow \mathbf{R}^+$
- Two vertices: s (source) and t (target)



Maximum flow

Assign directions and non-negative “flows” to edges so that

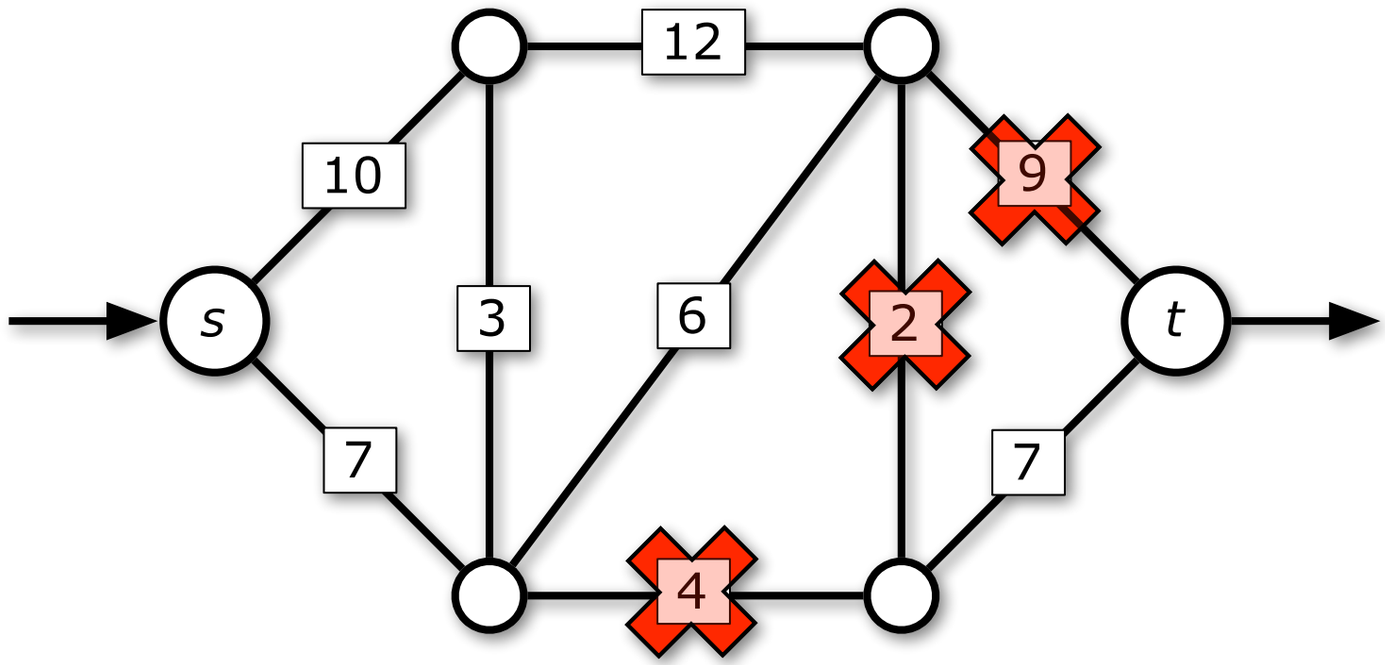
- ***Maximum***: Excess flow out of s (and into t) is maximized
- ***Feasible***: Flow through each edge is at most its capacity
- ***Flow***: Conservation at every vertex except s and t



Minimum cut

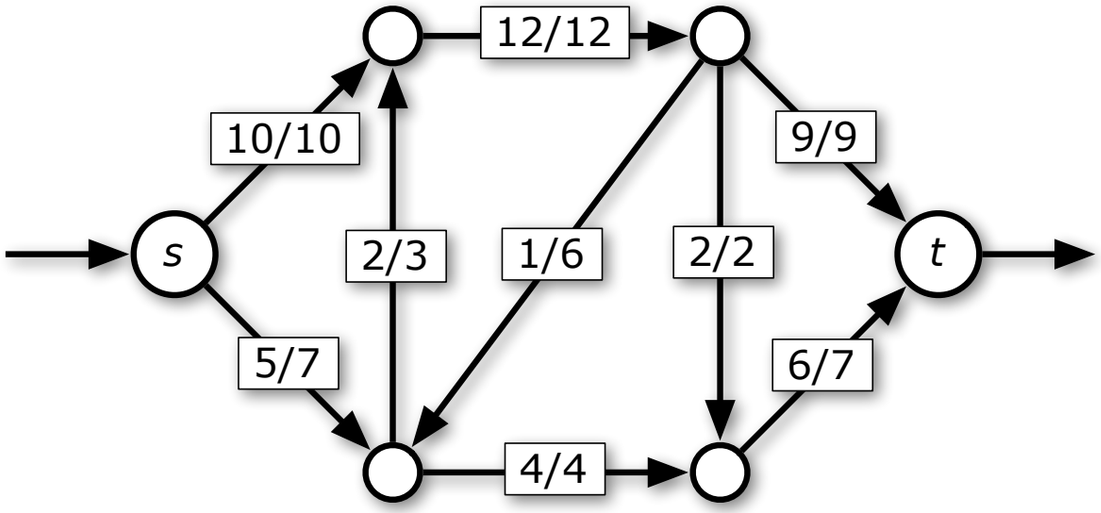
Find a subset ***X*** of edges such that

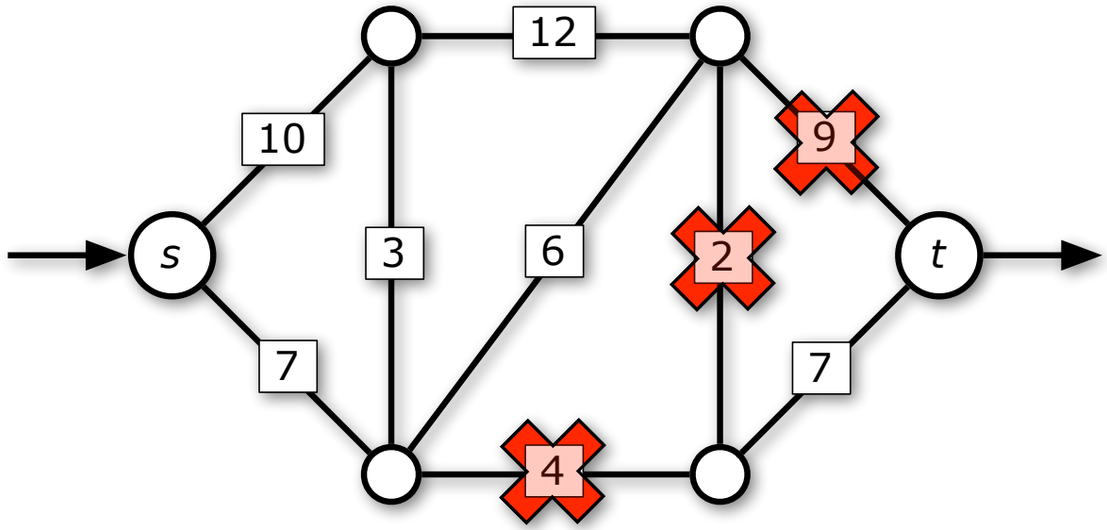
- ***Minimum***: The total capacity of the edges in *X* is minimized
- ***Cut***: Every path from *s* to *t* uses at least one edge in *X*



Max-flow Min-cut Theorem

- **Value of maximum flow = capacity of minimum cut**
- Special case of linear programming duality.





[Menger 1927] [Tucker 1948] [Ford Fulkerson 1954] [Elias Feinstein Shannon 1956]

Why do we care?

These are two of the most fundamental and broadly applied problems in combinatorial optimization.

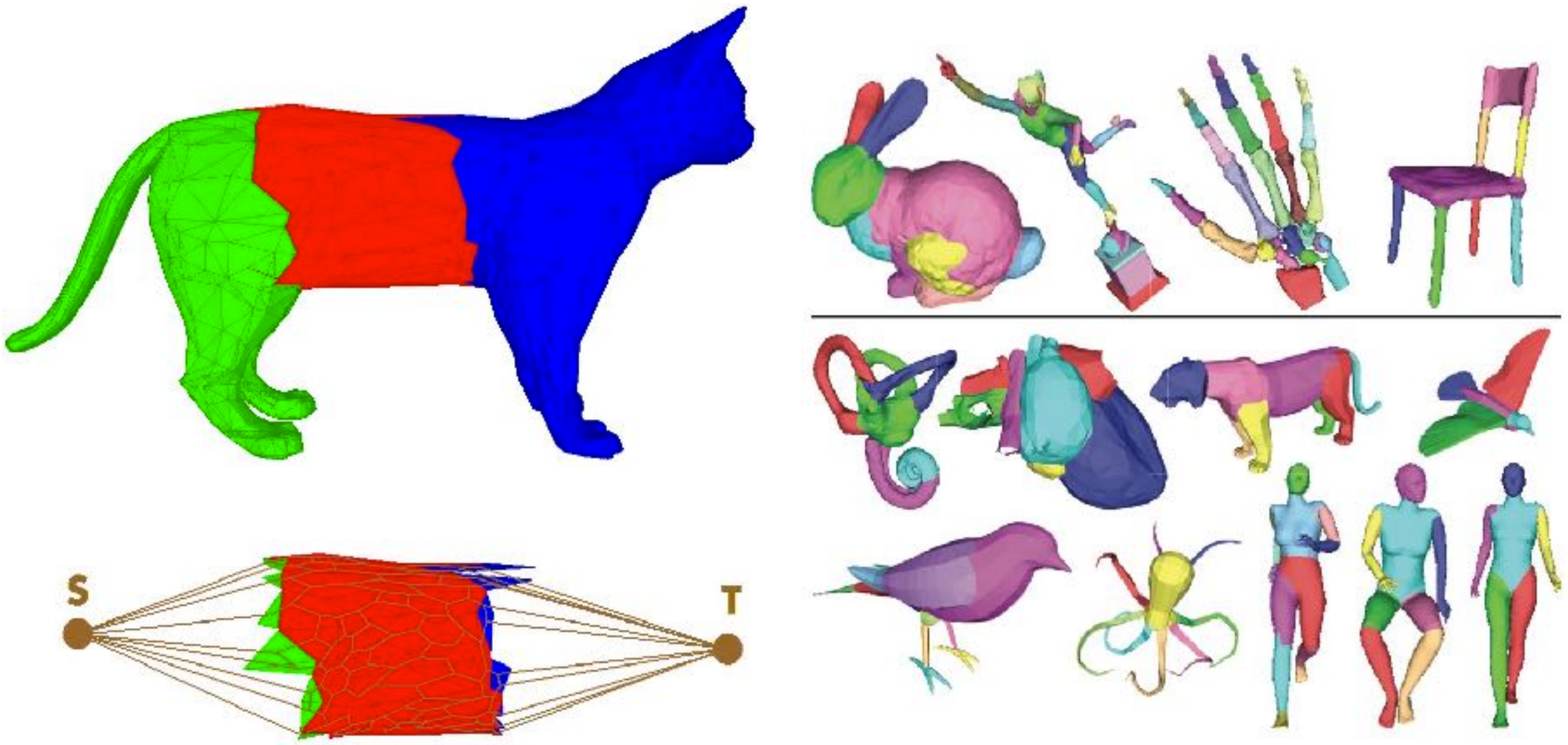
Applications:

- disjoint paths/routing
- assignment
- scheduling
- load balancing
- image/**mesh segmentation**
- baseball elimination
- **minimum surface computation**
- **homology localization**

Generalizations:

- minimum-*cost* flows
- multicommodity flows
- weighted matching
- network design
- spectral analysis of graphs
- metric embedding
- peg solitaire
- *linear programming*

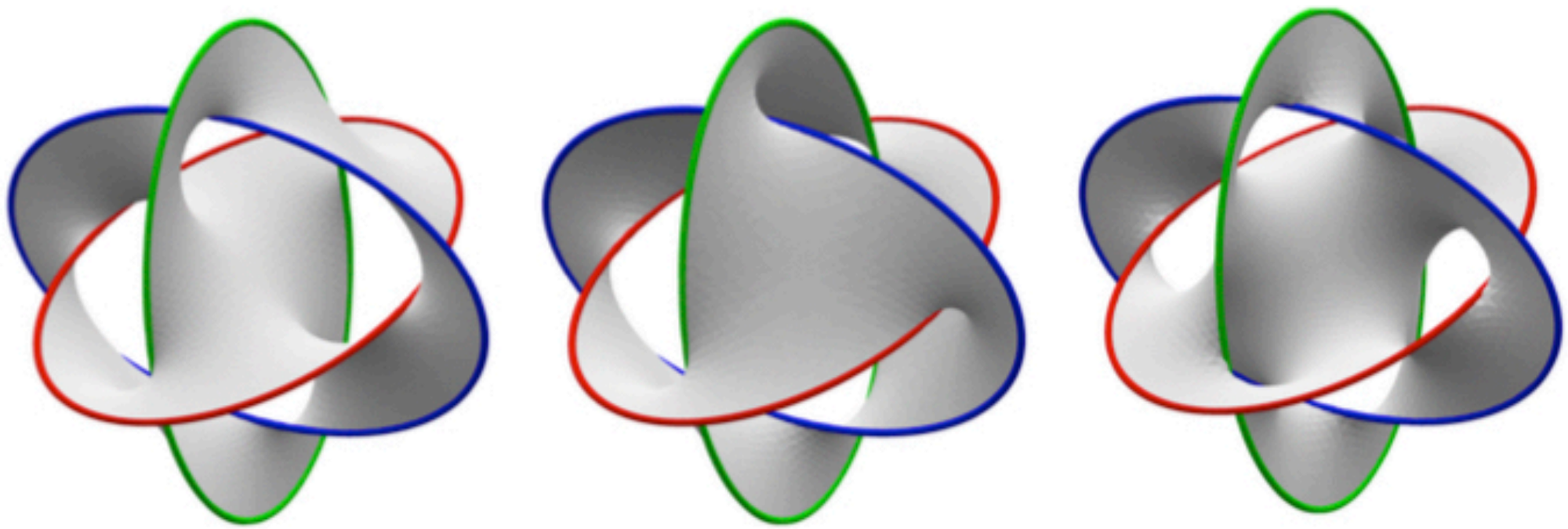
Surface segmentation



[Katz Tal ’03] [Katz Liefman Tal ’05] [Golovinskiy Funkhouser ’08]

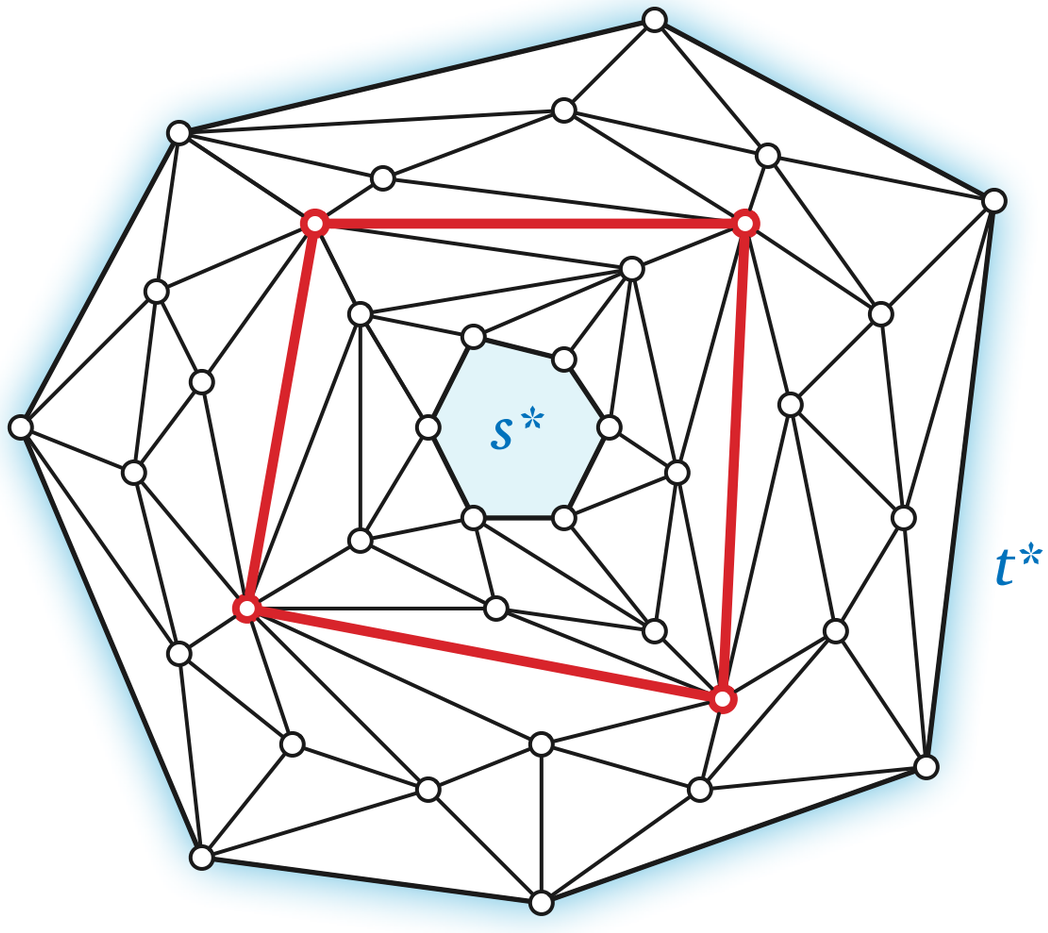
Minimal surfaces

- Plateau’s problem: Find a surface of minimum area with a given boundary curve



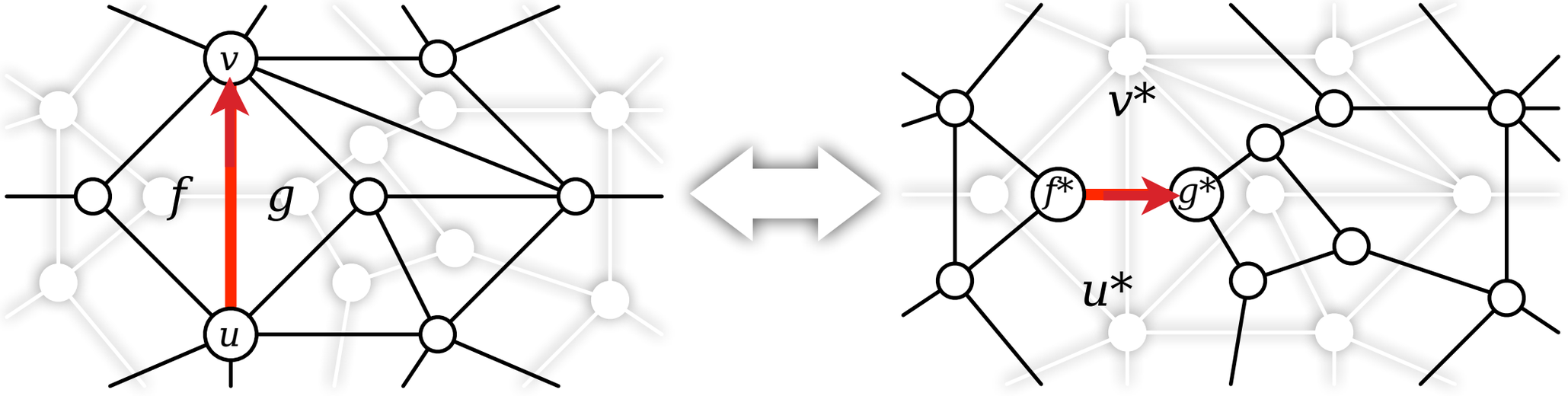
[Sullivan ‘90] [Hu et al ’92]...[Dey Hirani Krishnamoorthy ’10]...[Harrison Pugh ‘11]

Minimum cuts in planar graphs



Dual graph *G**

- faces of *G* ⟷ vertices of *G**
- edges of *G* ⟷ edges of *G**
- vertices of *G* ⟷ faces of *G**

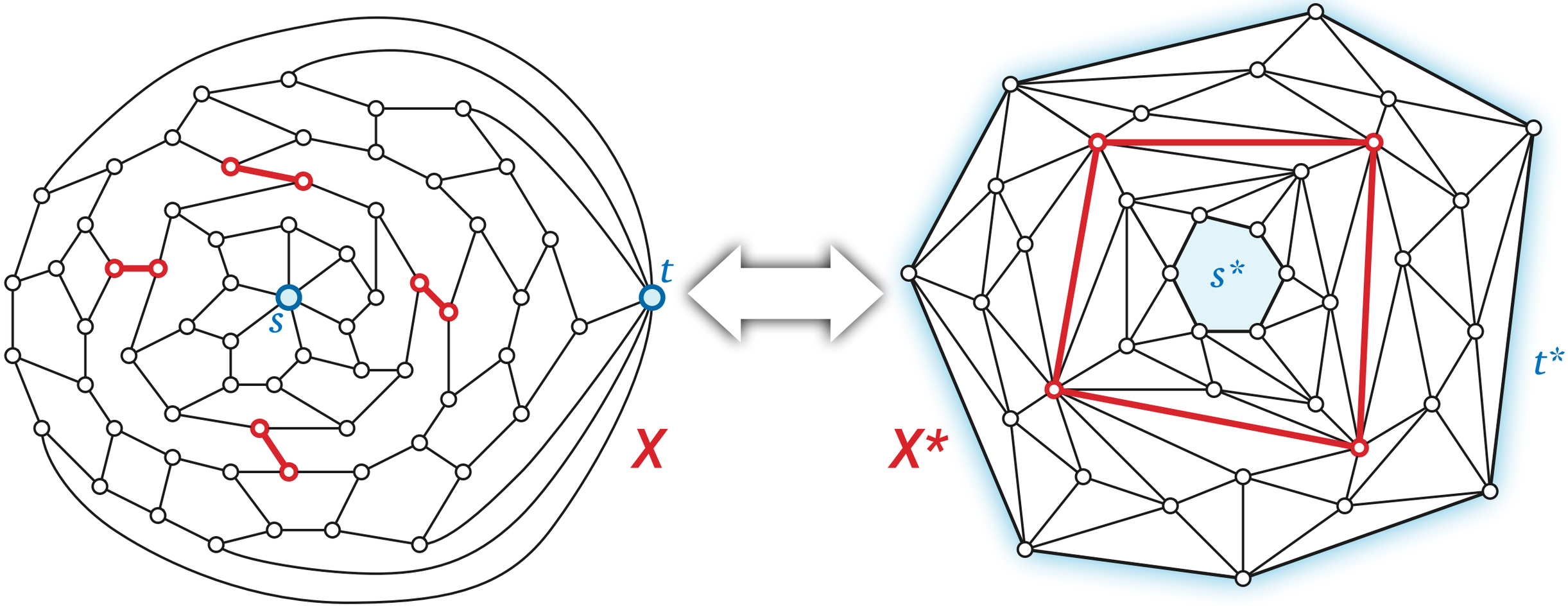


- Directed edges: *head*(*e**) := *right*(*e*)* and *tail*(*e**) = *left*(*e*)*
  (Intuitively, rotate 90° clockwise.)

Planar minimum cuts

X is the minimum (*s*, *t*)-cut in *G* if and only if

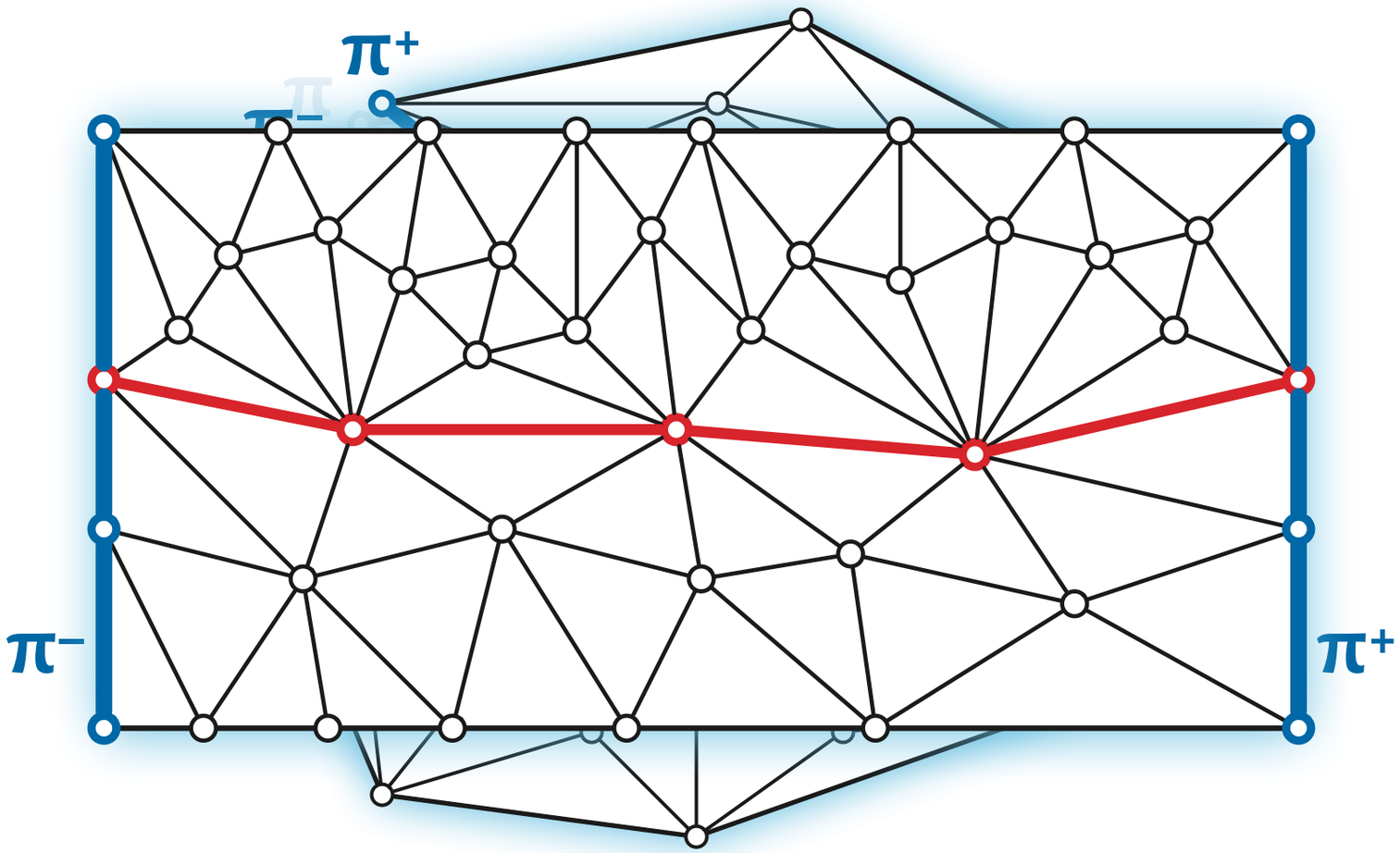
*X** is the minimum *cycle* in *G** separating *s** and *t**



[Jordan 1906] ***[Whitney 1932]*** *[Itai Shiloach '79]*

Planar minimum cuts

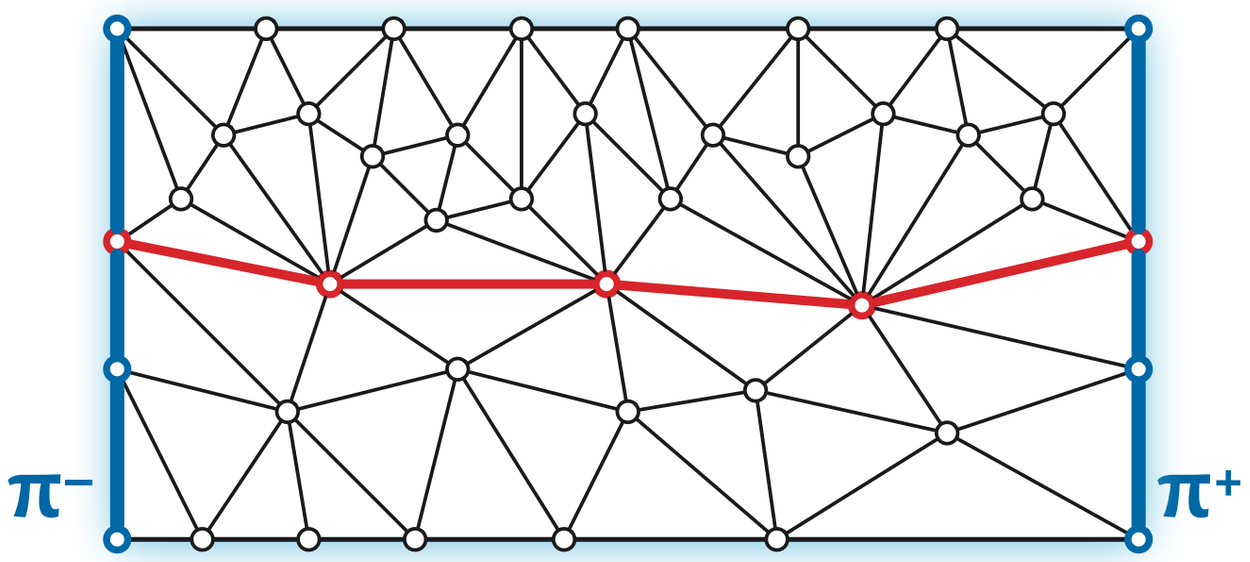
- Cut the dual annulus along a shortest path π
- X^* = shortest path between two copies of some vertex of π



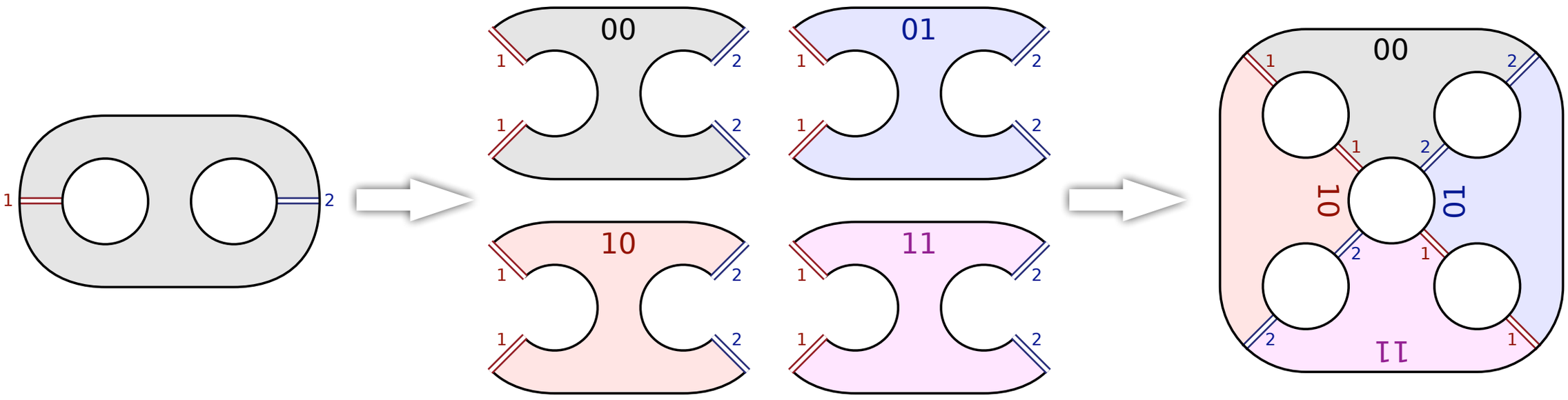
[Itai Shiloach 1979]

Planar minimum cuts

- Brute force: $O(n^2 \log n)$ *[Itai Shiloach '79]*
- Divide and conquer: $O(n \log^2 n)$ *[Reif '83]*
- Separator magic: $O(n \log n)$ *[Frederickson '87] [Henzinger et al. '97]*
- **Multiple source shortest paths:** $O(n \log n)$ *[Klein '05][Cabello Chambers '09]*
- Even more separator magic: $O(n \log \log n)$ *[Italiano et al. '11]*

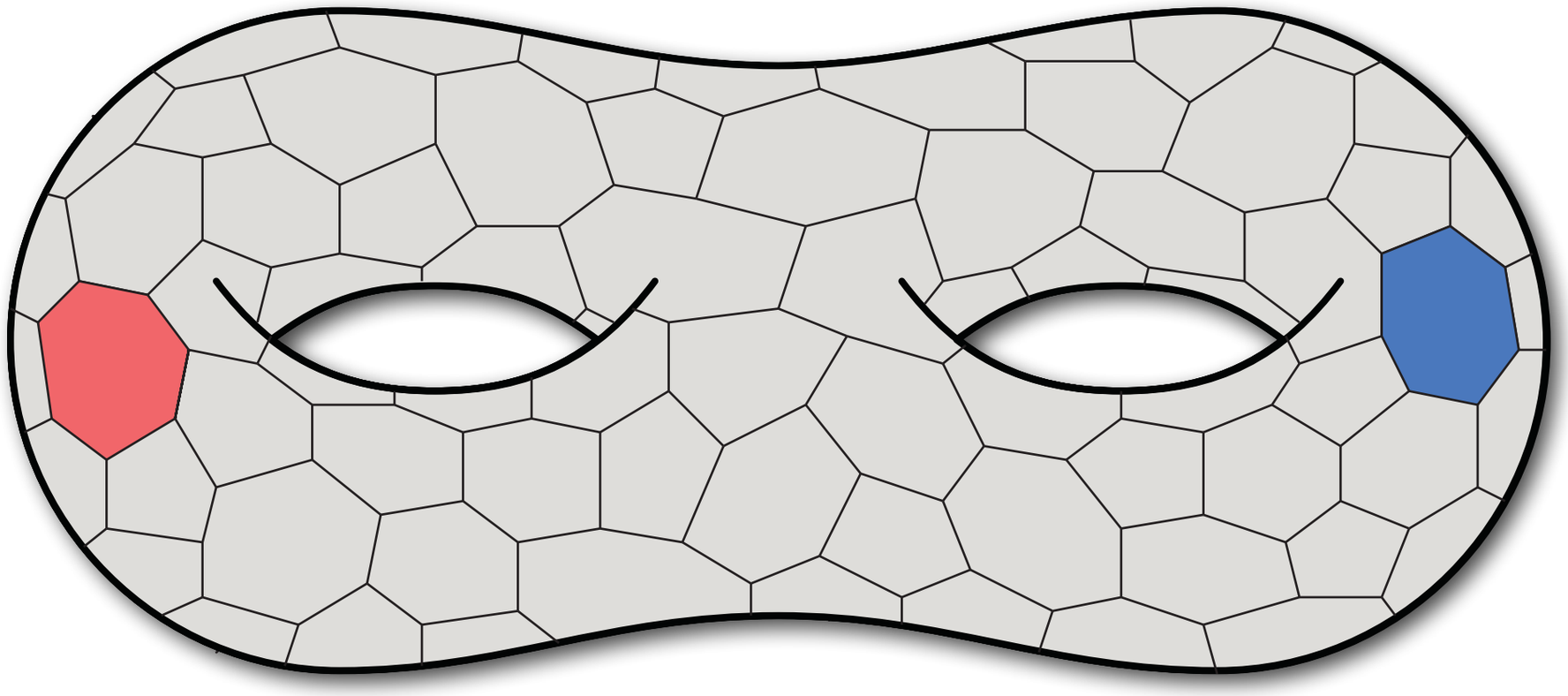


Minimum cuts in surface graphs



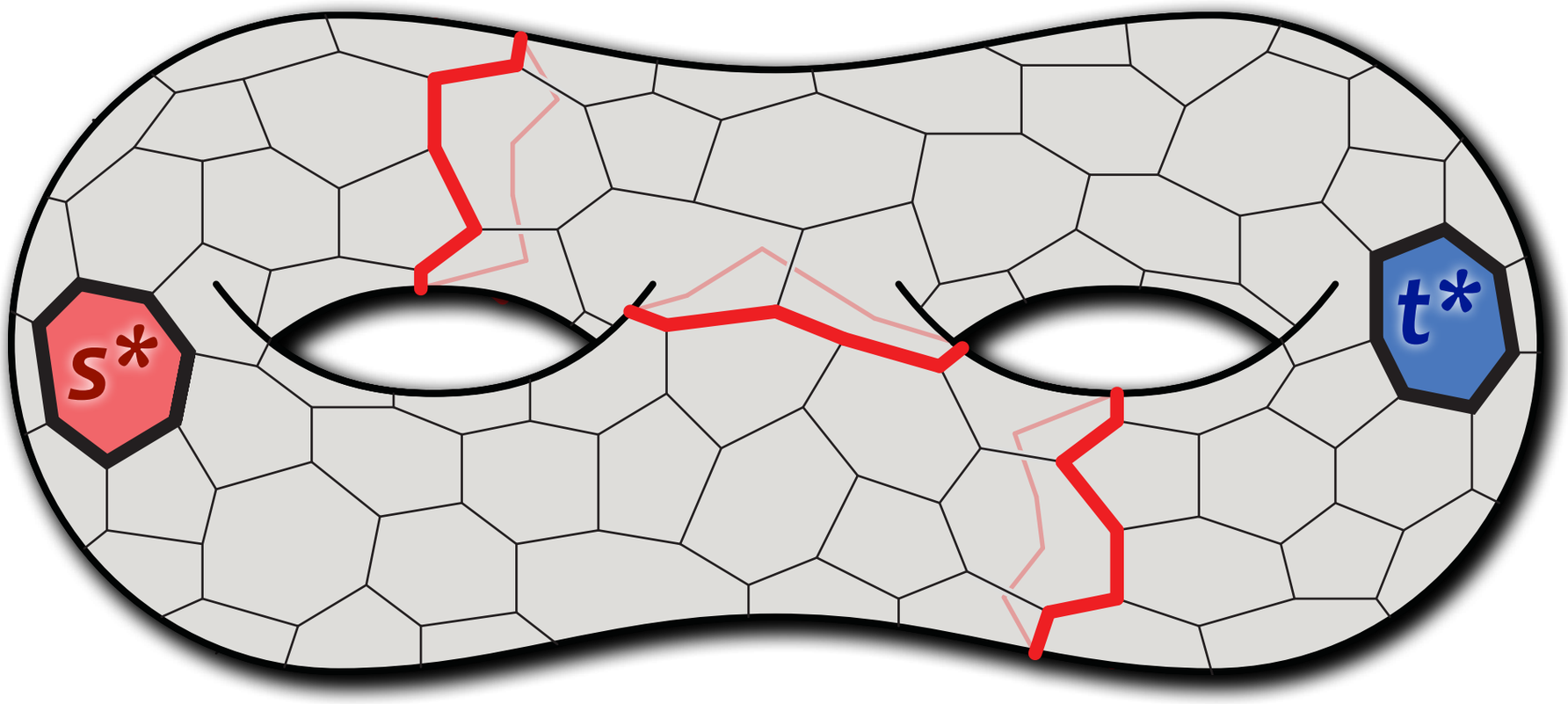
Surface graph duality

- Any surface graph G has a natural *dual* graph G^*:
  - vertices of G^* = faces of G
  - edges of G^* = edges of G
  - faces of G^* = vertices of G



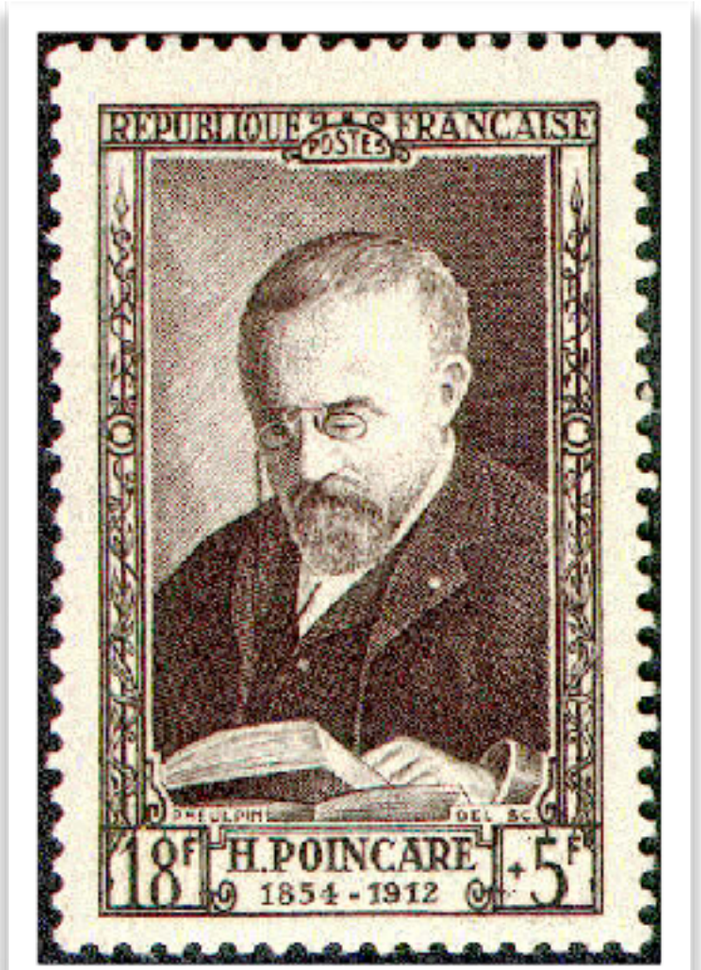
Minimum cuts in surface graphs

- X = minimum (s,t)-cut in $G \iff X^*$ = minimum-cost *subgraph* of G^* separating s^* and t^*
- X^* is the union of up to $g+1$ cycles



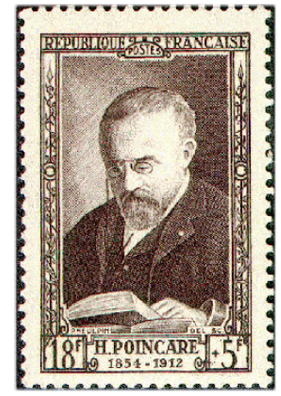
Z_2-homology

- Two subgraphs of *G** are **Z_2-*homologous*** if their symmetric difference is a boundary of the union of faces of *G**.

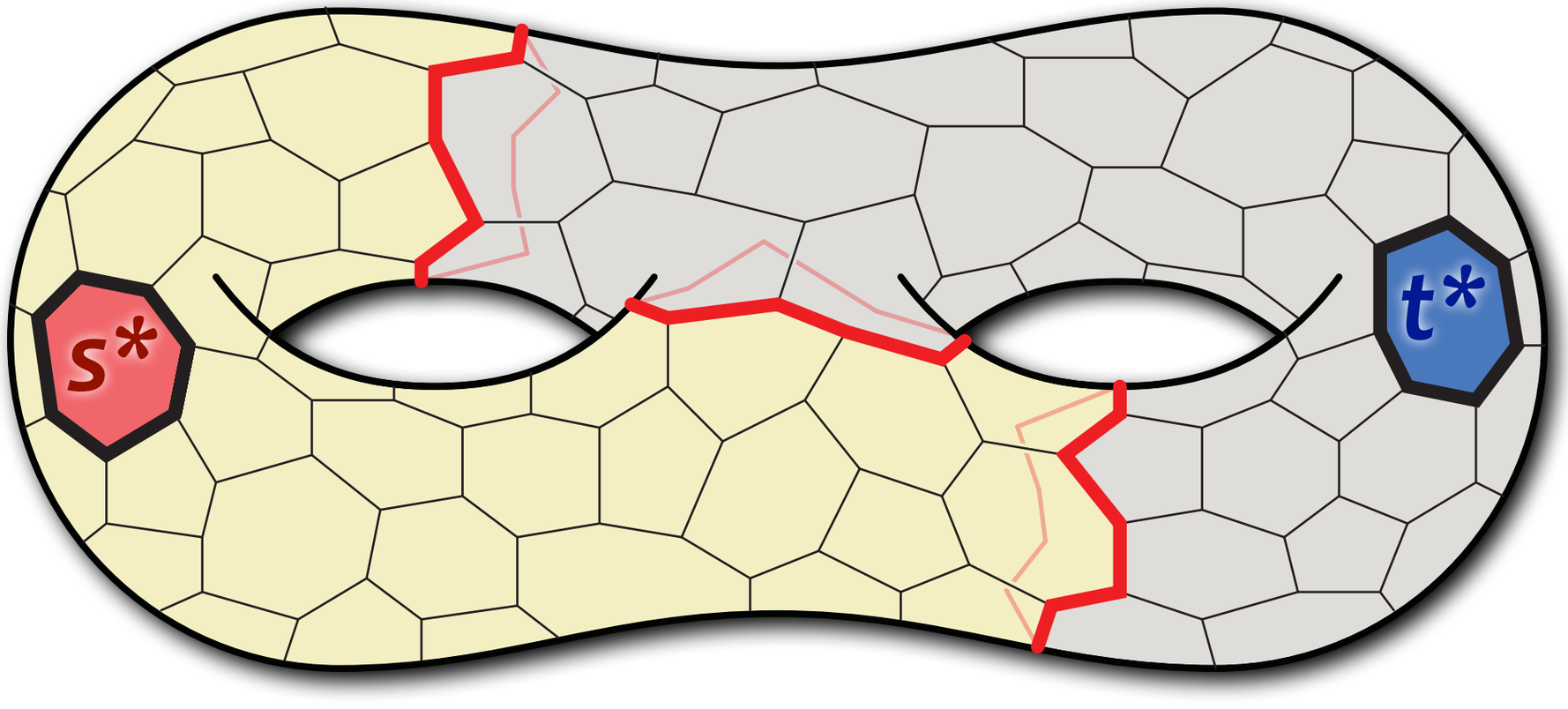


[Poincaré 1900]

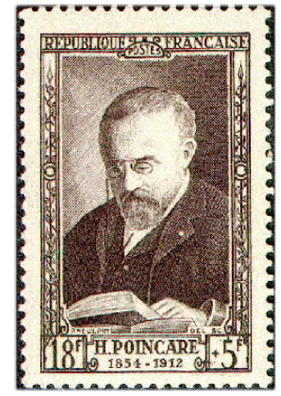
Z_2-homology



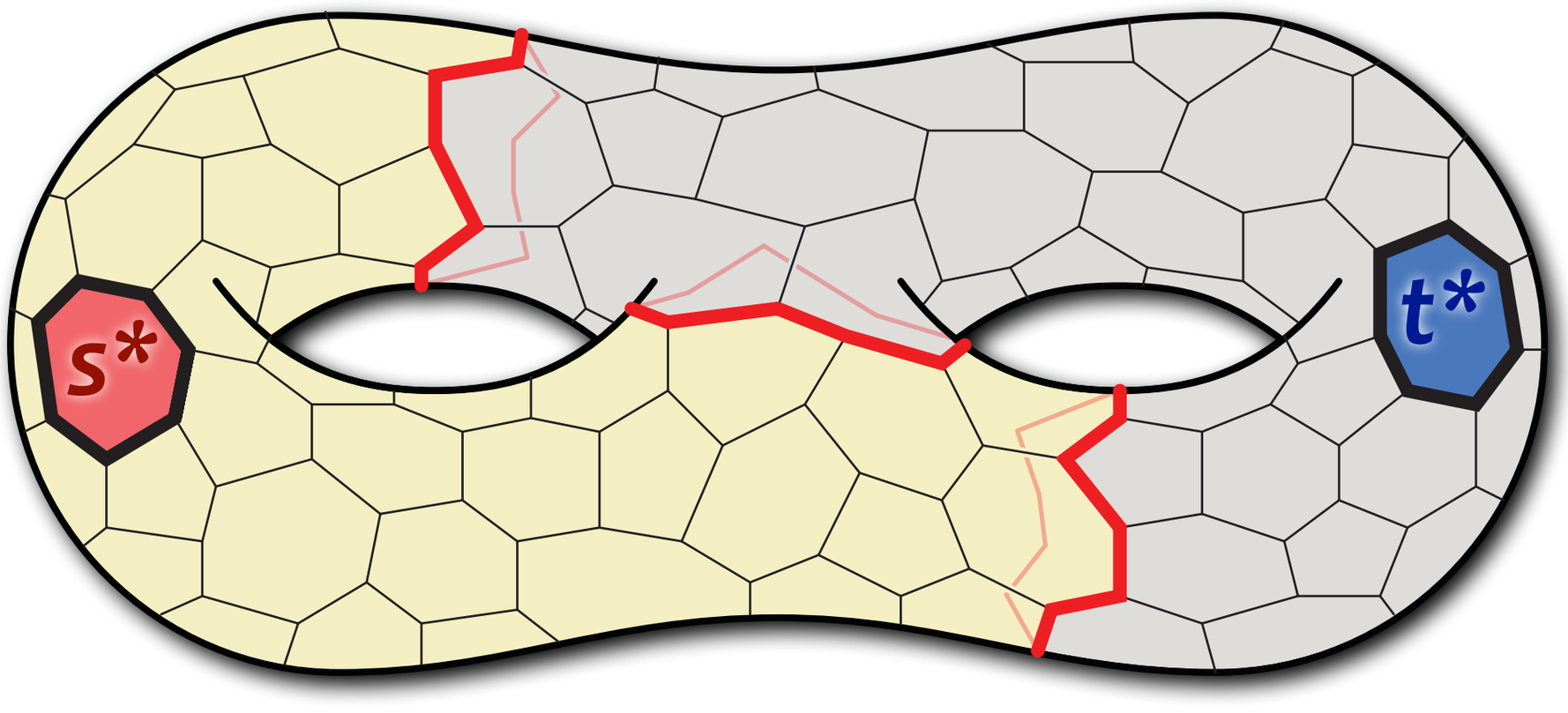
- Two subgraphs of *G** are ***Z_2-homologous*** if their symmetric difference is a boundary of the union of faces of *G**.



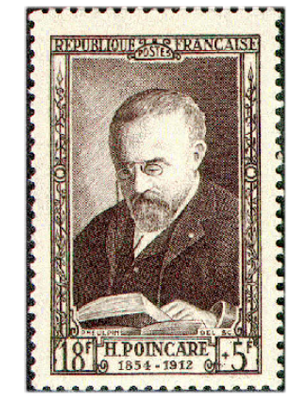
Z_2-homology of cuts



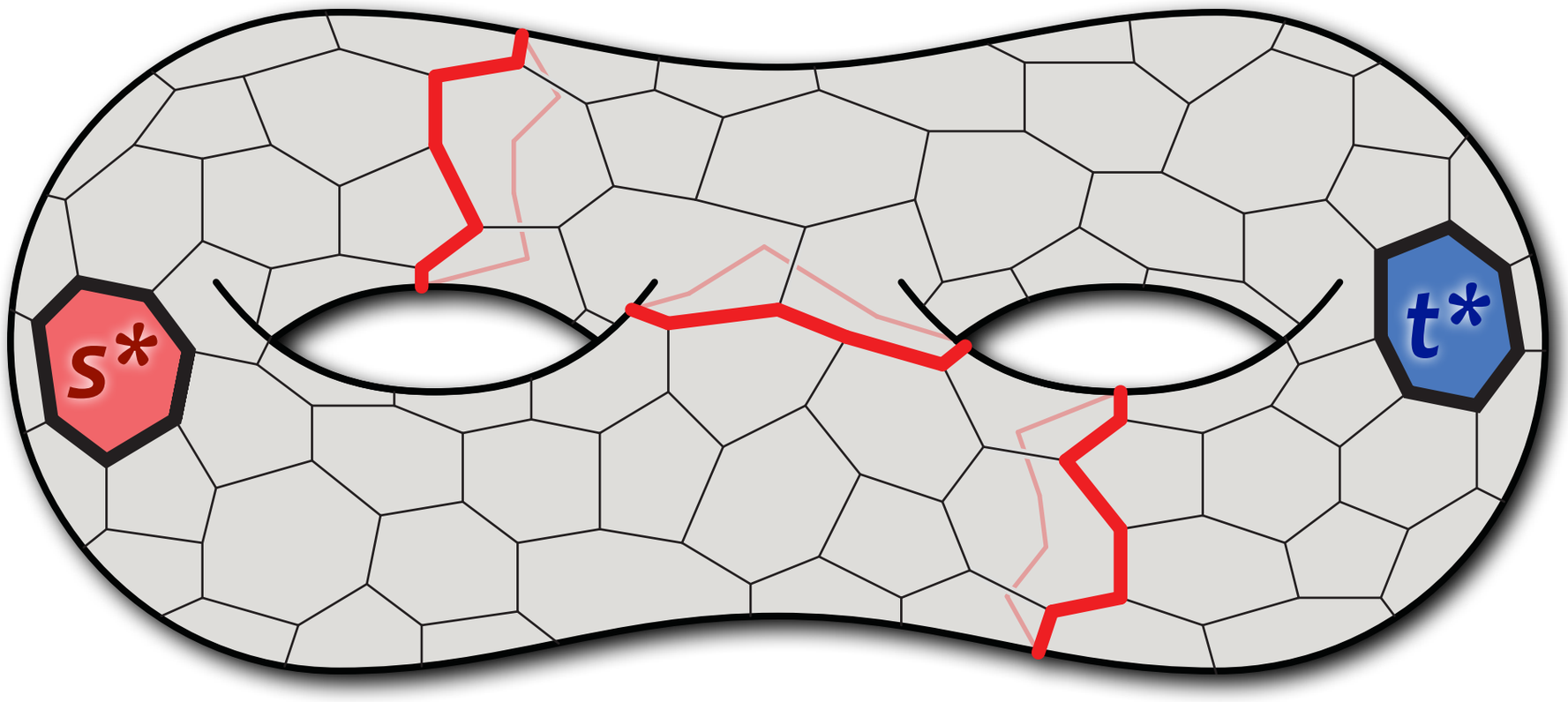
- X^* is the minimum-cost subgraph that is $\mathbf{Z_2}$-homologous with ∂s^* in the surface $\Sigma \setminus (s^* \cup t^*)$



Strategy



- Each component of X^* is the shortest cycle in its own $\mathbf{Z}_2$-homology class.
- Find the shortest cycle in every $\mathbf{Z}_2$-homology class, and then sew them together

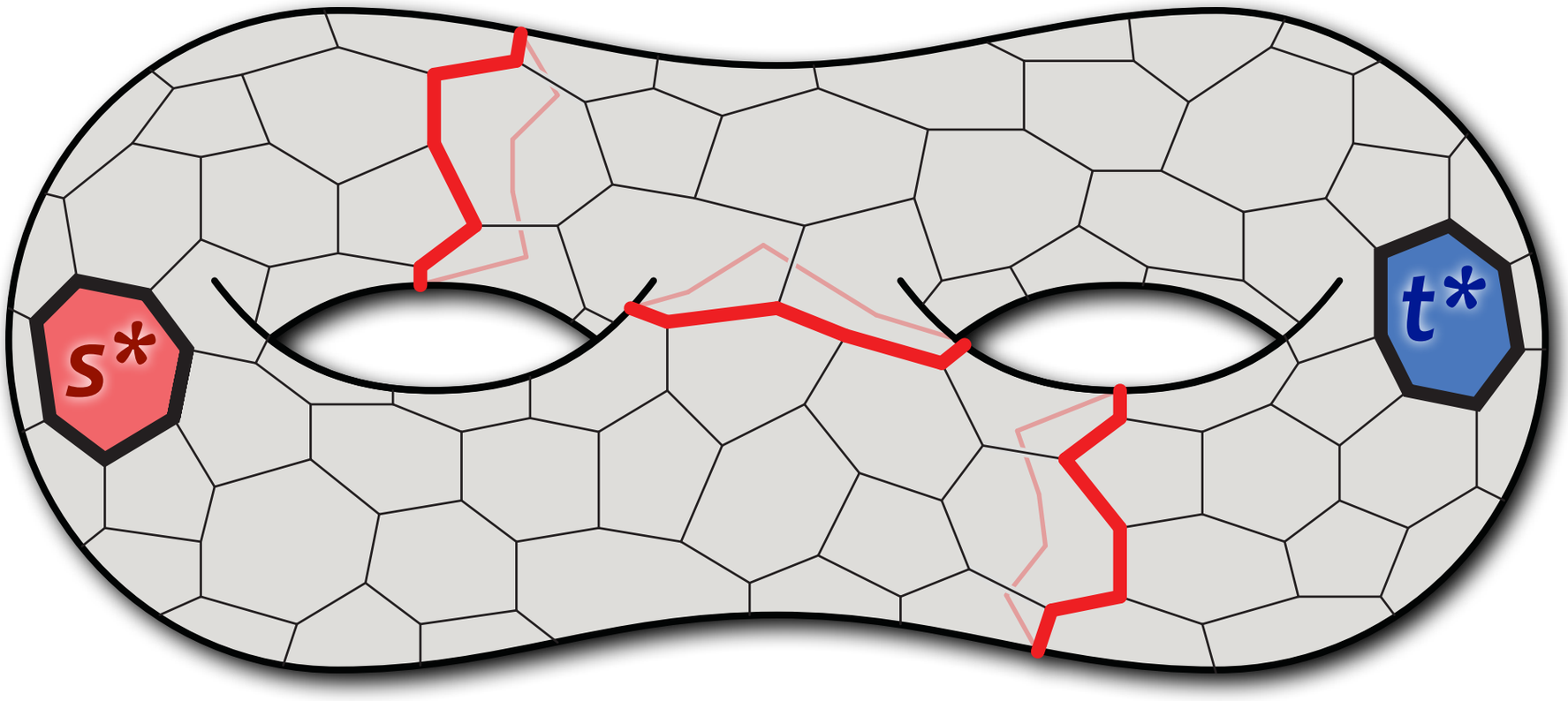


$\mathbf{Z}_2$-homology group

- $\mathbf{Z}_2$-homology classes define a finite vector space:

$$H_1(\Sigma \backslash (s^* \cup t^*); \mathbb{Z}_2) \cong \mathbb{Z}_2^{2g+1}$$

- There are exactly 2^{2g+1} distinct $\mathbf{Z}_2$-homology classes



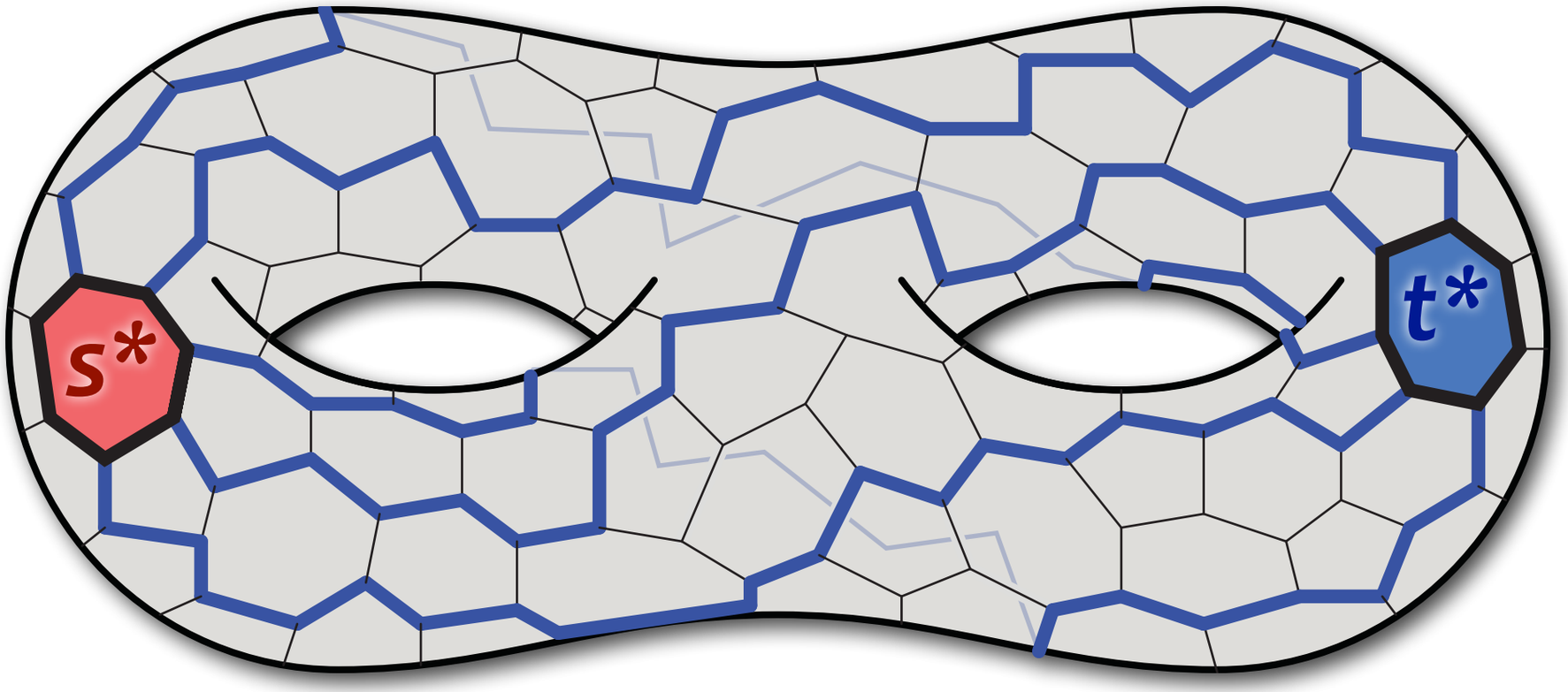
Some vector spaces

vector space	notation	dimension (#bits)
even subgraphs (cycle space)	Z_1	$\|E\|-\|V\|+1$
boundary subgraphs	B_1	$\|F\|-2$
homology classes of even subgraphs	$H_1 = Z_1/B_1$	$\|E\|-\|V\|-\|F\|+3 = \mathbf{2g+1}$

Euler's formula: $|V|-|E|+|F| = 2-2g$

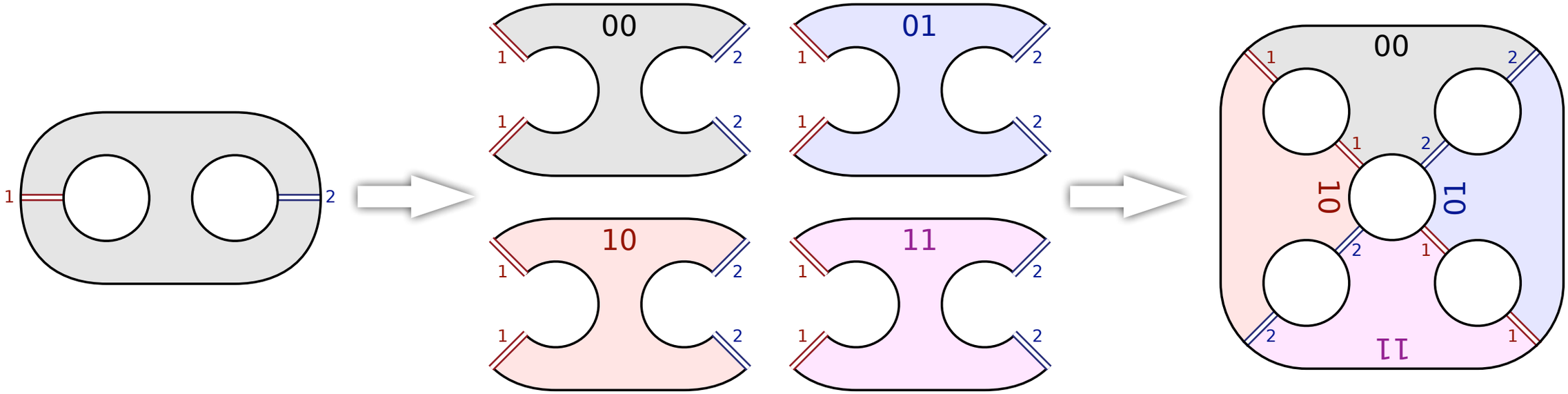
Measuring $\mathbf{Z}_2$-homology

- Cut $\Sigma \setminus (s^* \cup t^*)$ along $2g+1$ paths $\pi_1, \pi_2, \ldots, \pi_{2g+1}$ from s^* to t^* to get a disk D
- The $\mathbf{Z}_2$-homology class $[\gamma]$ of a cycle γ is characterized by which paths π_i cross γ an *odd* number of times.



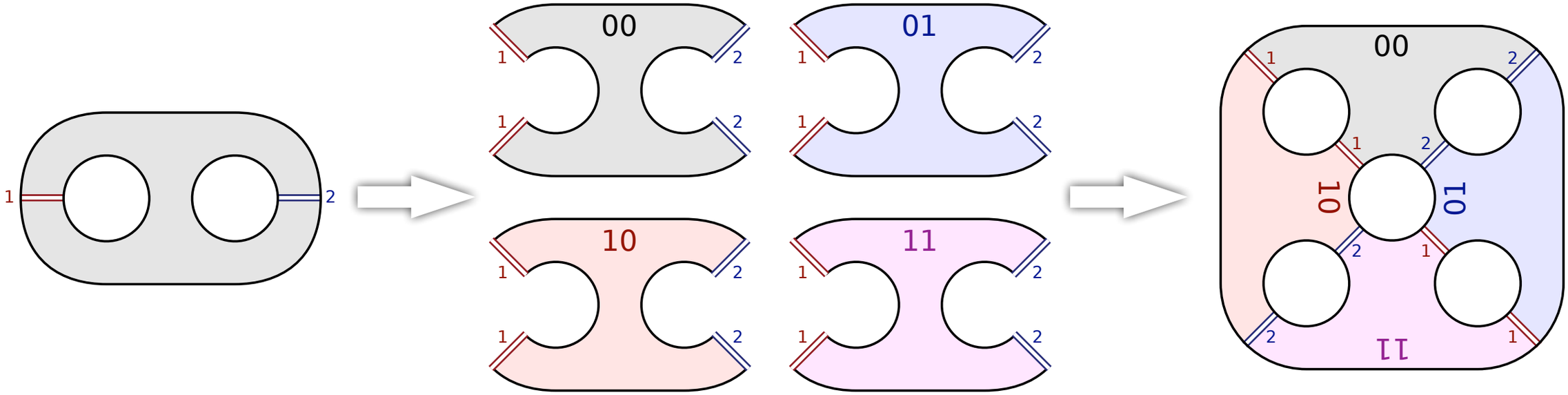
Z_2-homology cover

- Make 2^{2g+1} copies of D, one for each homology class
- Glue copies that differ by 1 bit along corresponding path π_i



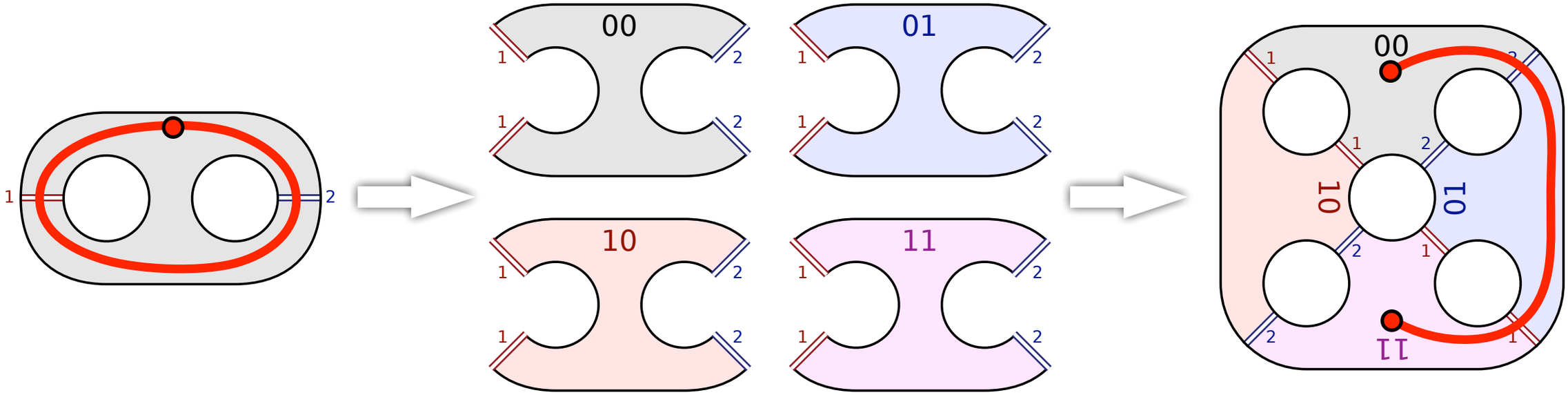
Z_2-homology cover

- Resulting graph $\hat{G}$ has $\hat{n} = 2^{2g+1} n$ vertices (v, h)
- Embedded on a surface with genus $\hat{g} = 2^{O(g)}$



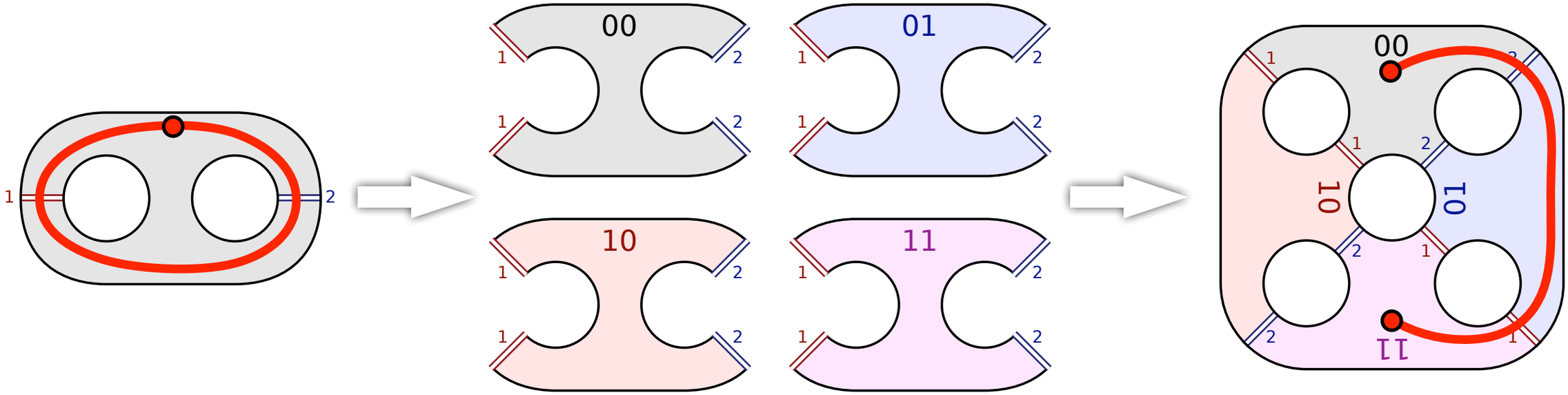
Z_2-homology cover

- Fix any vertex v of any cycle γ in G.
- Then γ is the projection of a path in $\hat{G}$ from $(v, 0)$ to $(v, [\gamma])$.



Shortest Z_2-homologous cycles

- *Shortest* cycle in G in any class h = projection of *shortest* path in $\hat{G}$ from $(v,0)$ to (v,h), for any vertex v on the cycle



Shortest $\mathbb{Z}_2$-homologous cycles

- The minimum cycle in ***every*** $\mathbf{Z}_2$-homology class can be computed in $O(g\,\hat{g}\,n \log n) = 2^{O(g)}\,n \log n$ time.
  - Using multiple-source shortest paths! *[Cabello Chambers Erickson 2013]*

- The minimum *subgraph* in every $\mathbf{Z}_2$-homology class can be computed in $2^{O(g)}$ additional time by dynamic programming.

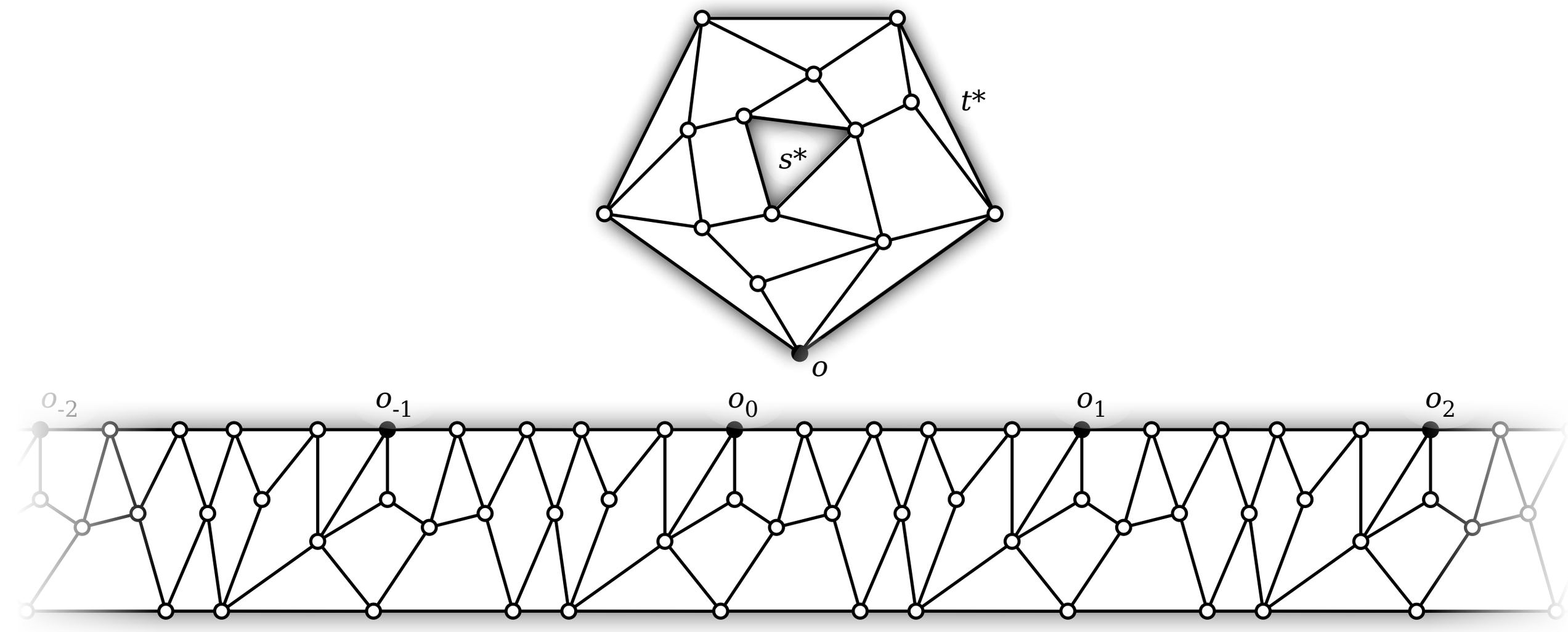
Summary

[Erickson Nayyeri 2011]

- Minimum (s,t)-cuts in *undirected* surface graphs can be computed in $2^{O(g)}\, n \log n$ time.

- Unfortunately:
  - Solves exponentially many instances of an NP-hard problem!
    [Chambers, Colin de Verdiére, Erickson, Lazarus, Whittlesey '08]
    [Cabello, Colin de Verdiére, Lazarus '11] [Chambers Erickson Nayyeri '09]
  - Fails for directed graphs, *even in the plane!* The only approach known here is to compute the maximum flow.

Maximum flows in planar graphs



Duality with shortest paths

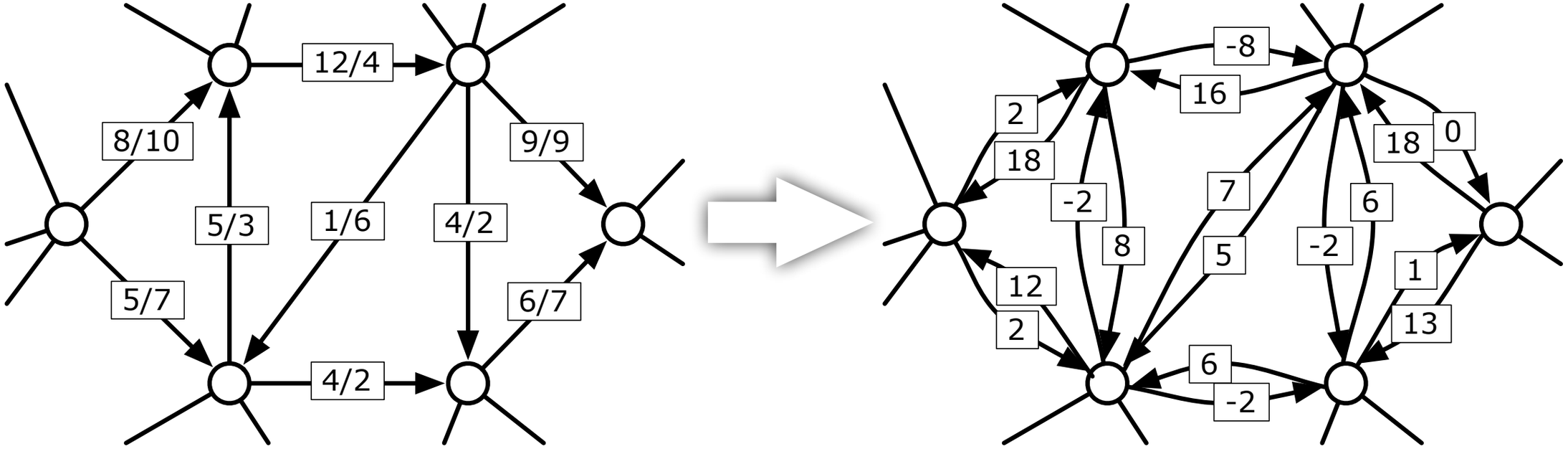
Let f be an arbitrary flow in a planar flow network G.

There is a *feasible* flow with value $|f|$ in G
if and only if
the *dual residual network* G_f^* has no negative cycles.

[Hassin '81] ***[Venkatesan '83]*** *[Miller Naor '89, '95]*

Residual network G_f

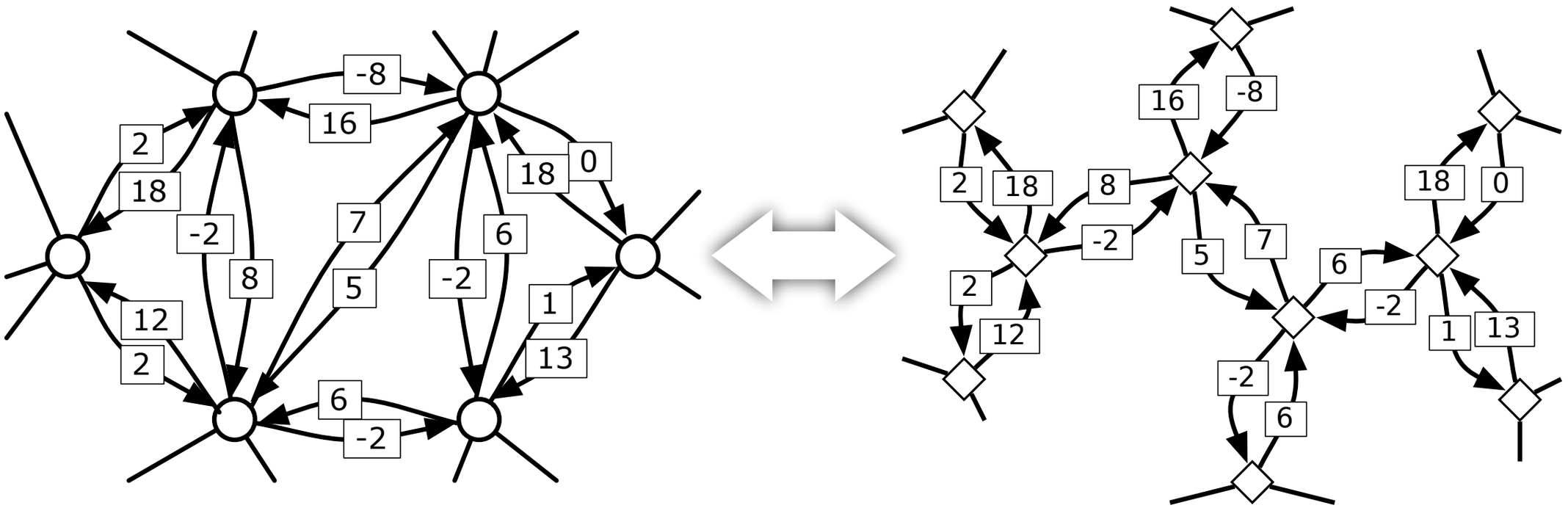
- Fix a directed planar graph G with edge capacities c.
  - Any flow f defines *residual capacities* $c_f(e) := c(e) - f(e)$
  - f is feasible $\iff$ $c_f(e) \geq 0$ for every edge e



[Ford Fulkerson ’55]

Dual residual network G_f^*

- residual capacities $c_f(e)$ $\longleftrightarrow$ costs $c_f(e^*)$

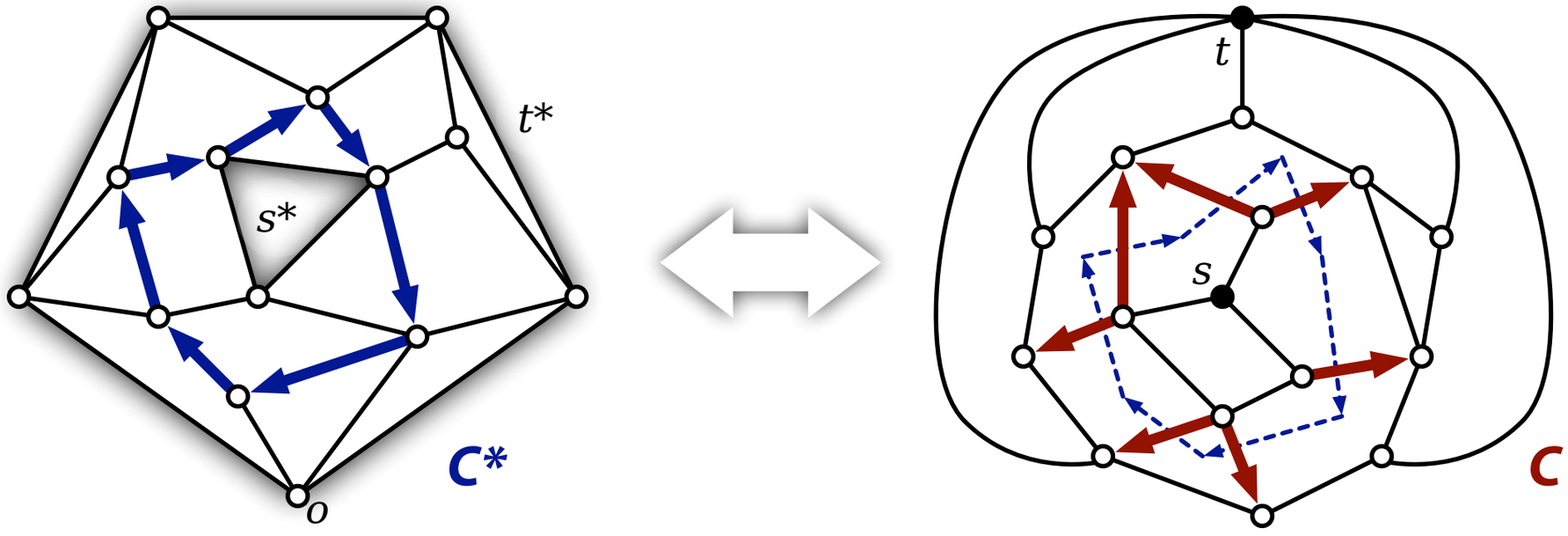


Proof

- First, suppose G_f^* has no negative cycles.
  - Fix a dual "origin" vertex o.
  - Let $dist_f(p)$:= *shortest-path distance* from o to p in G_f^*.
  - Let $\varphi(e) := dist_f(right(e)^*) - dist_f(left(e)^*) + f(e)$.
  - It is easy to check that φ is a feasible flow with value $|f|$.

Proof

- On the other hand, suppose G_f^* has a negative cycle C^*.
  - Then the dual of C^* is a cut C with capacity less than $|f|$.



[Whitney '32]

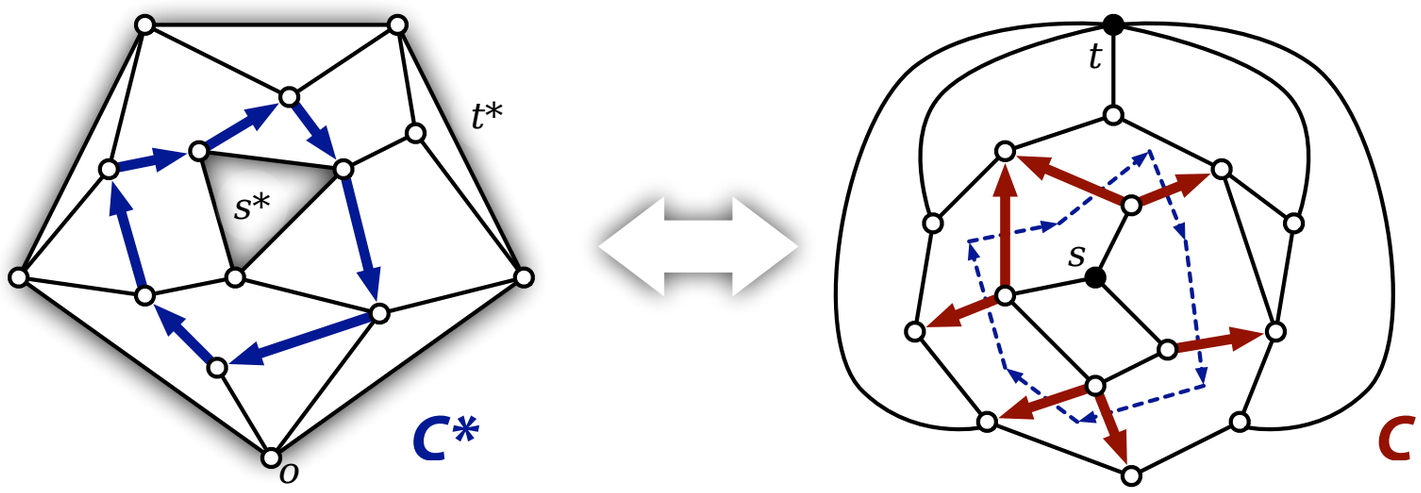
Parametric shortest paths

[Karp Orlin '81]

- Duality reduces planar max-flow to the following problem:

> Given a graph G_λ^* whose edge weights are linear functions of a *parameter* λ, find the largest value of λ such that G_λ^* has no negative cycles.

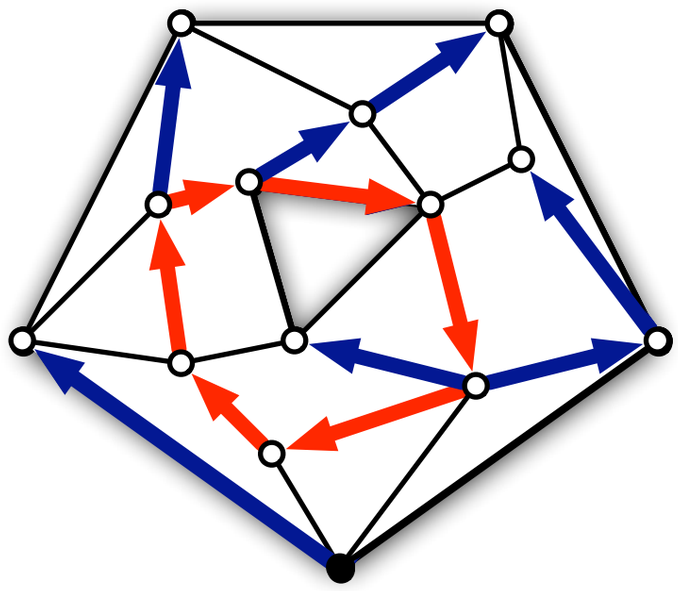
- Fix an arbitrary flow f with value 1, and define $G_\lambda^* := (G_{\lambda \cdot f})^*$



Parametric shortest paths

[Karp Orlin '81]

- Maintain shortest path tree T_λ as λ increases *continuously* from 0.
- T_λ changes only at certain *critical values* of λ.
- *Pivot:* non-tree edge $p \to q$ replaces tree edge $p' \to q$.
- Stop when a pivot introduces a cycle into T_λ (dual of min cut)



Time analysis

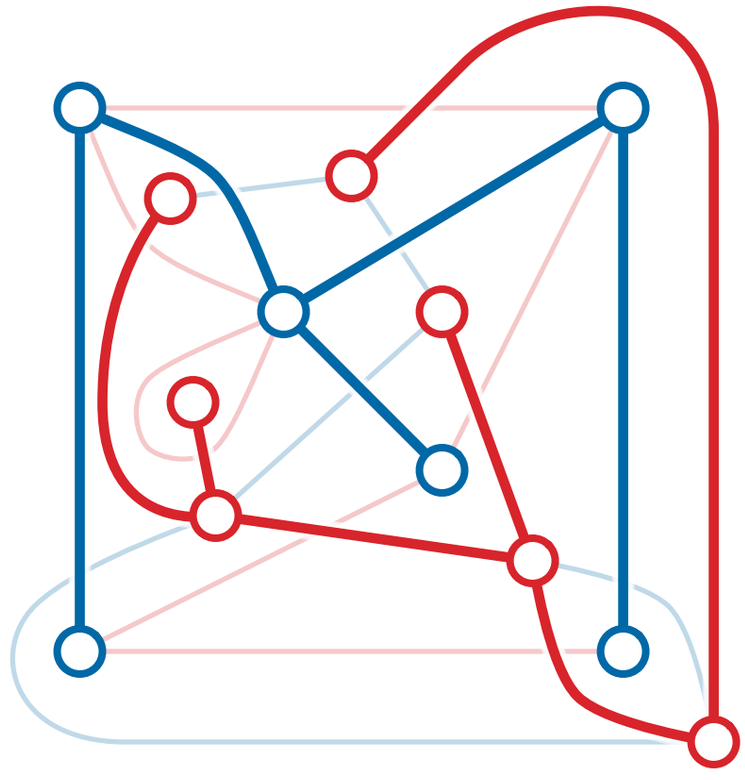
[Borradaile Klein '09] [E' 10]

- Each pivot can be executed in $O(\log n)$ time.
  - Proof uses cycle-cut duality
  - Using standard *dynamic forest* data structures
    [Sleator Tarjan '83] [Alstrup et al '05] [Tarjan Werneck '07]
- The algorithm halts after $O(n)$ pivots.
  - Proof uses a *covering space* argument and cycle-cut duality
- Thus, the overall running time is $O(n \log n)$.

Interdigitating trees

[Von Staudt 1847]

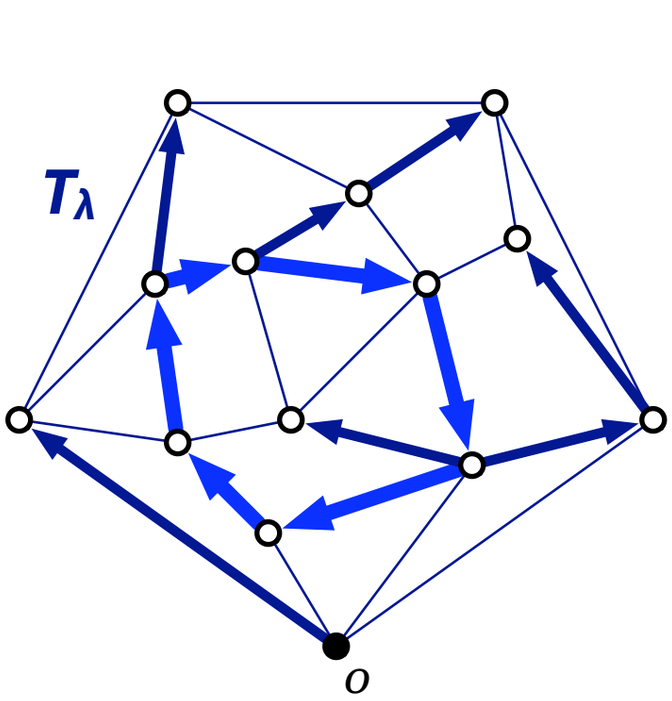
- Let T be an arbitrary spanning tree of a planar graph G.
- Then $L^* = (G \setminus T)^*$ is a spanning tree of G^*.
  - T is connected ⇒ L^* is acyclic
  - T is acyclic ⇒ L^* is connected

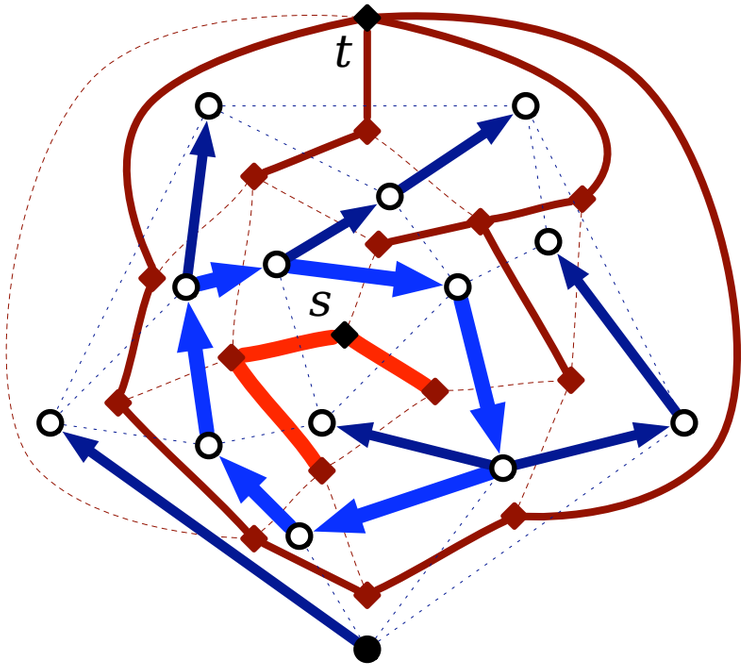


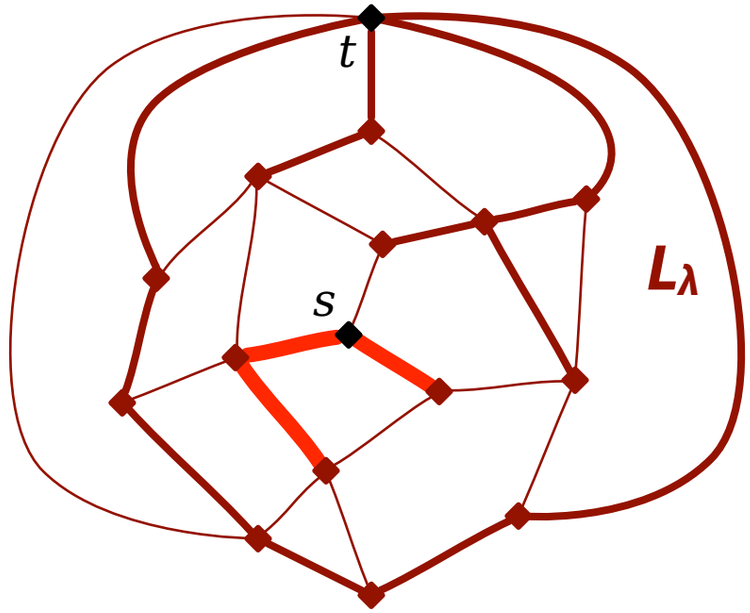
Which edges can pivot?

- LP_λ := unique path from s to t in the spanning tree $L_\lambda = G \setminus T_\lambda^*$

e^* can pivot $\iff$ e is in LP_λ

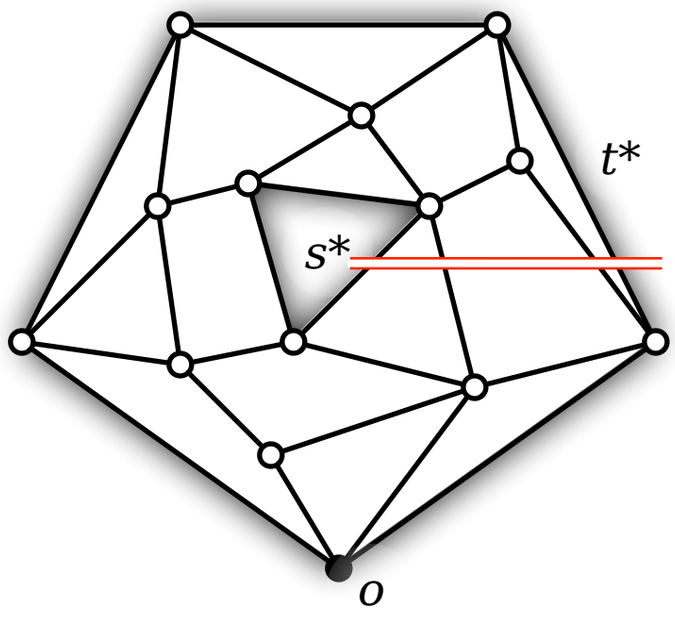






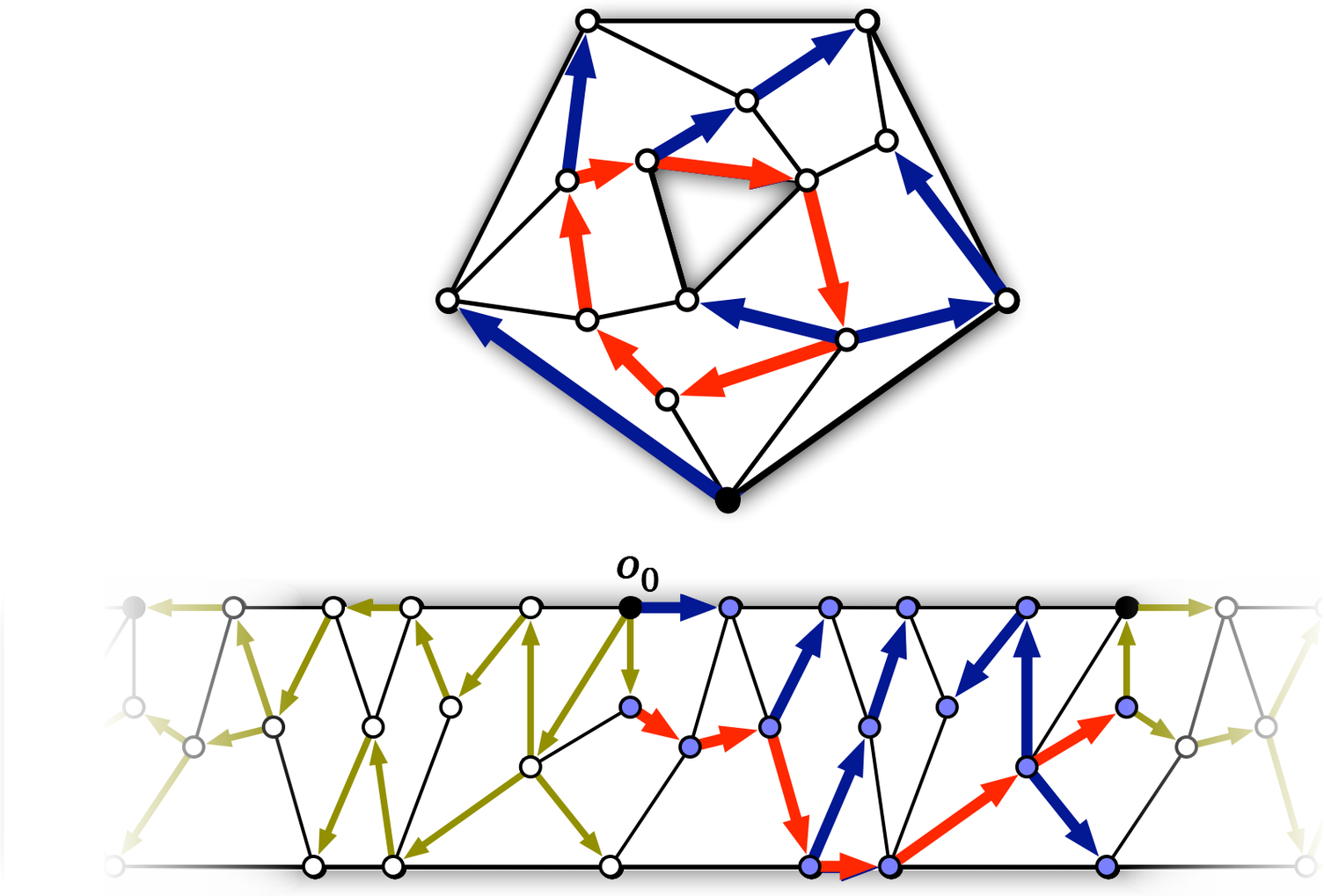
Universal cover $\overline{G^*}$

- Unroll the annulus carrying G^* into an infinite strip.



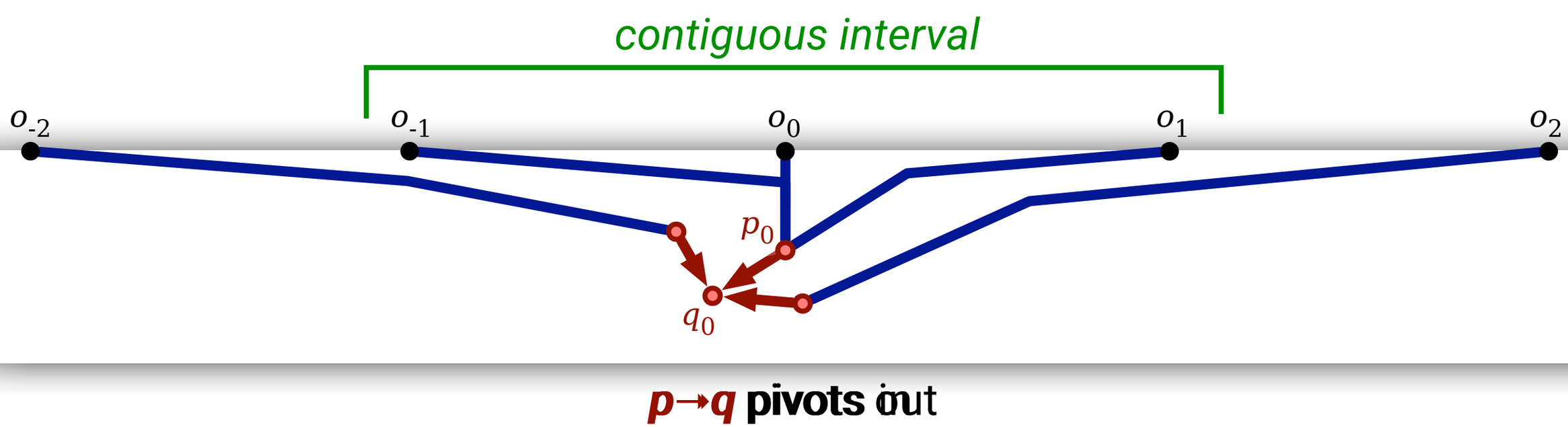
Pivots in $\overline{G}^*$

- Every shortest path in $G_\lambda{}^*$ lifts to a shortest path in $\overline{G}^*$



Counting pivots

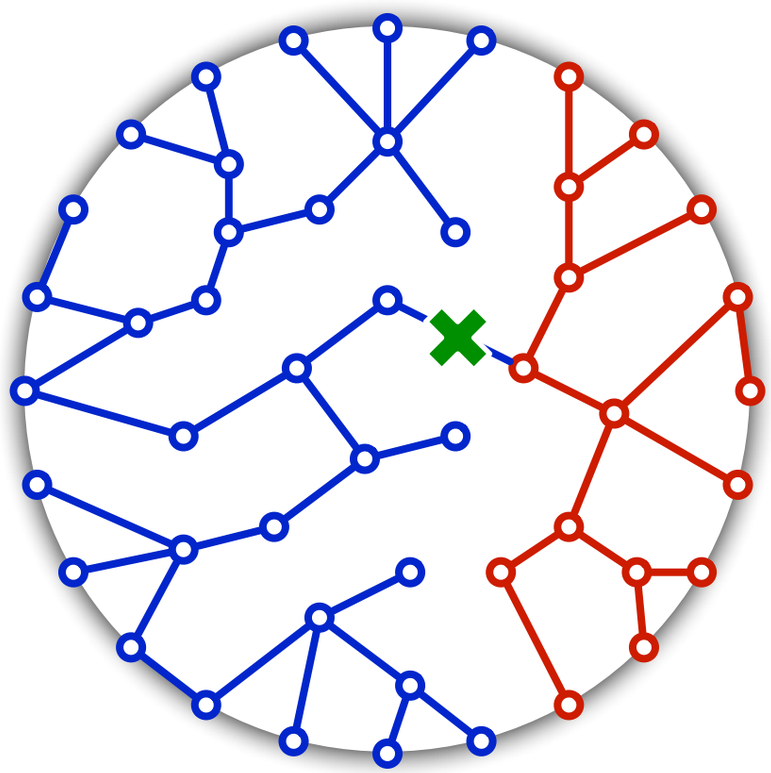
- Consider any arc $p \to q$. Fix a lift of q.
- Whenever $path_\lambda(q)$ changes, lift of o moves one step *left*.



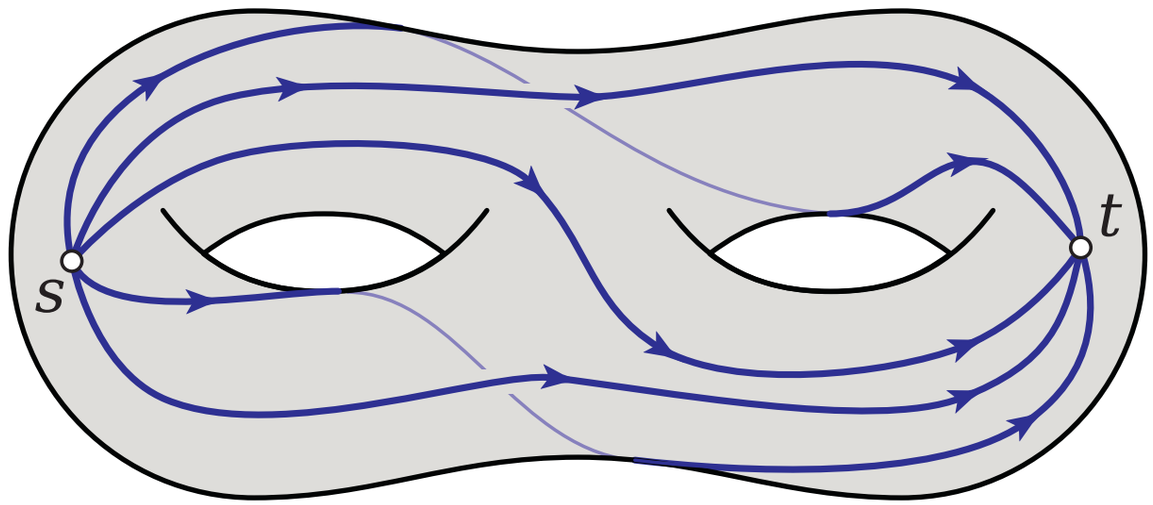
- $p \to q$ pivots in *at most once* and out *at most once*.

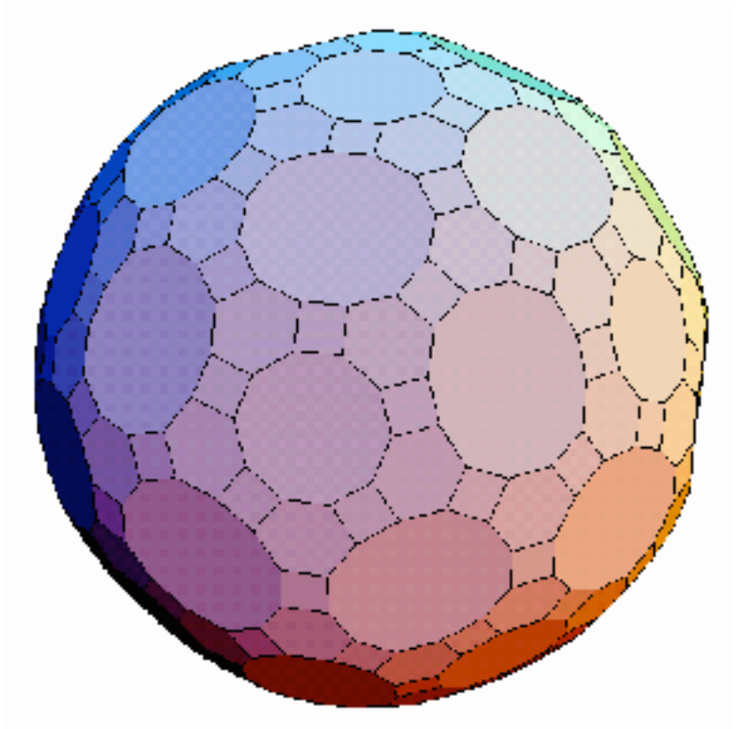
The disk-tree lemma

- Let T be any tree embedded on a closed disk. Vertices of T subdivide the boundary of the disk into intervals.
- Deleting any edge splits T into two subtrees R and B.
- At most two intervals with one end in R and the other in B.



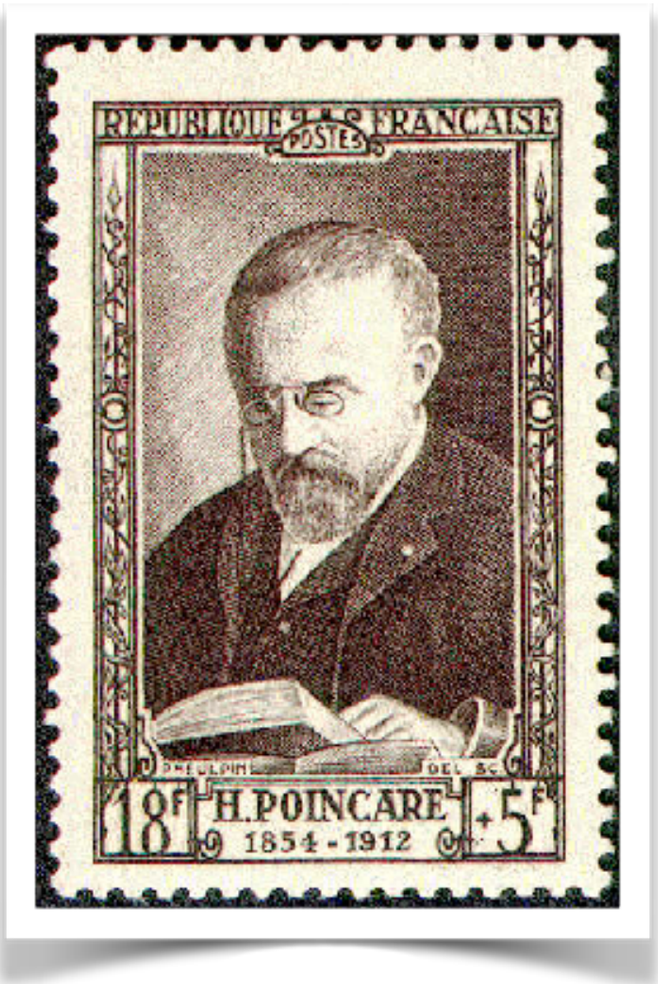
Maximum flows in surface graphs





Flows in surface graphs

- We can’t use the planar algorithm, because the duality between flows and shortest paths is more complicated.
- Once again, we need *homology*.

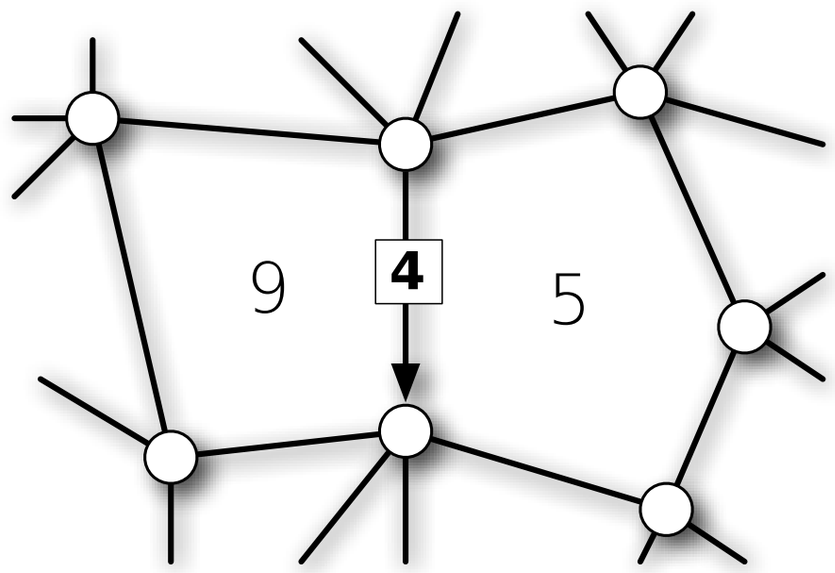


Boundary circulations

- A *boundary circulation* is a function $\partial\alpha : E \to \mathbf{R}$ of the form

$$\partial\alpha(u \to v) := \alpha(\text{right}(u \to v)) - \alpha(\text{left}(u \to v))$$

  for some function $\alpha : F \to \mathbf{R}$.



- α is an “Alexander numbering” of the circulation

Flow homology

- Two flows φ and ψ are *homologous* (or in the same *homology class*) if and only if their difference $\varphi - \psi$ is a boundary circulation.

- Two homologous flows always have the same value.

- Flow homology classes define a real vector space

$$H_1(\Sigma, \{s, t\}; \mathbb{R}) \cong \mathbb{R}^{2g+1}$$

[Chambers Erickson Nayyeri ‘09]

Some real vector spaces

vector space	notation	dimension
(s,t)-flows	Z_1	$\lvert E\rvert - \lvert V\rvert + \mathbf{2}$
boundary circulations	B_1	$\lvert F\rvert - \mathbf{1}$
homology classes of (s,t)-flows	$H_1 = Z_1/B_1$	$\lvert E\rvert - \lvert V\rvert - \lvert F\rvert + 3 = \mathbf{2g+1}$

Euler's formula: $\lvert V\rvert - \lvert E\rvert + \lvert F\rvert = 2 - 2g$

Feasible homologous flows

Let f be an arbitrary flow in a
surface-embedded flow network G

There is a *feasible* flow in G that is ***homologous with*** f
if and only if
the *dual residual network* G_f^* has no negative cycles.

[Chambers Erickson Nayyeri '09]

Feasible homologous flow

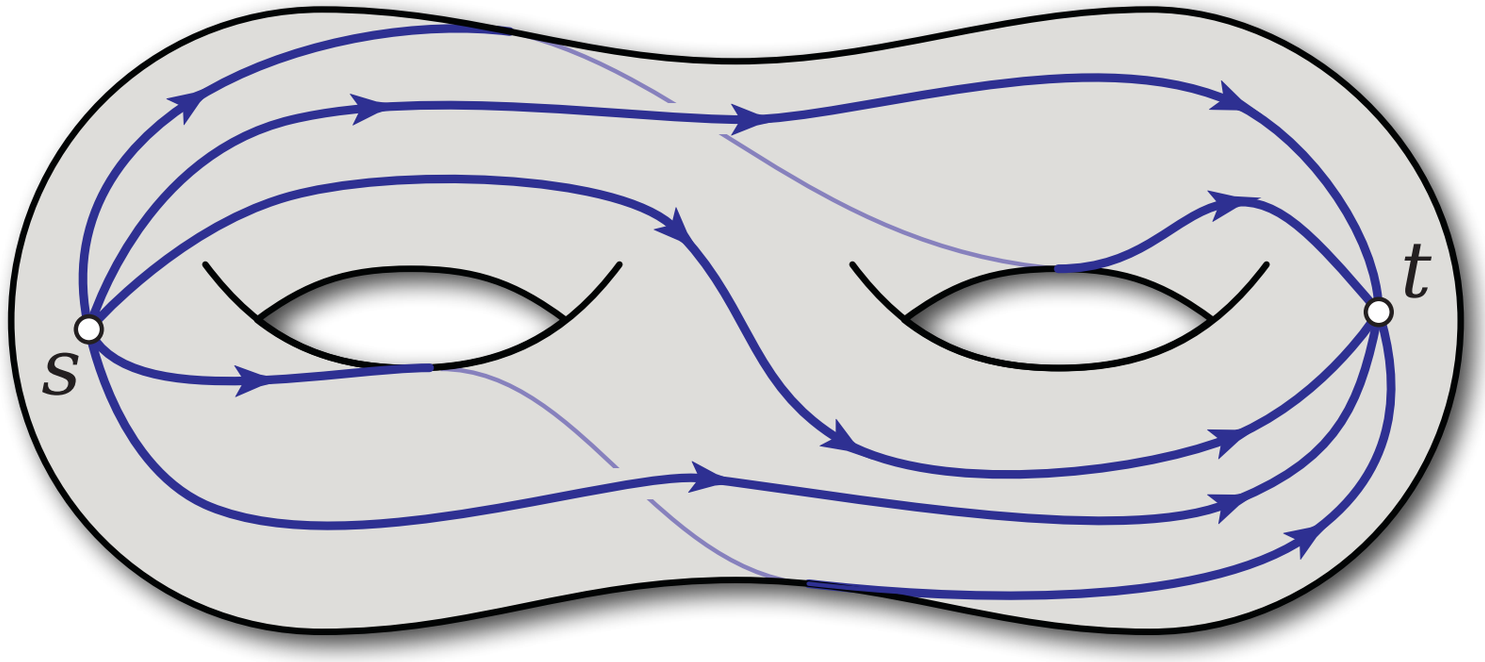
- **Lemma:** We can compute either a shortest path tree or a negative cycle in G_f^* in $O(g^2 n \log^2 n)$ time.

- Given an arbitrary flow f, we can compute either a flow homologous with f, or a *cocycle* over-saturated by f, in $O(g^2 n \log^2 n)$ time.

[Klein Mozes Weimann '09] [Chambers Erickson Nayyeri '09]

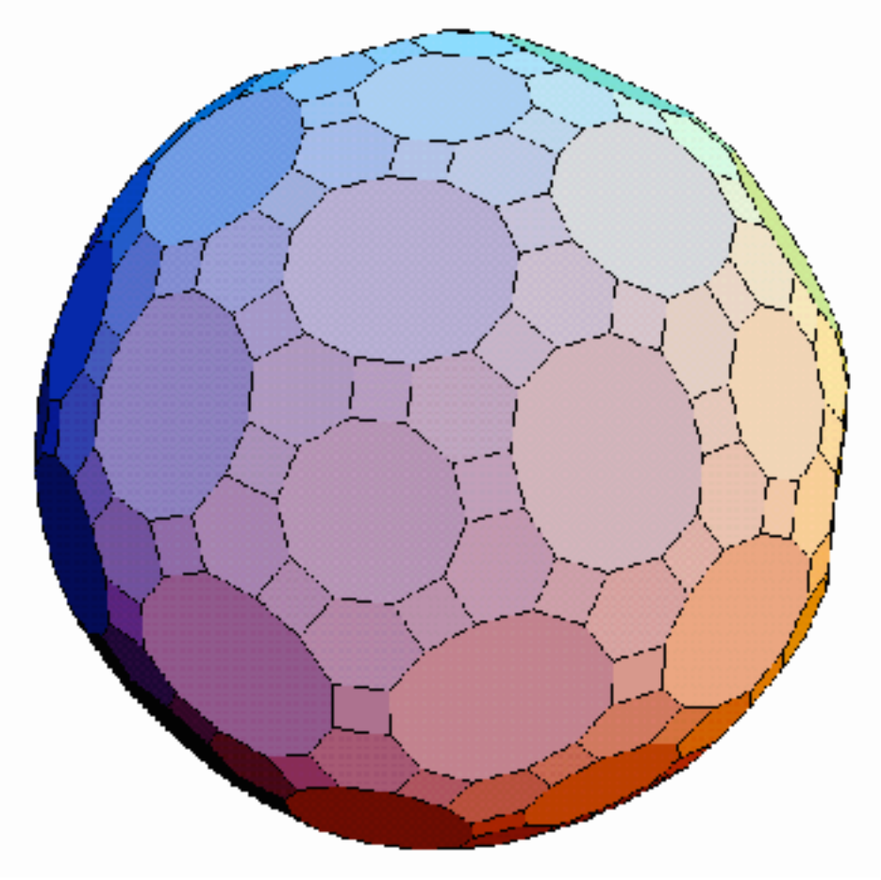
Flow homology basis

- Let $\pi_1, \pi_2, \ldots, \pi_{2g+1}$ be (s,t)-paths that cut Σ into a disk
- Every (s,t)-flow is homologous to $\sum_i \varphi_i \pi_i$ for some *unique* vector $(\varphi_1, \varphi_2, \ldots, \varphi_{2g+1})$



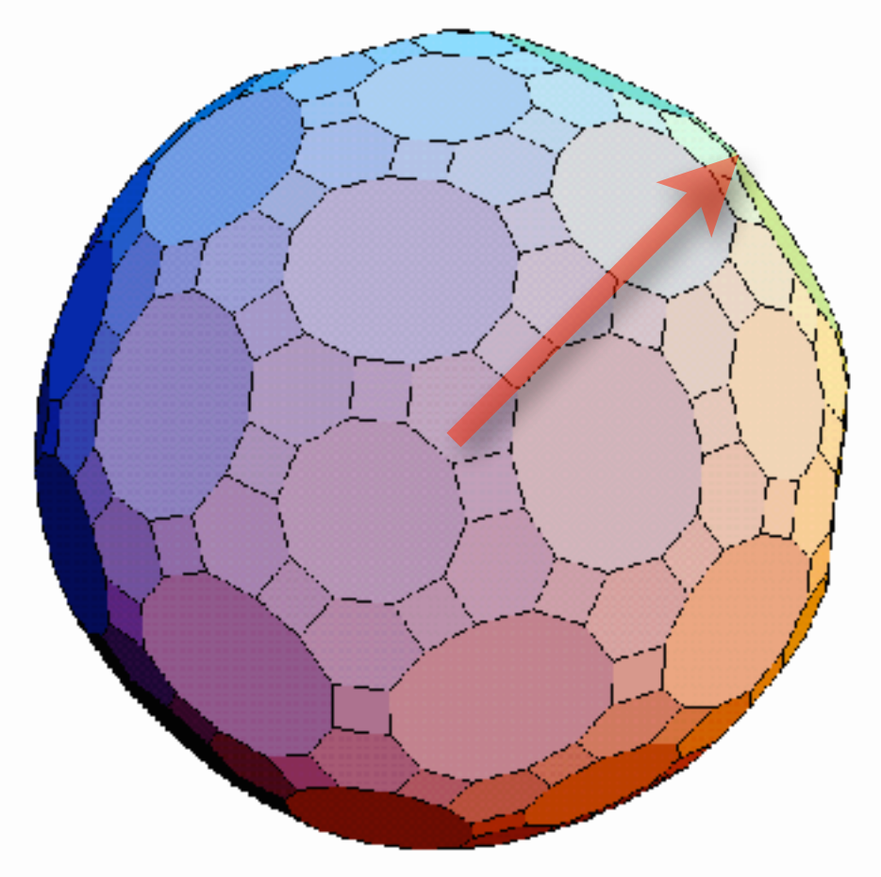
Flow homology polytope

- Set of feasible homology vectors = *convex polytope* in $\mathbf{R}^{2g+1}$
  - = Projection of the flow polytope into the homology subspace



Flow homology linear program

- Find a feasible homology vector $(\varphi_1, \varphi_2, \ldots, \varphi_{2g+1})$ that maximizes the flow value $\varphi_1 + \varphi_2 + \cdots + \varphi_{2g+1}$
- $2g+1$ variables, but *exponentially* many constraints



Implicit linear programming

- We can solve this LP *implicitly* using two oracles:
  - *Membership*: Is $(\varphi_1, \varphi_2, \ldots, \varphi_{2g+1})$ feasible?
  - *Separation*: If not, find a violated constraint.
- We have an oracle that runs in $O(g^2 n \log^2 n)$ time!
- If all capacities are integers $\leq C$, we can compute maximum flows using implicit linear programming methods:
  - Central-cut ellipsoid $\Rightarrow O(g^8 n \log^2 n \log^2 C)$ time.
    [Shor Nemirovsky Yudin '72] [Khachiyan '79] [Grötschel Lovász Schrijver '81, '93]
  - Random walk sampling $\Rightarrow O(g^6 n \log^2 n \log C)$ expected time.
    [Bertsimas Vempala '04]

Open questions

- Can we compute maximum flows on bounded-genus surface graphs in $O(n \log n)$ time?
  - Open even for unit-capacity undirected graphs on the torus (g=1)!
- Can we compute minimum cuts or maximum flows in $O(\text{poly}(g)\ n\ \text{polylog}\ n)$ time?
  - Open even for unit-capacity undirected graphs on the torus (g=1)!
- Generalize to *minimum-cost* flows?
  - Current best is $O(n^2 \log n)$, even for undirected *planar* graphs!

Thank you!

“Blush” — prototype radiator design
[Thorunn Arnadottir 2007]