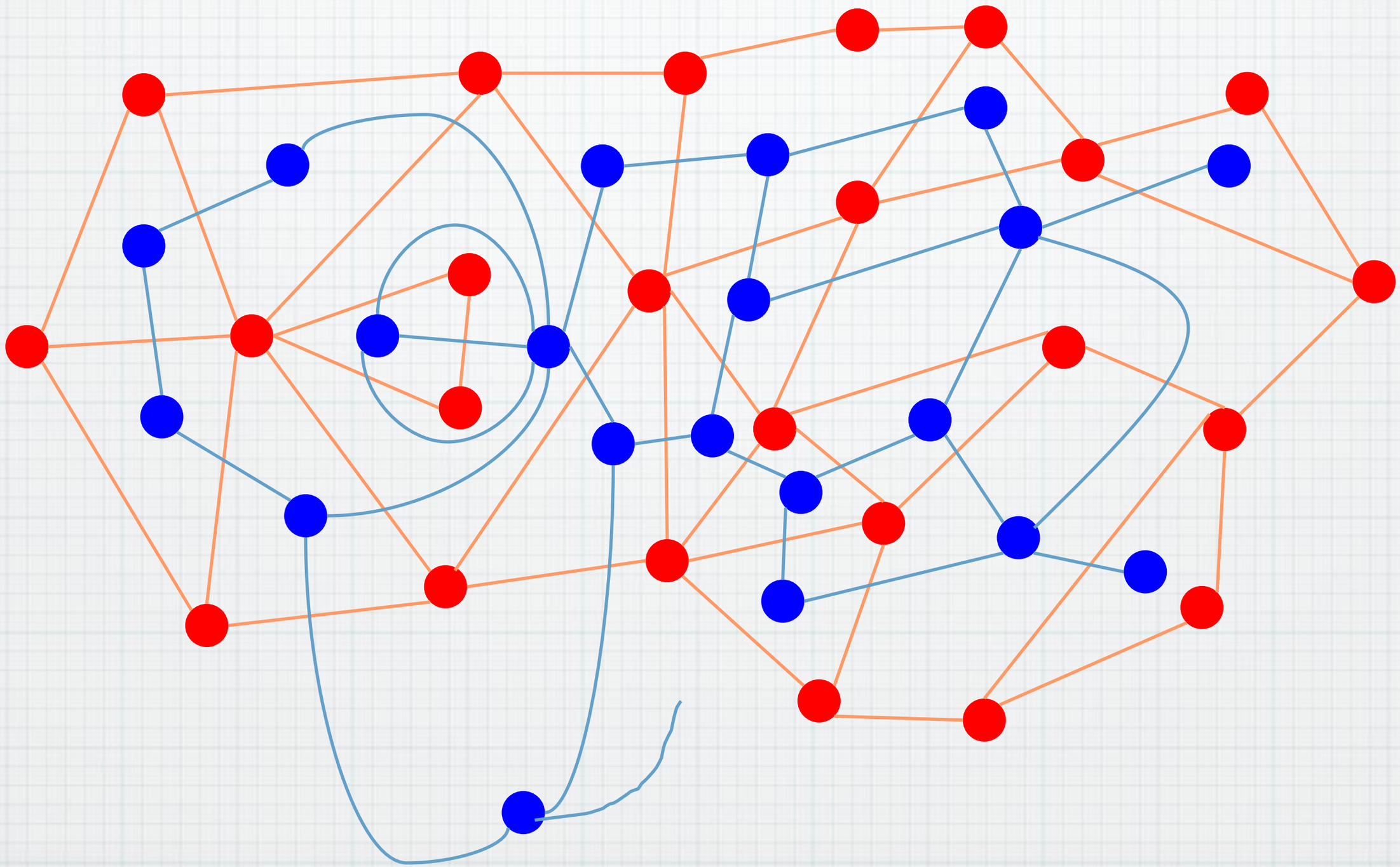
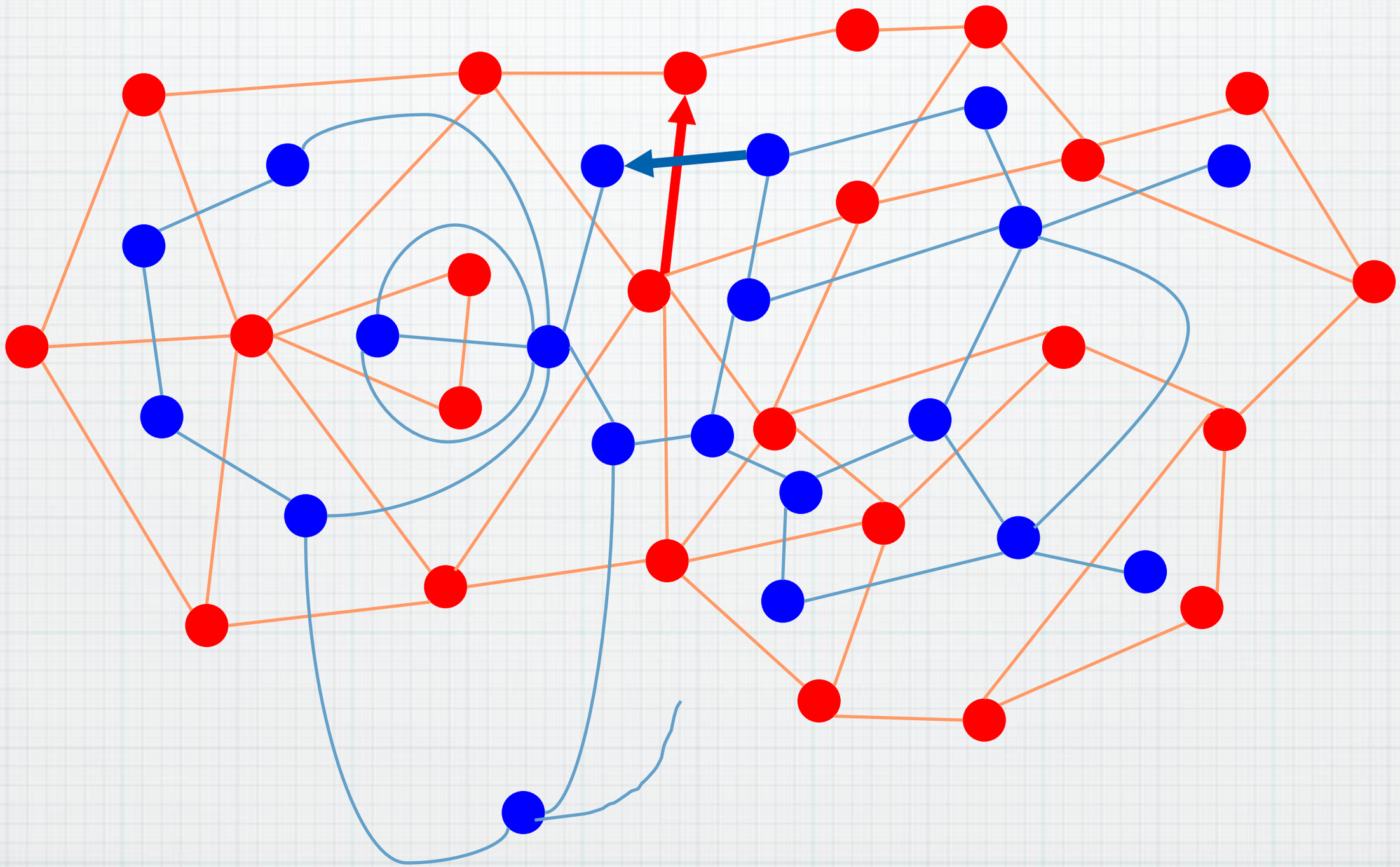
Maximum Flow in Planar Graphs

Planar Graph and it Dual

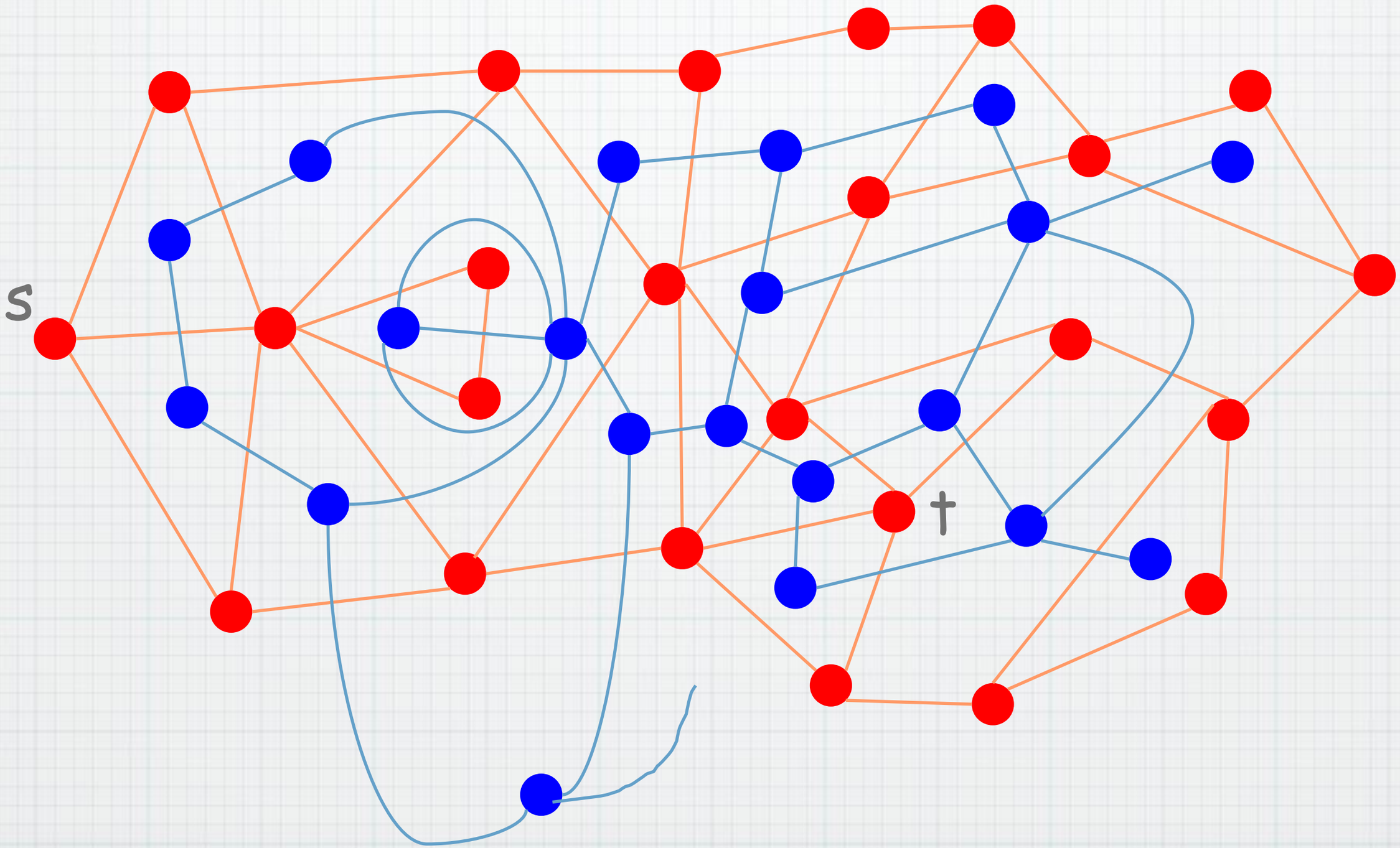


Duality is defined for directed planar graphs as well

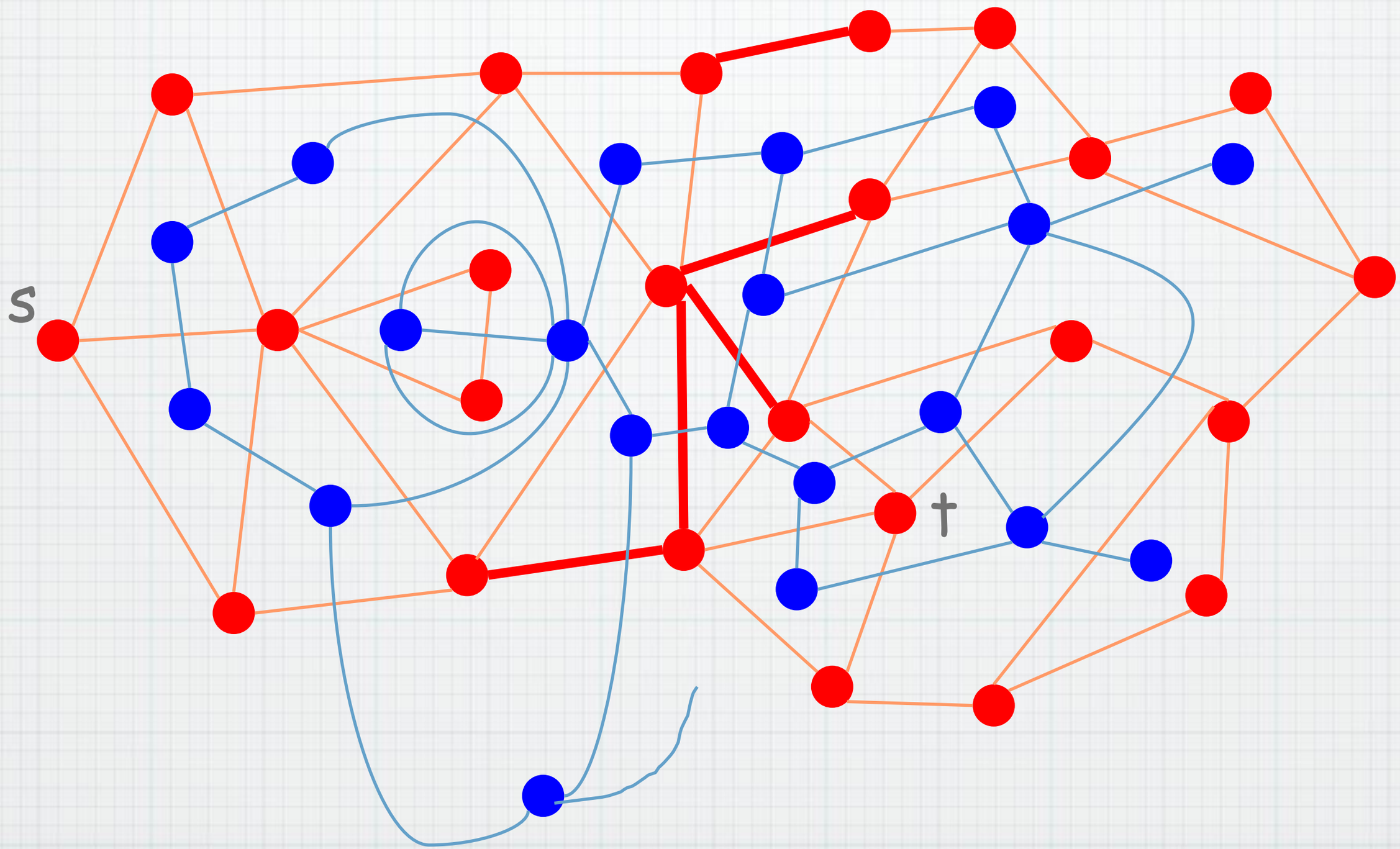


Minimum s-t cut in undirected planar graphs

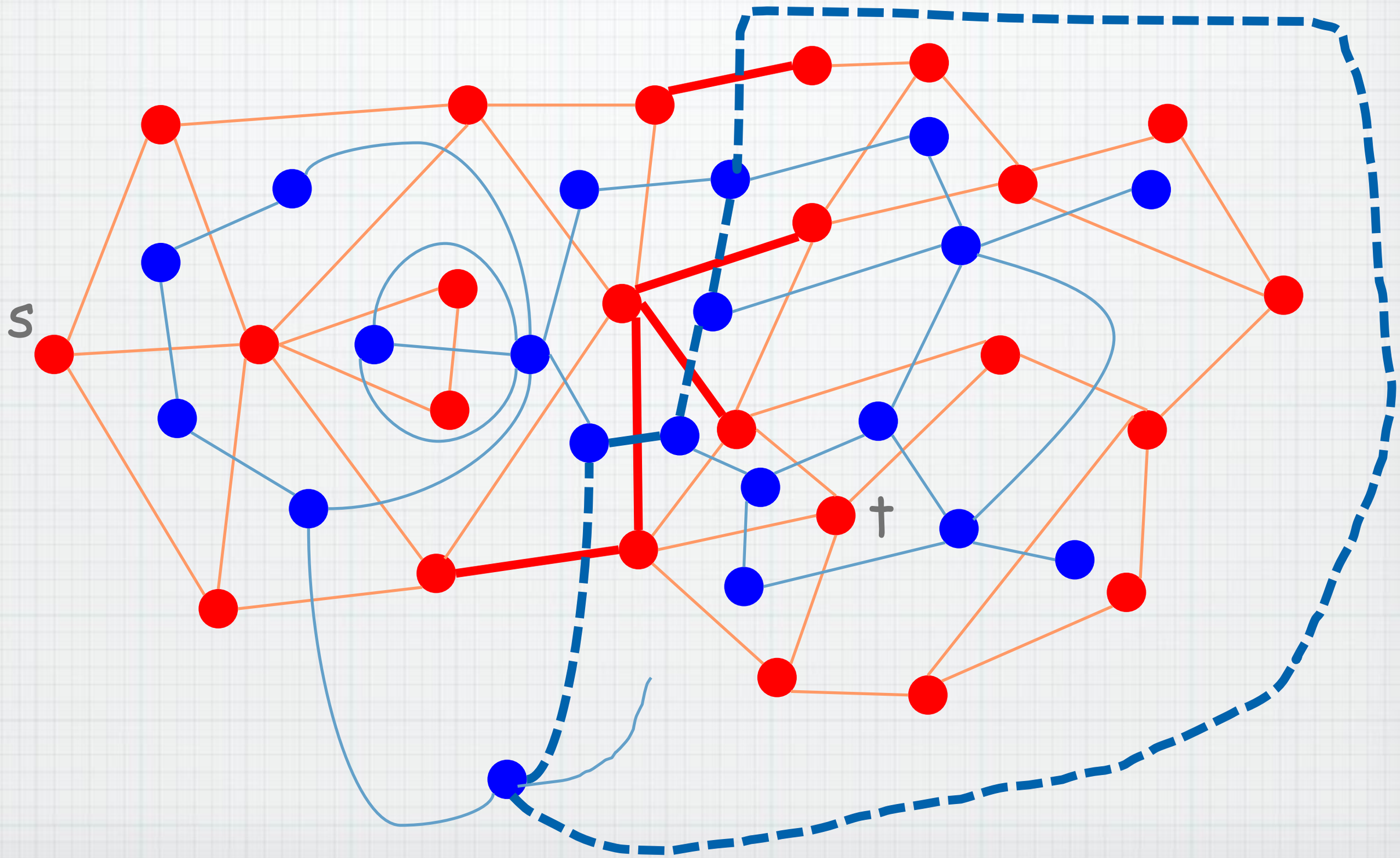
An s-t cut (undirected graphs)



An s-t cut



The dual to the cut



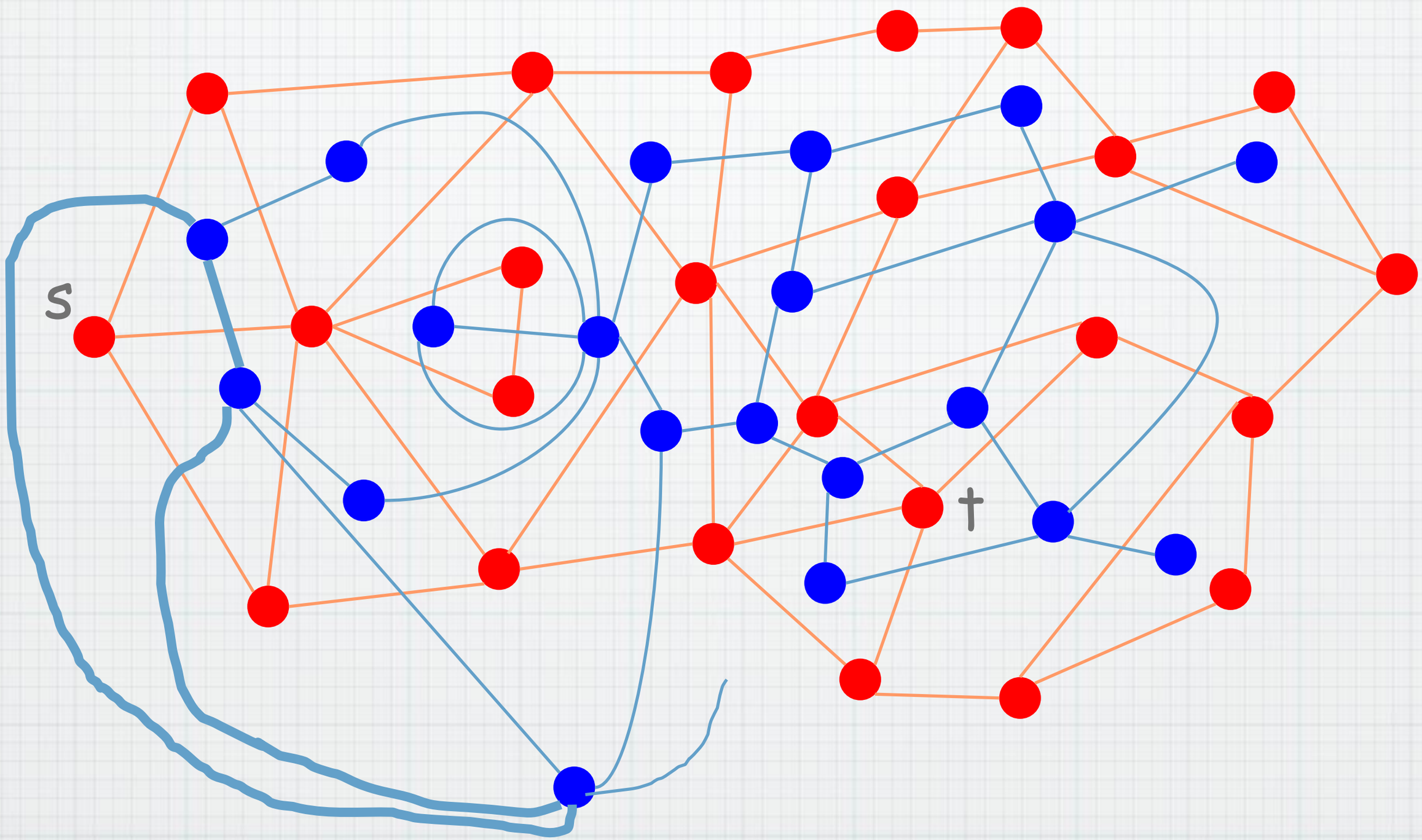
Cuts/Cycles

A cut that separates the graph into two connected components one containing s and one containing t (we can assume the min-cut is like this)

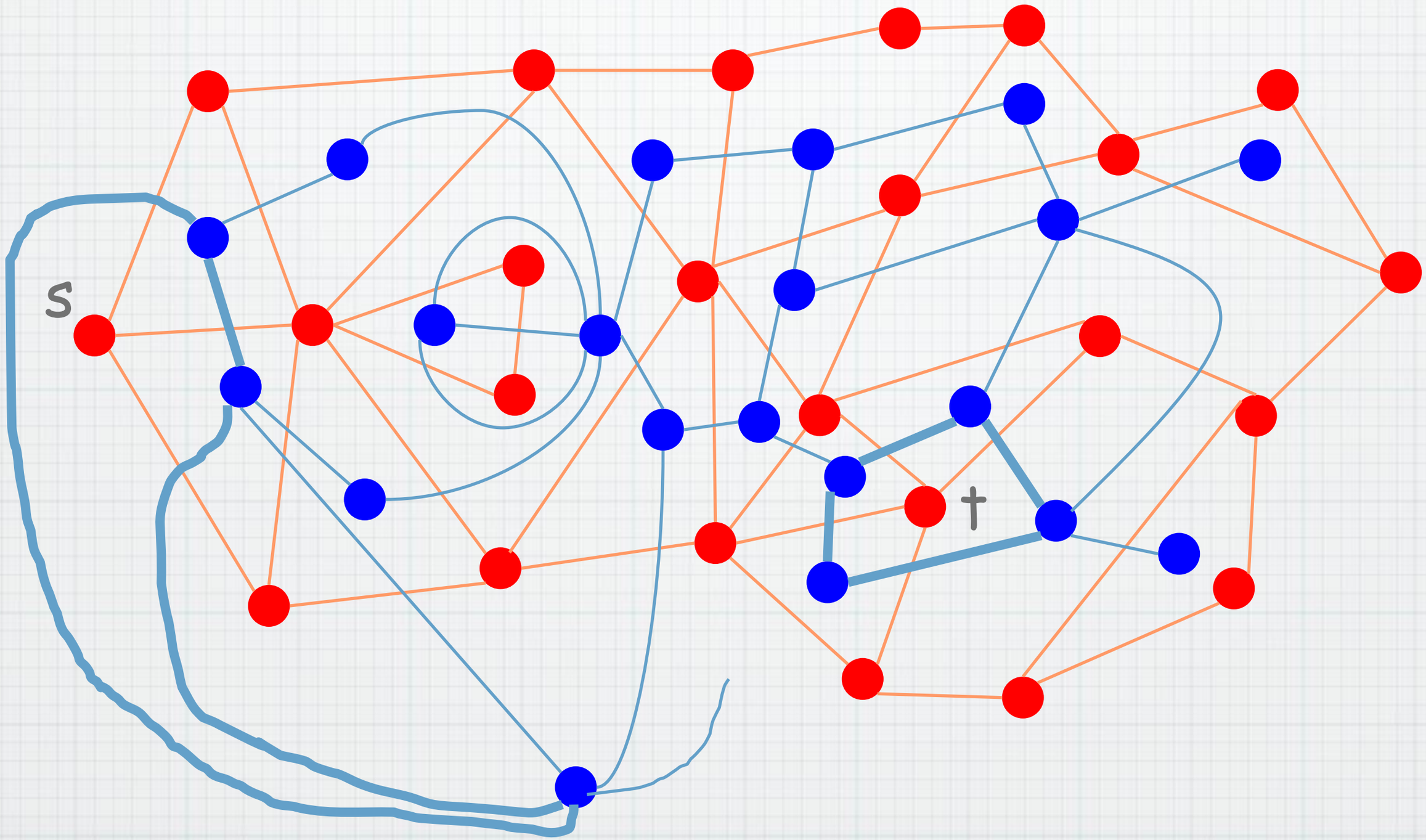
A simple cycle with t inside and s outside

➔ Look for a shortest such cycle in the dual (lengths in the dual are capacities in the primal)

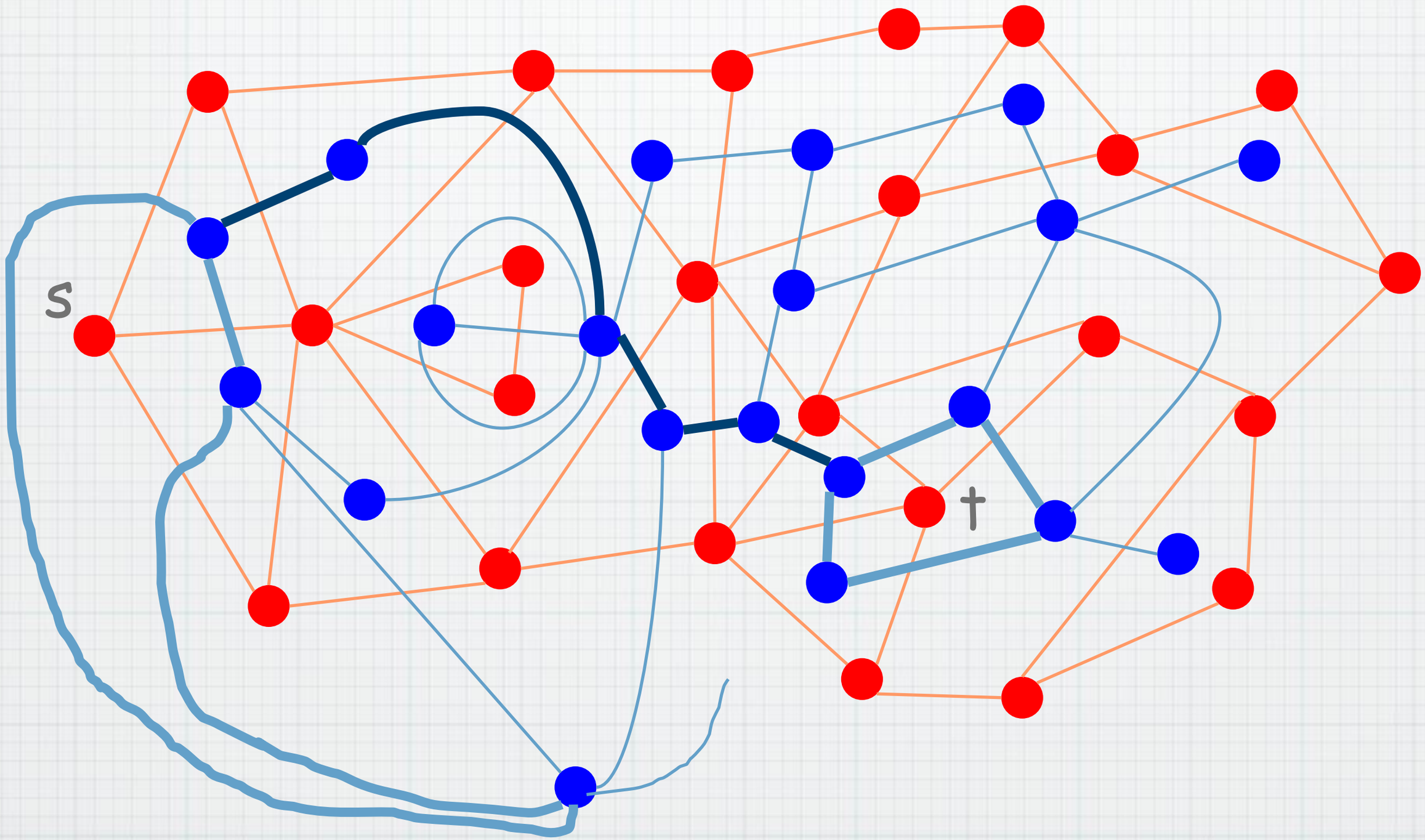
The face containing s



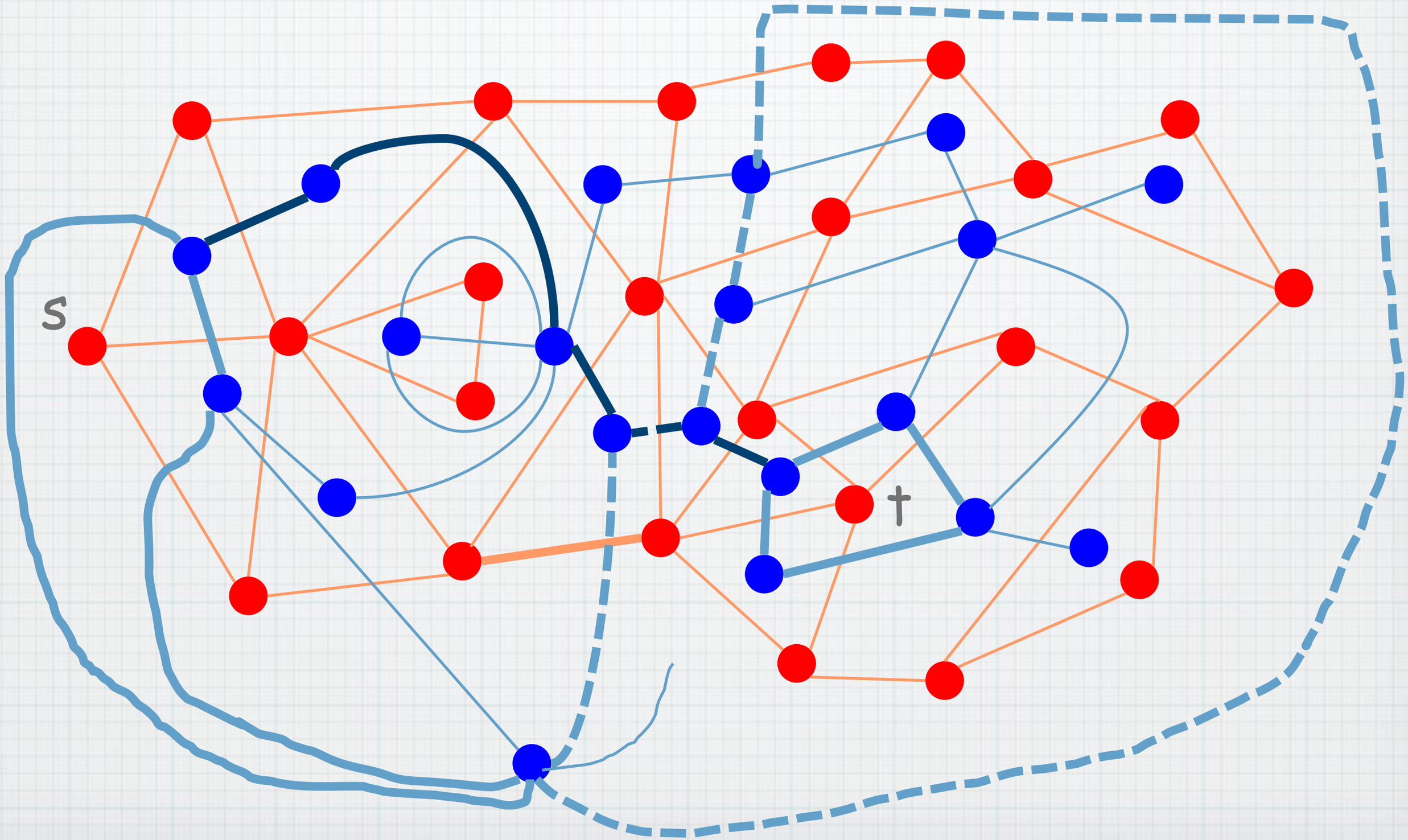
and the face containing t



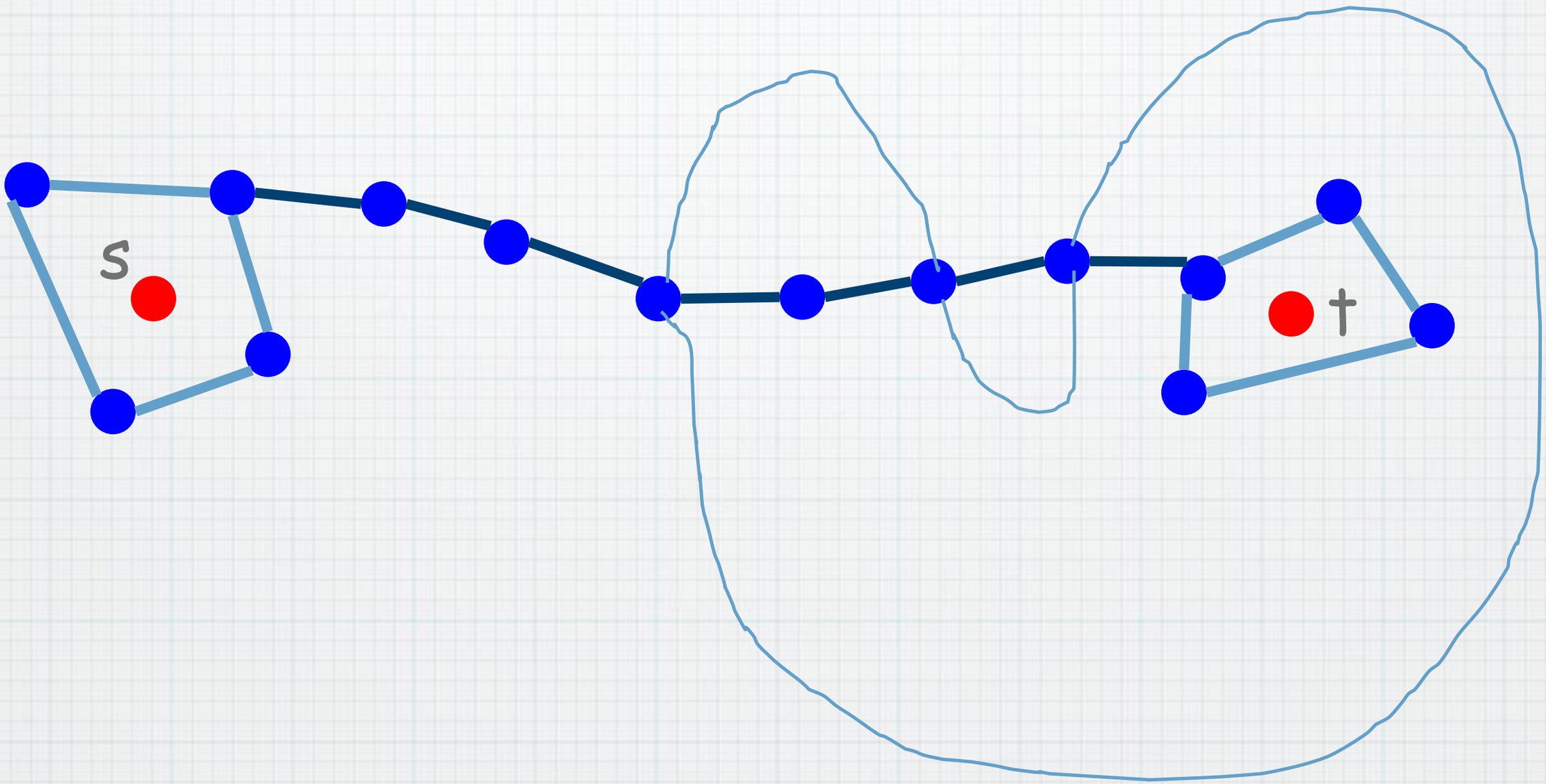
Let P be the shortest path between them



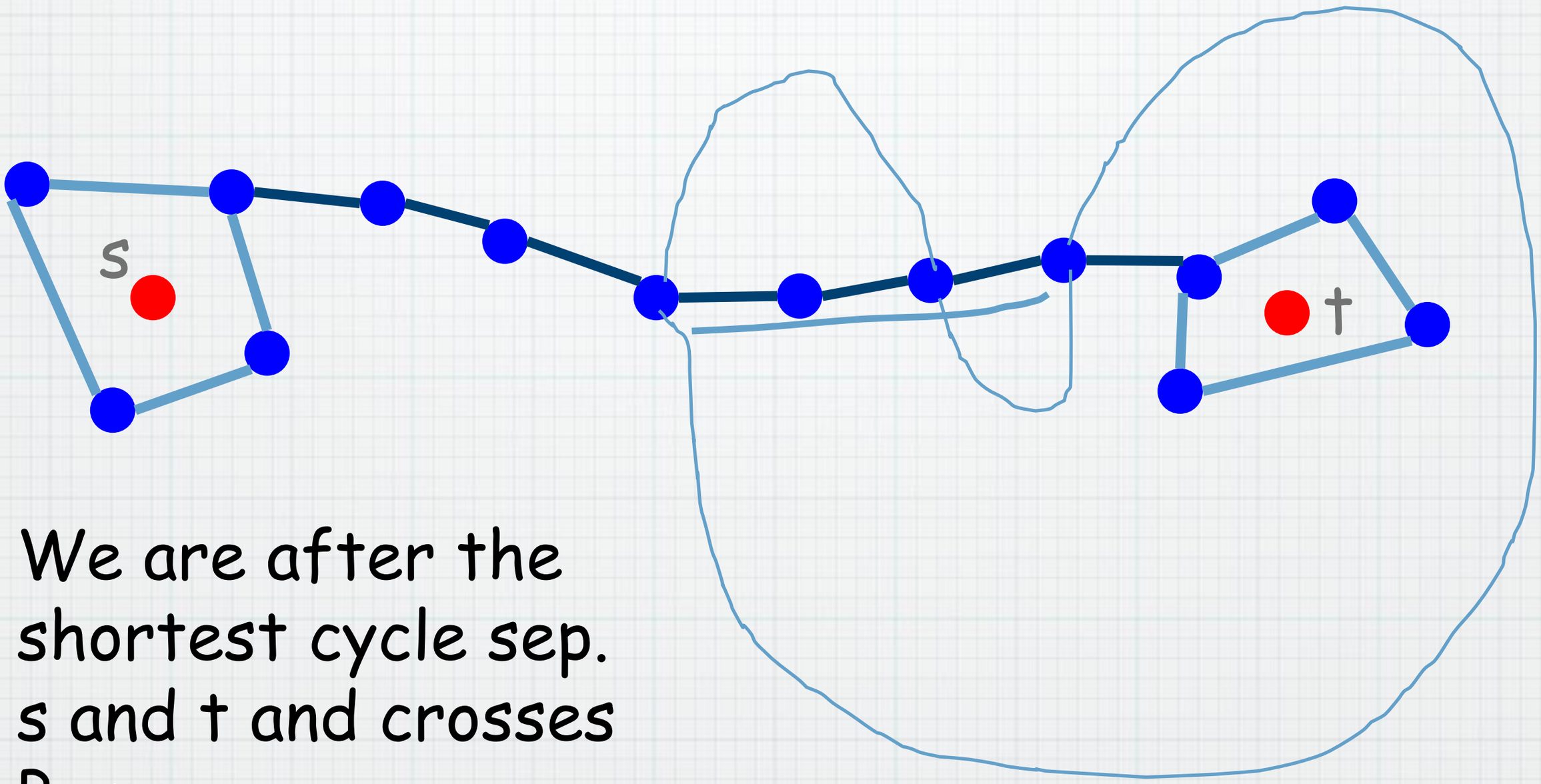
Any cycle sep. s and t crosses P



The shortest cycle will cross P once

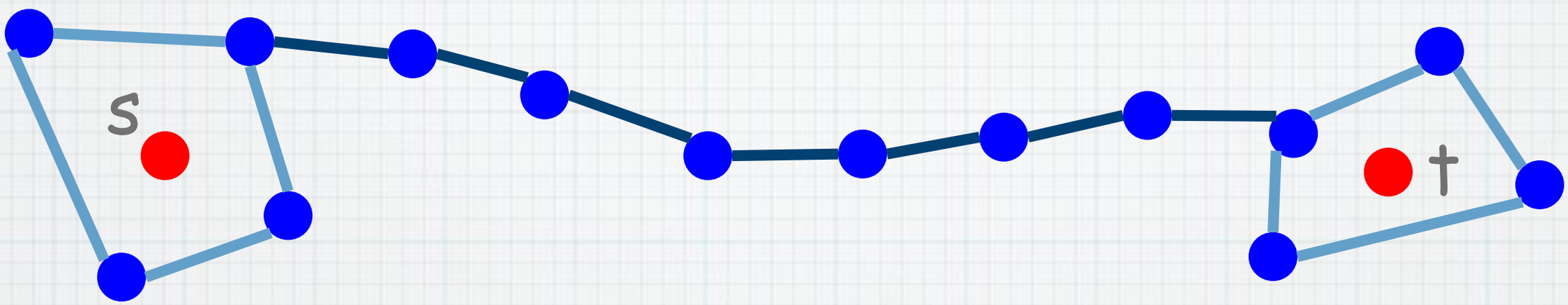


The shortest cycle will cross it once

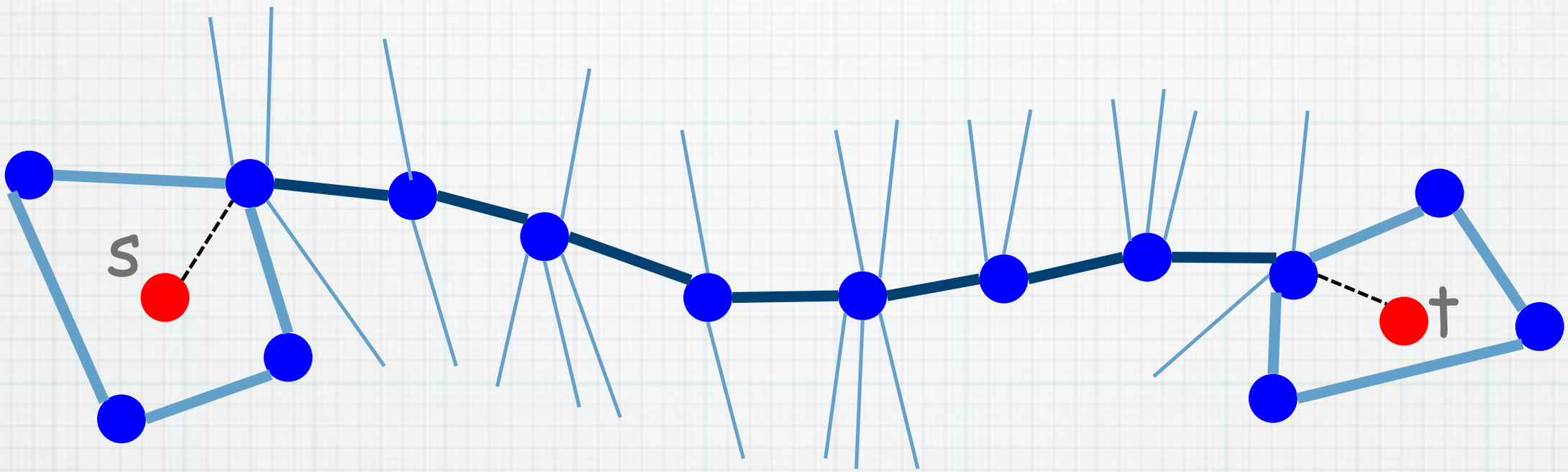


We are after the shortest cycle sep. s and t and crosses P once.

Finding such shortest cycle

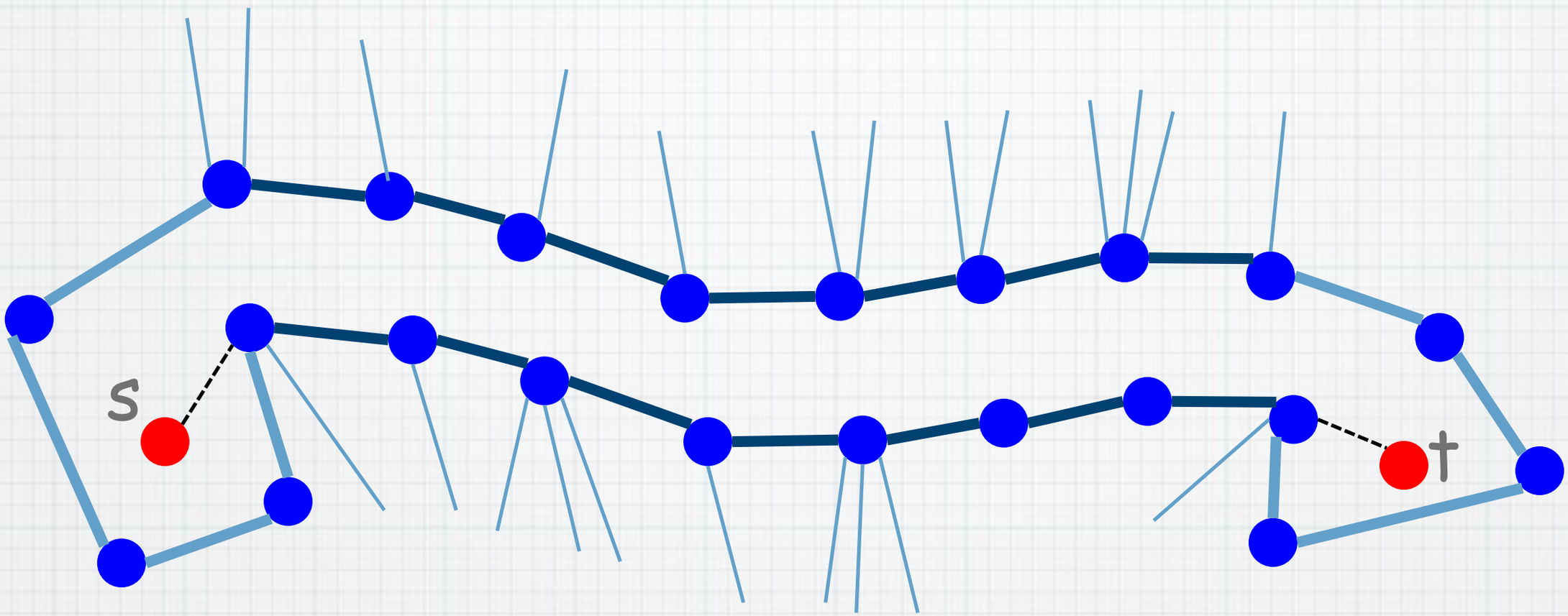


Finding such shortest cycle



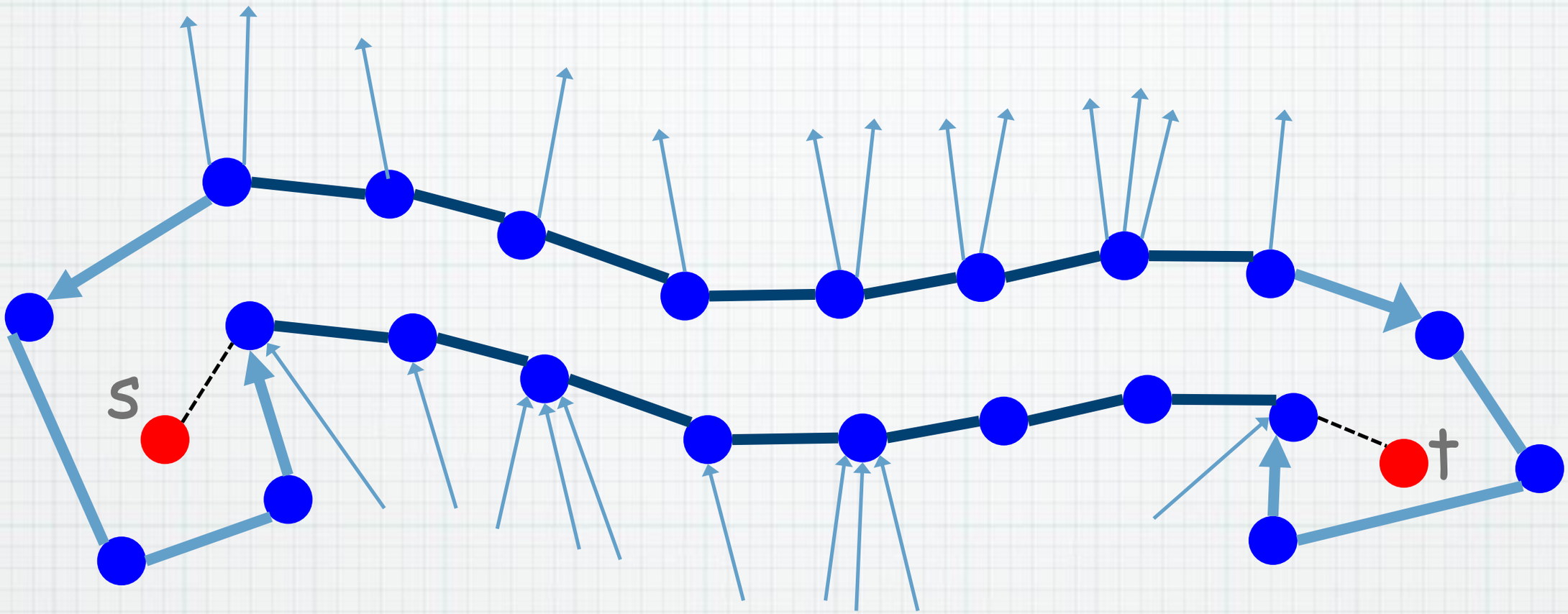
Classify edges incident to the path as
left or right

Finding such shortest cycle



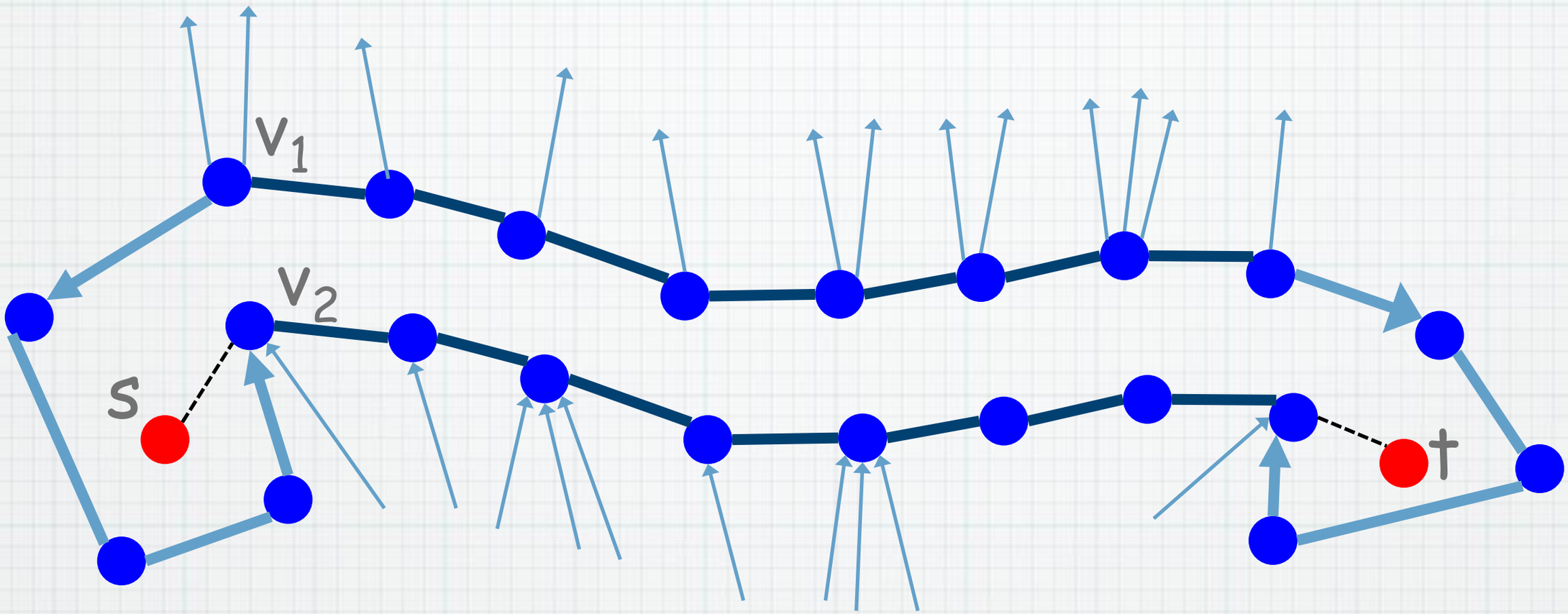
Cut the path open

Finding such shortest cycle



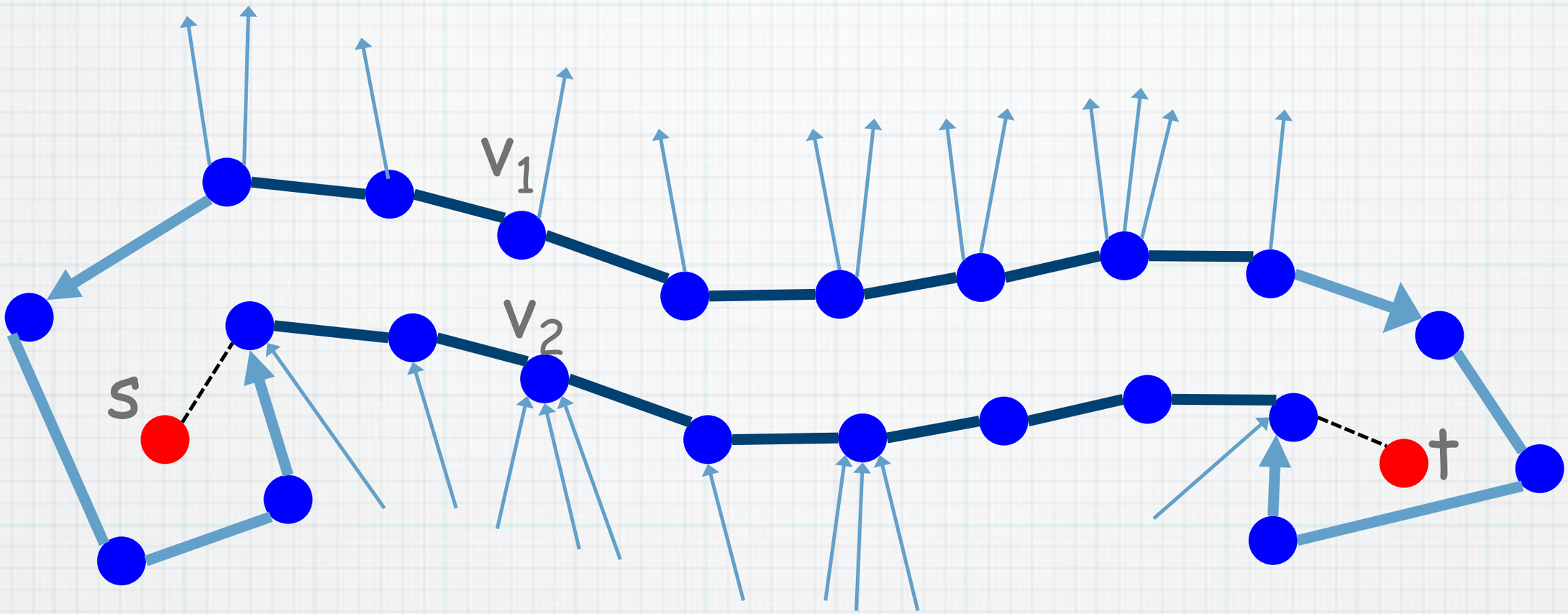
Direct the edges incident to the path

Finding such shortest cycle



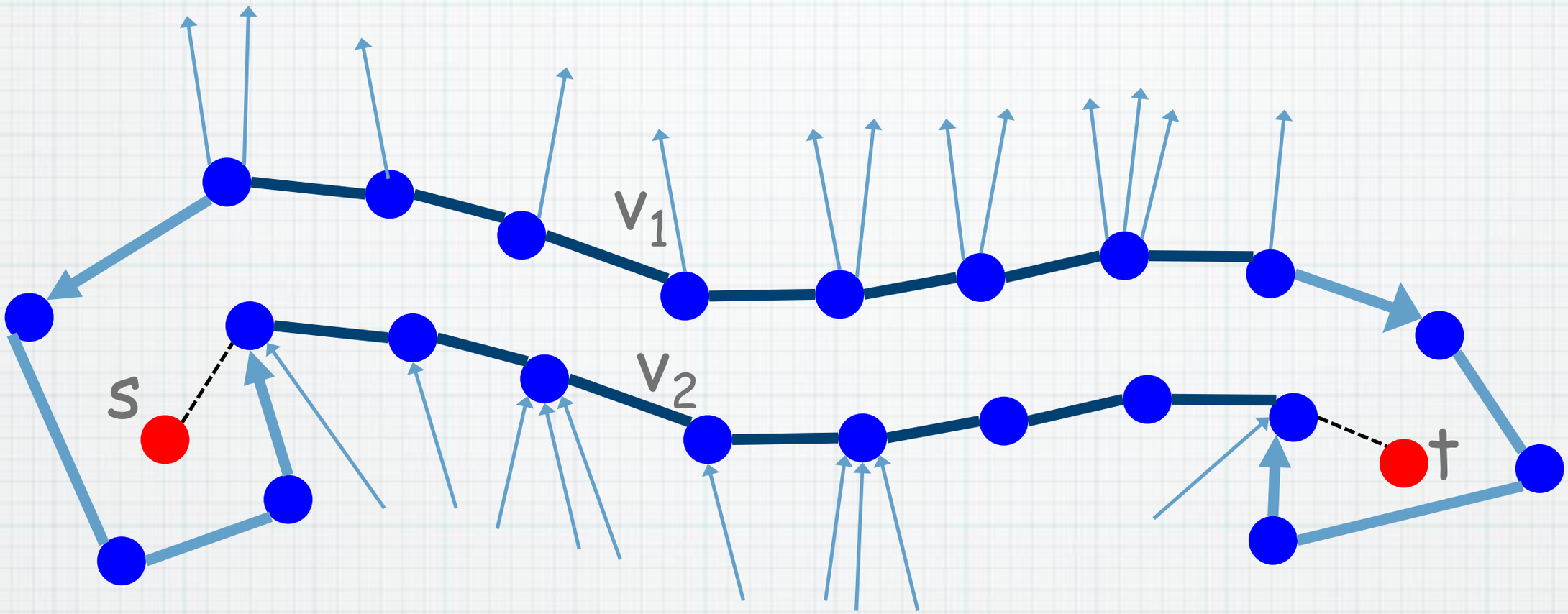
Find shortest path between every pair v_1, v_2

Finding such shortest cycle



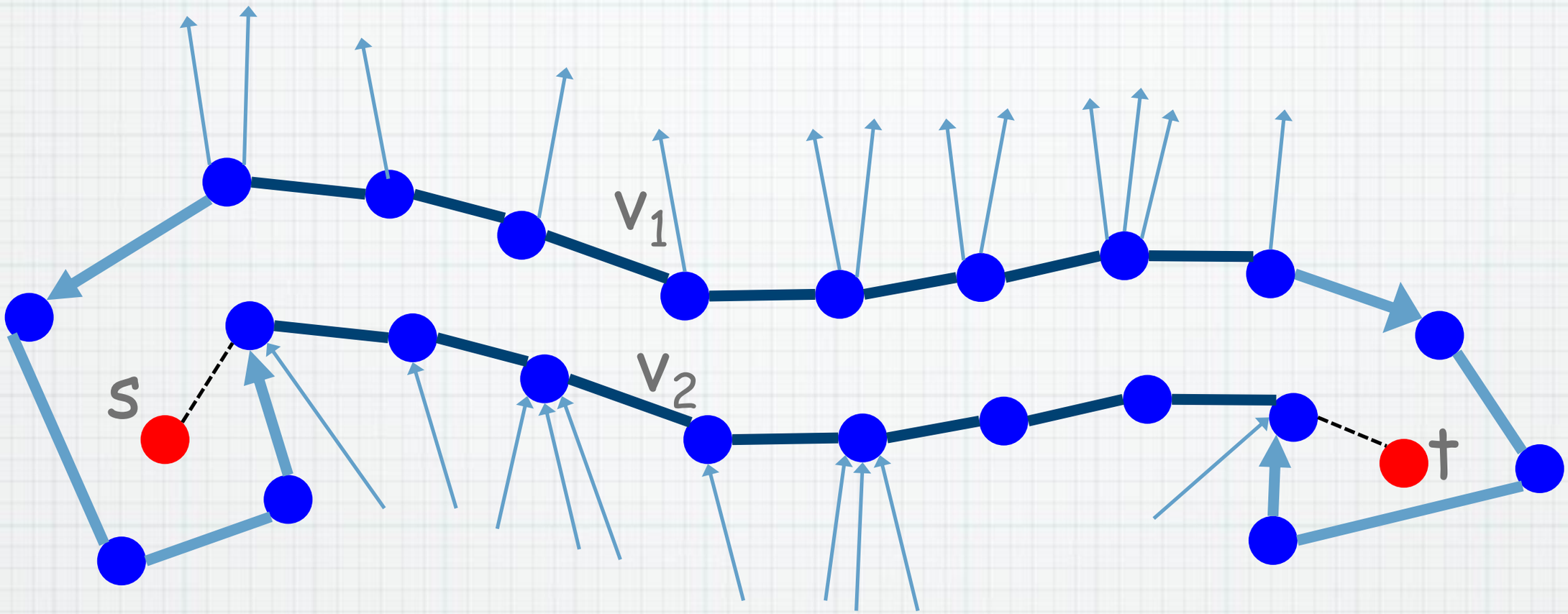
Find shortest path between every pair v_1, v_2

Finding such shortest cycle



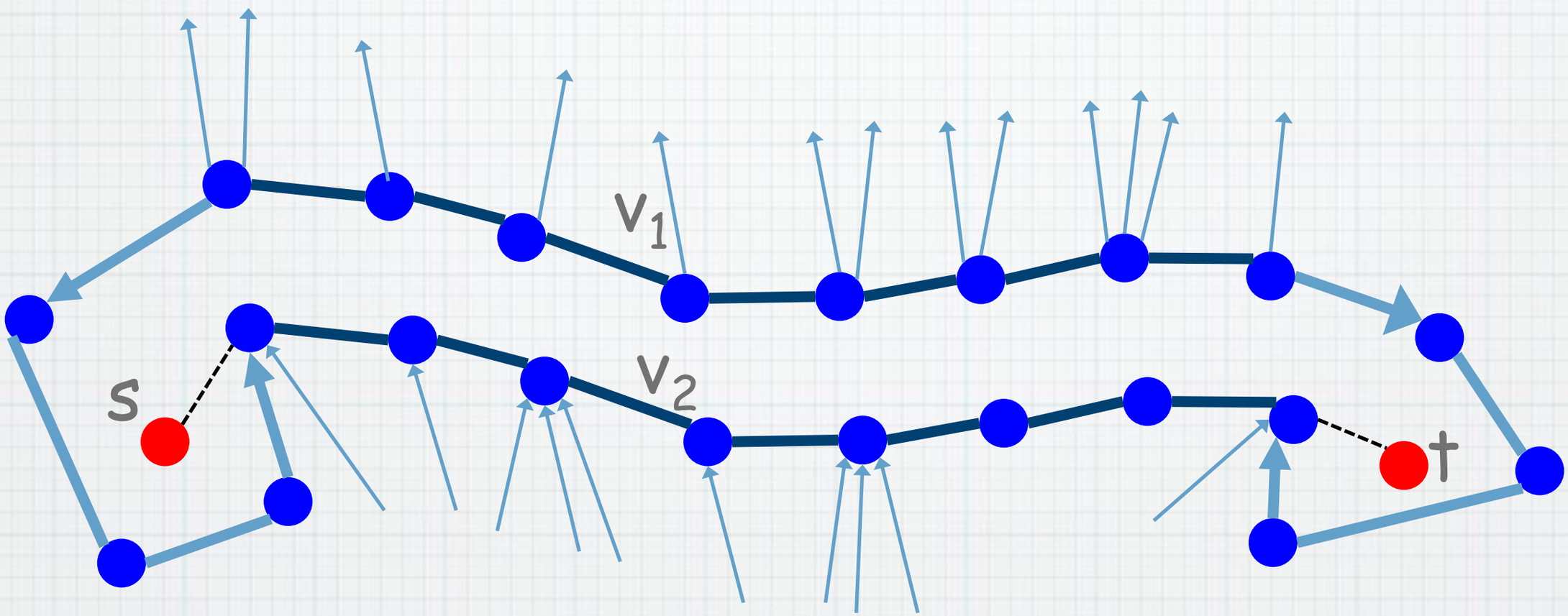
Find shortest path between every pair v_1, v_2

Finding such shortest cycle

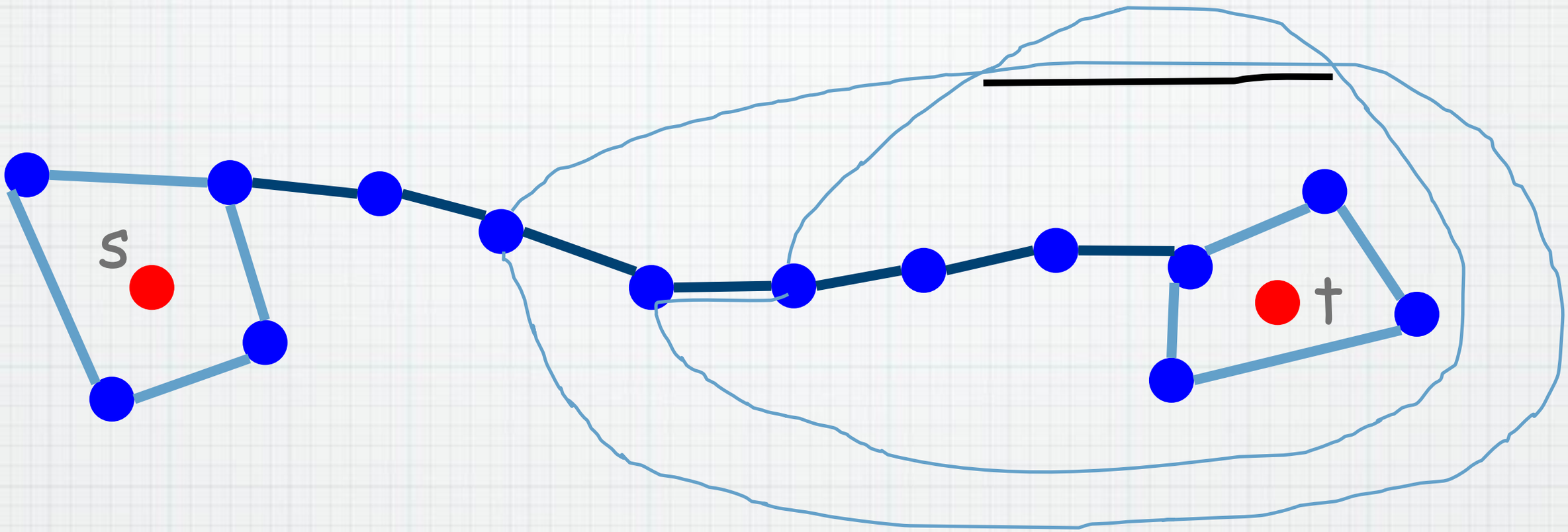


Take the shortest among these shortest paths

Speeding up by divide and conquer

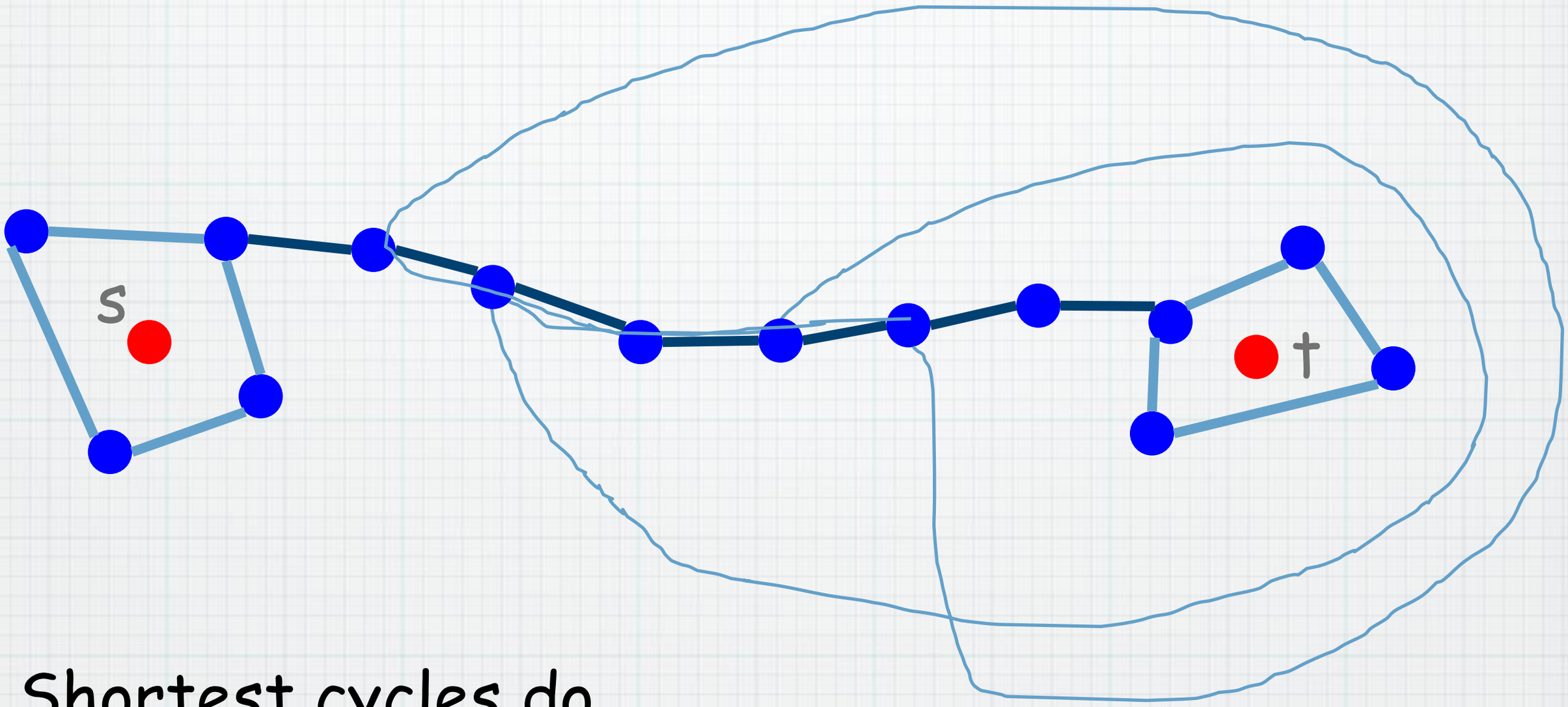


Speeding up by divide and conquer



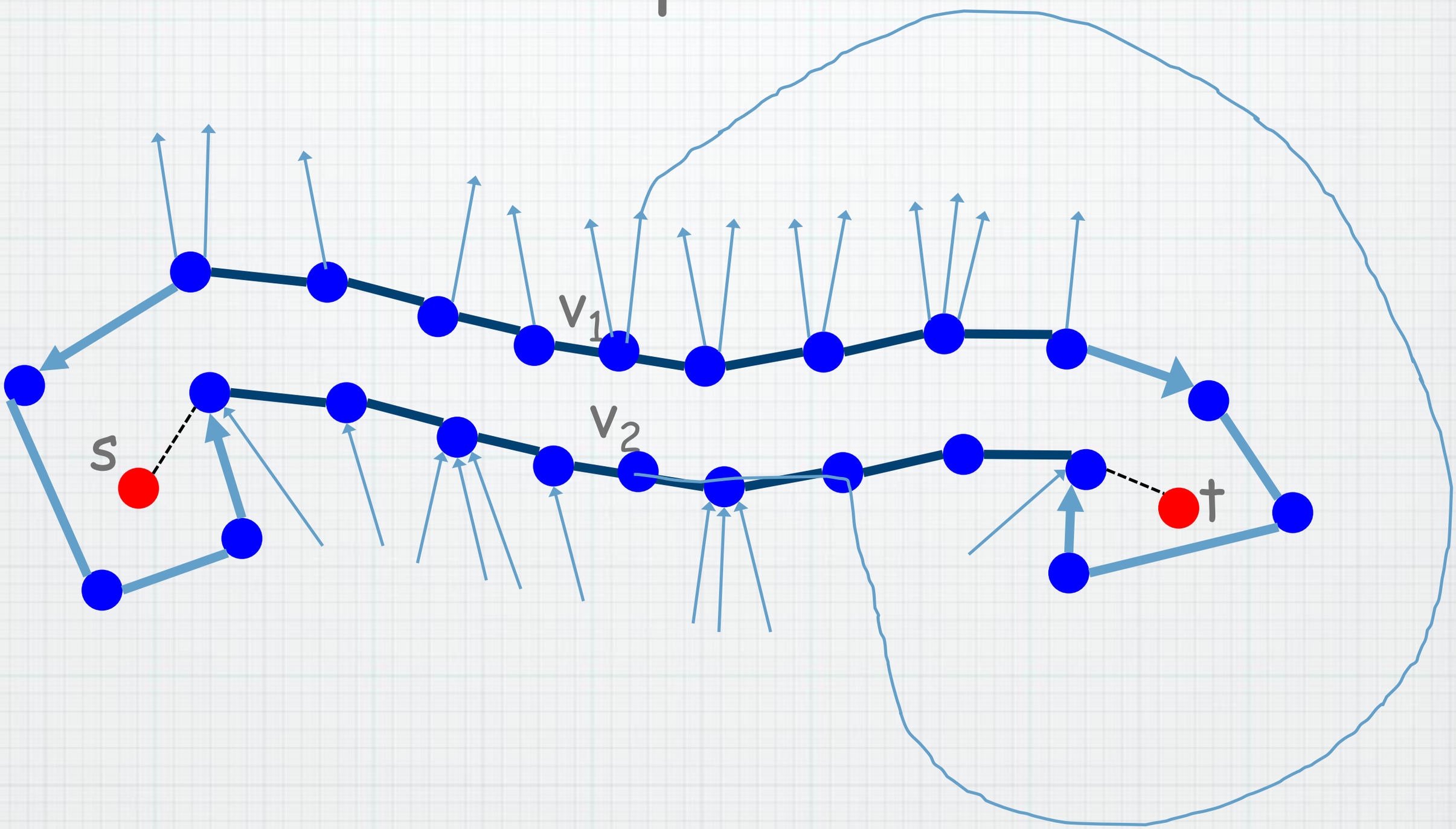
Shortest cycles do not cross

Speeding up by divide and conquer

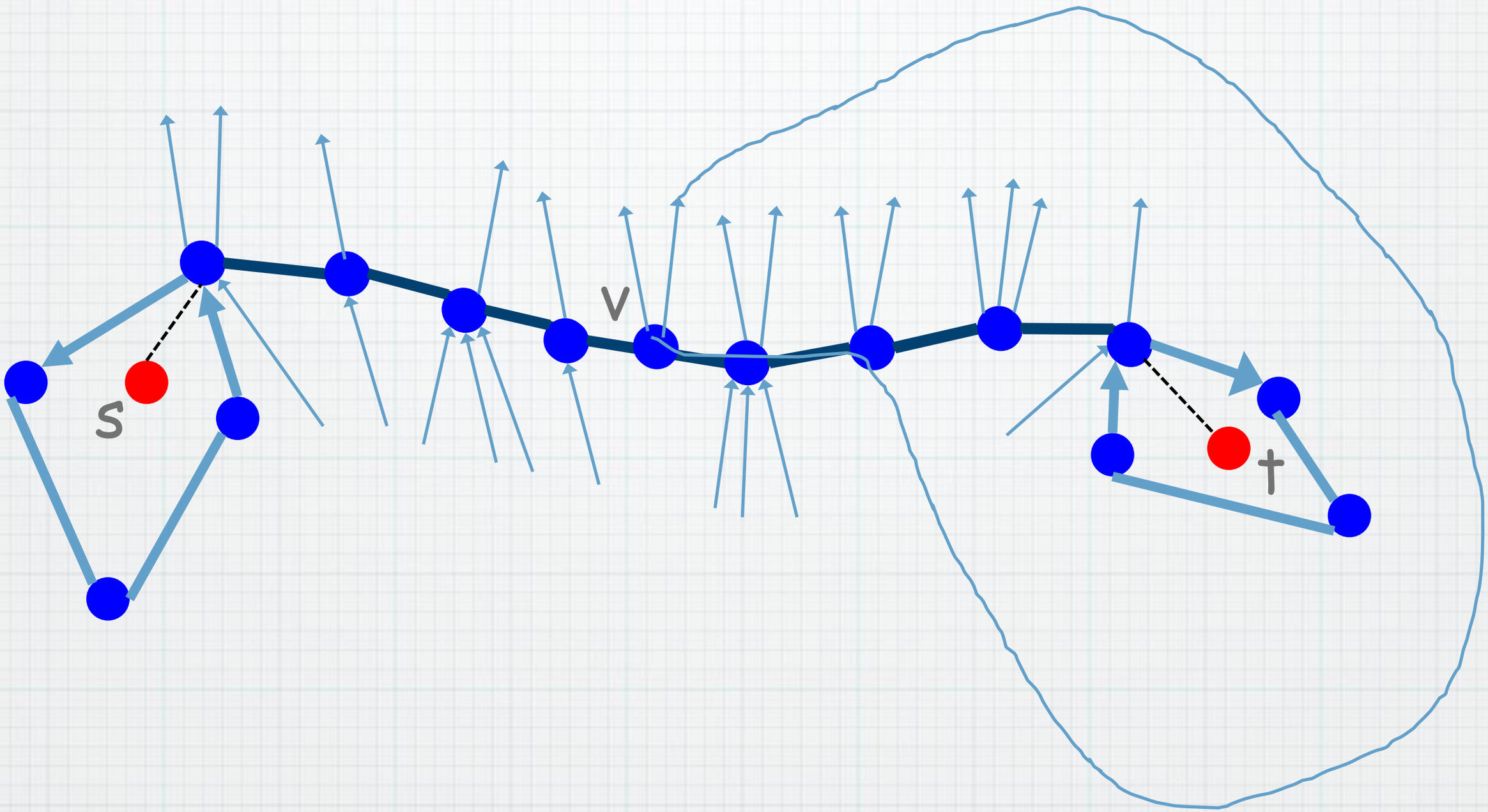


Shortest cycles do not cross

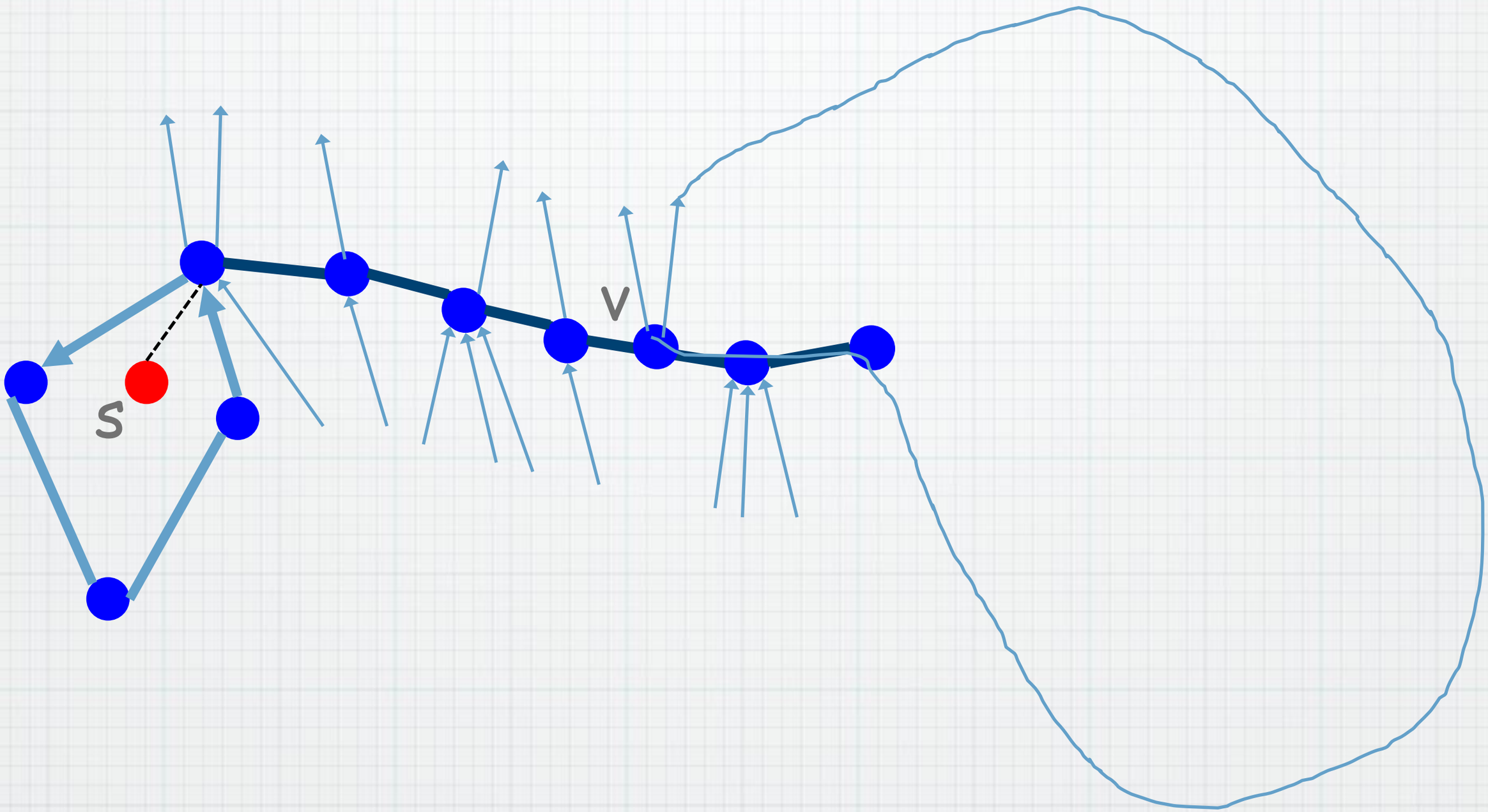
Take v_1 and v_2 to be the middle pair



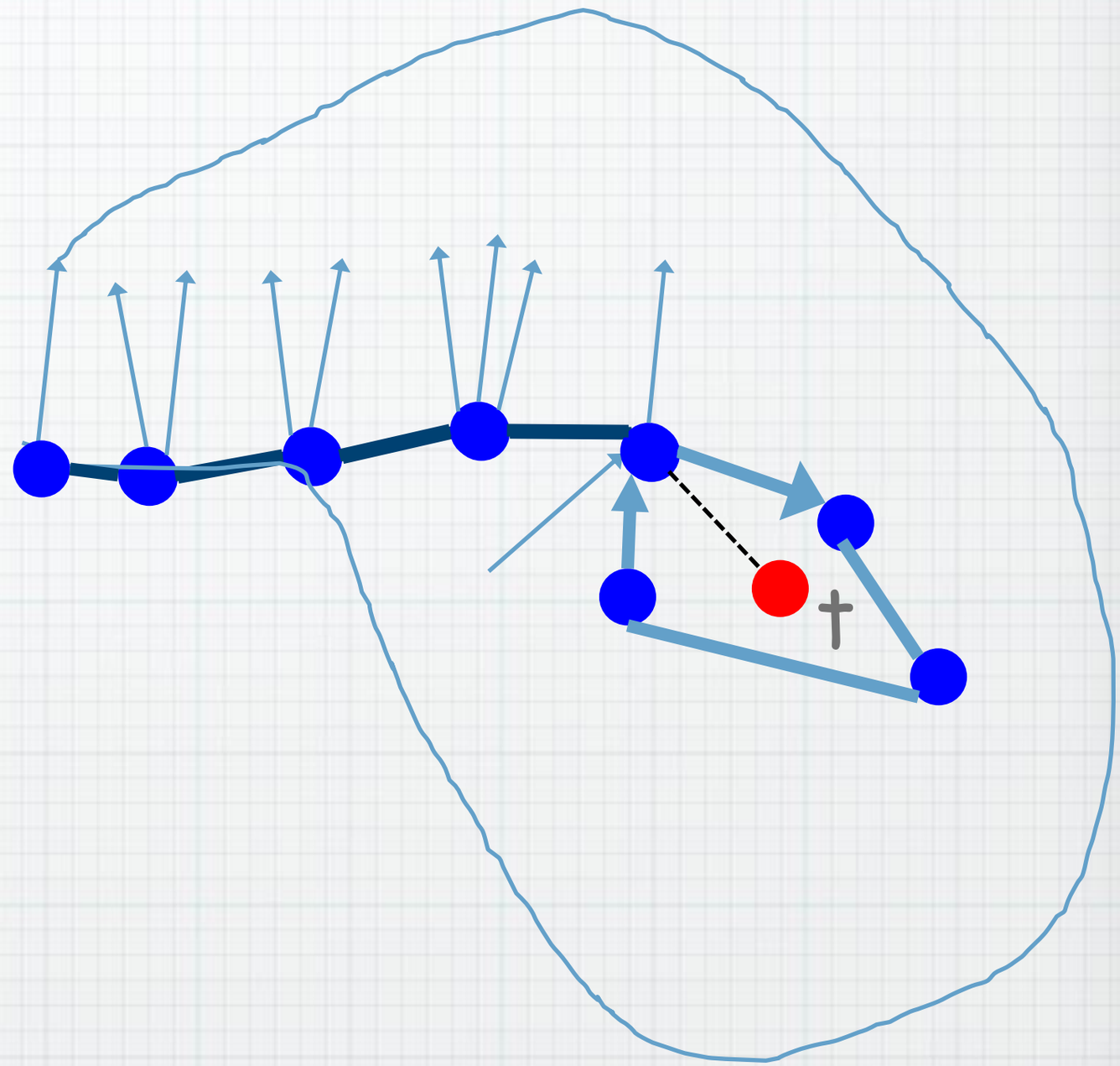
Take v_1 and v_2 to be the middle pair



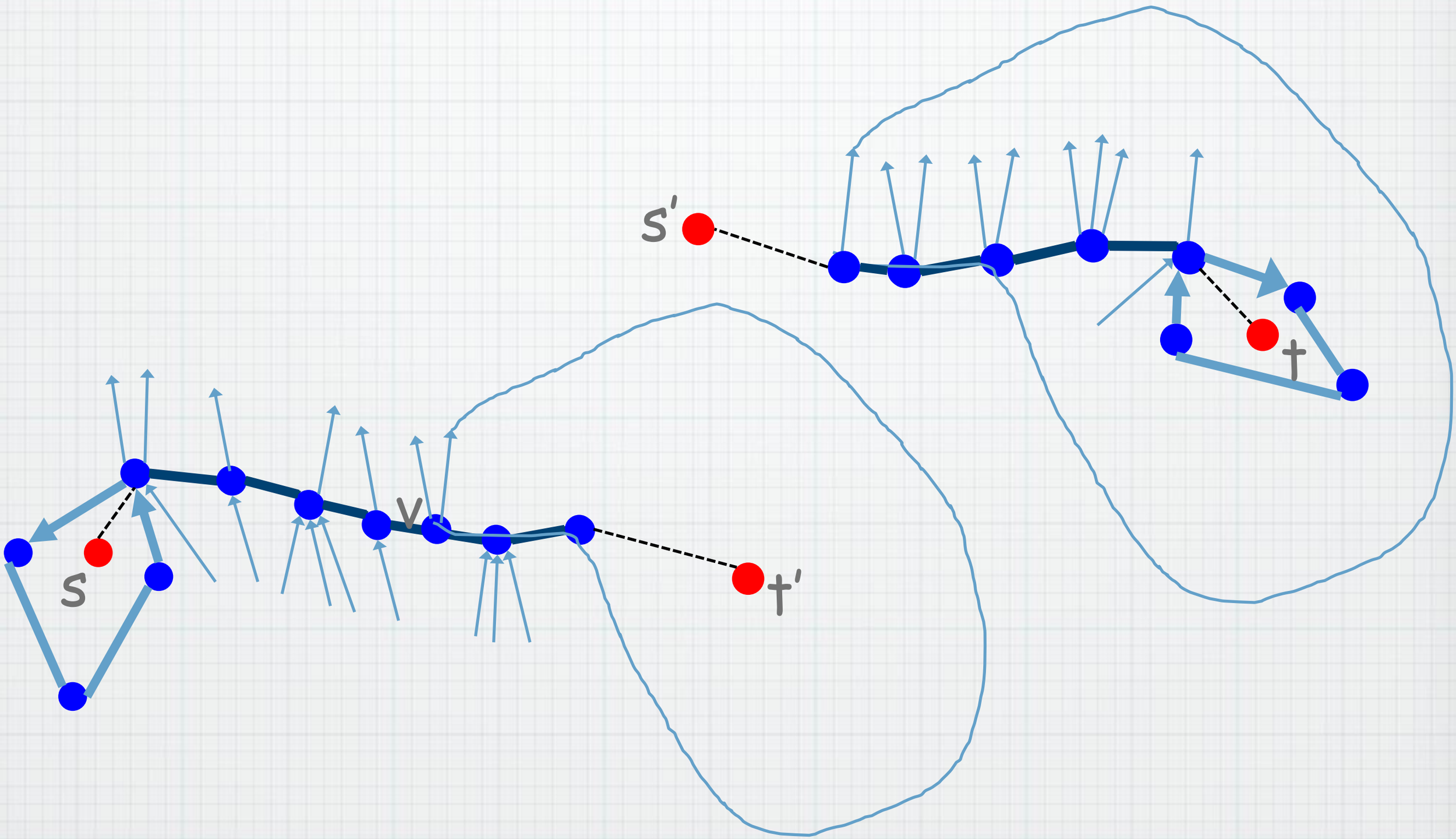
Split the problem



Split the problem

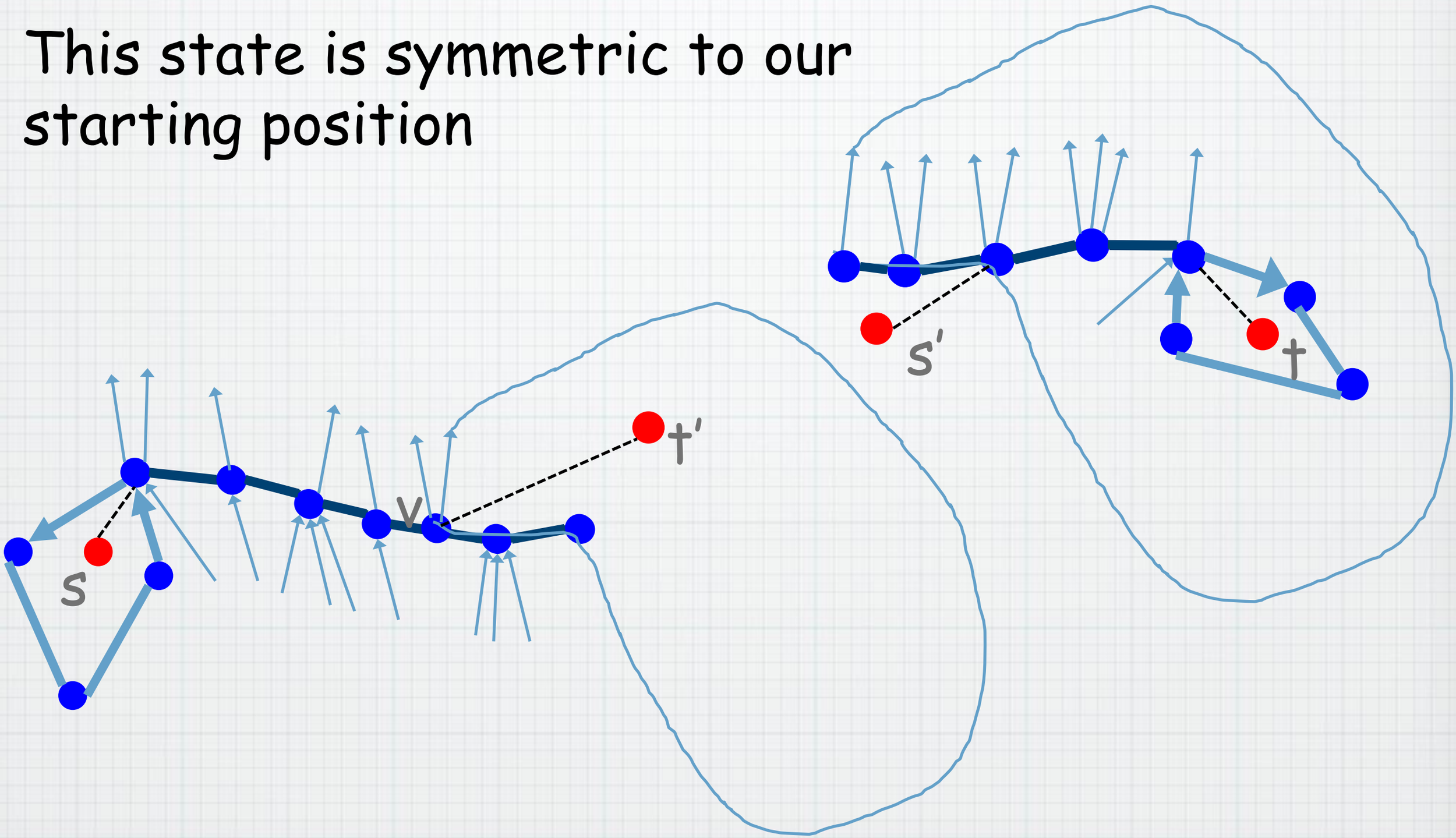


Add new source/sink



In fact

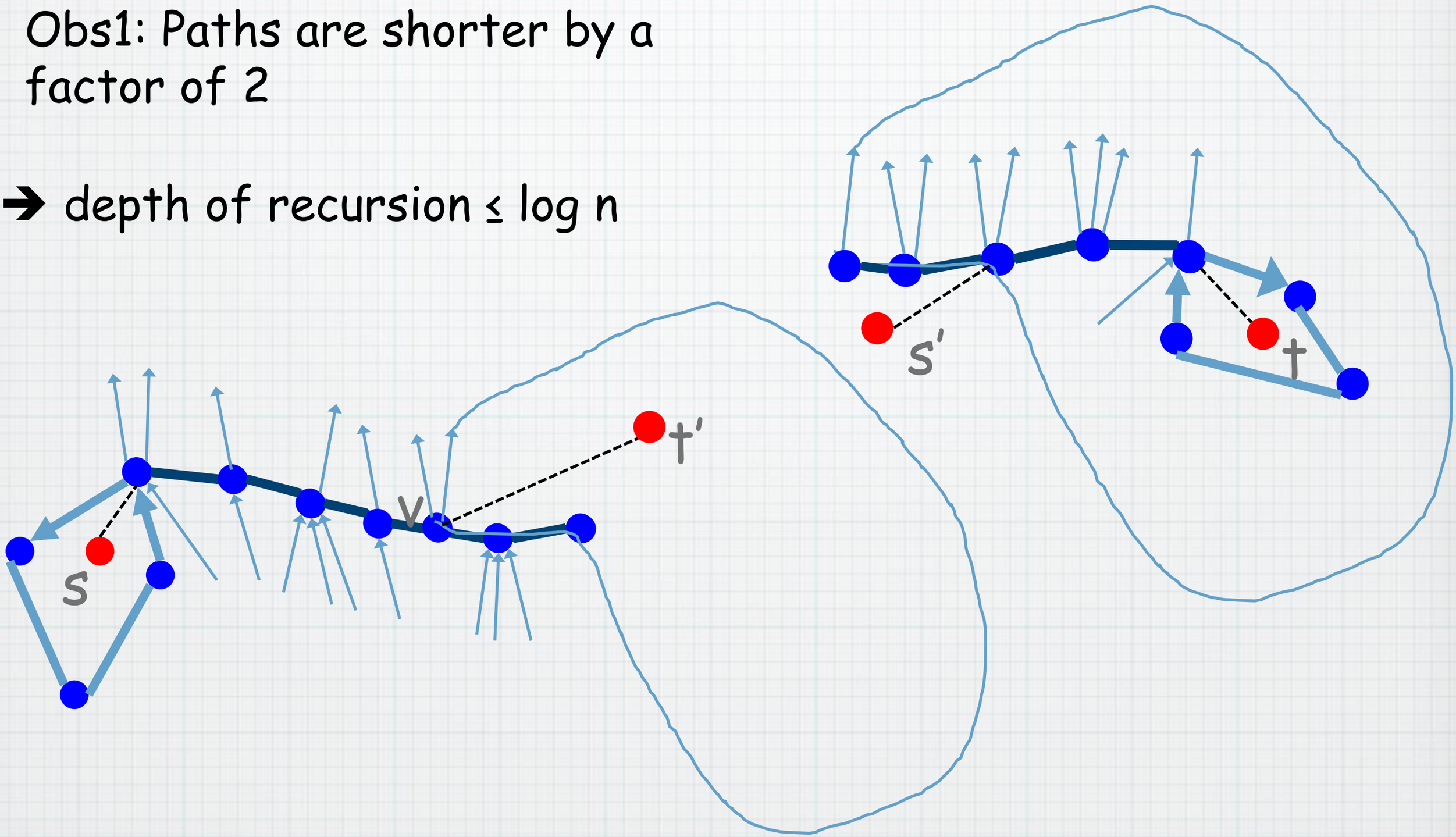
This state is symmetric to our starting position



Analysis

Obs1: Paths are shorter by a factor of 2

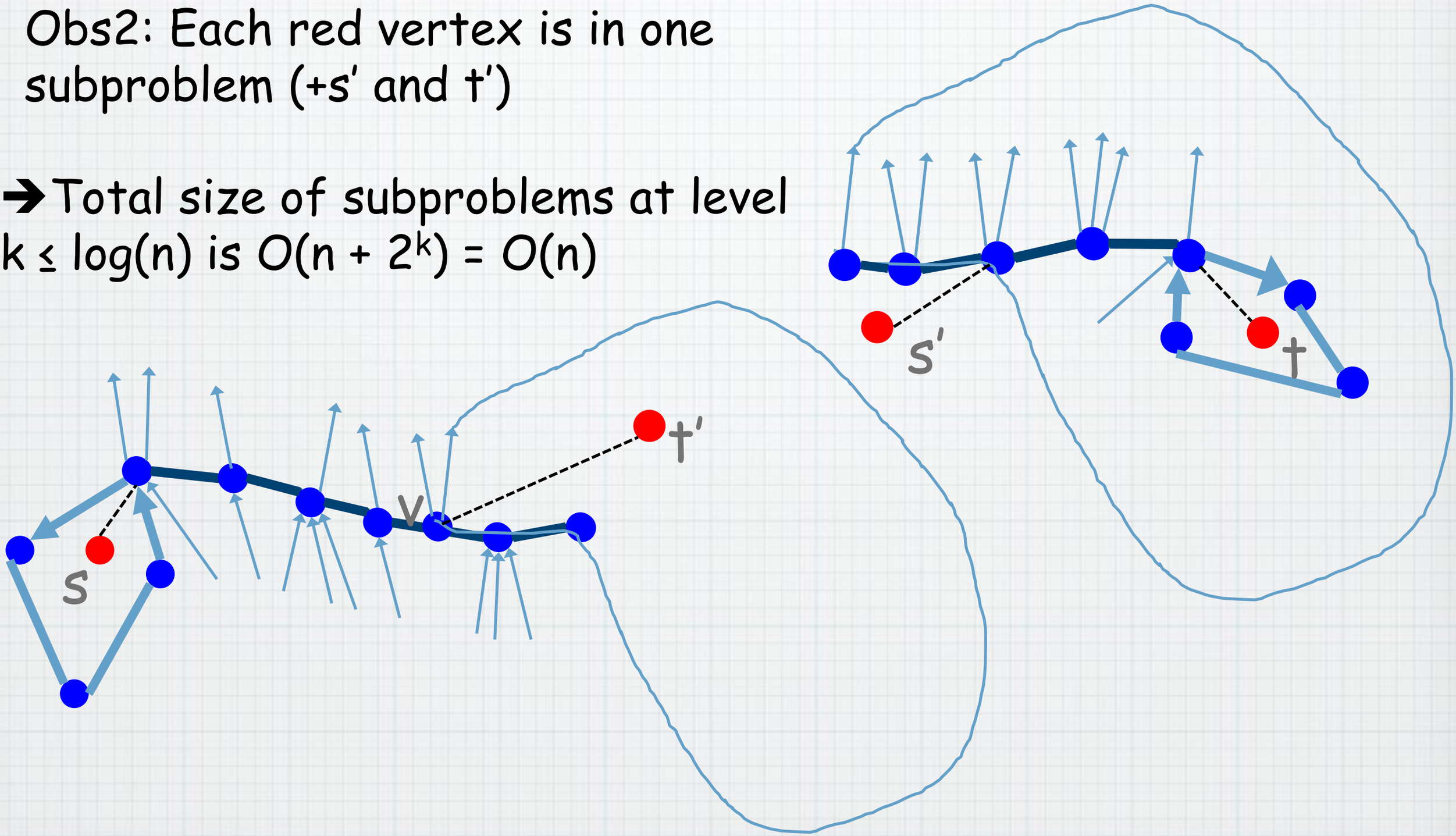
➔ depth of recursion ≤ log n



Analysis

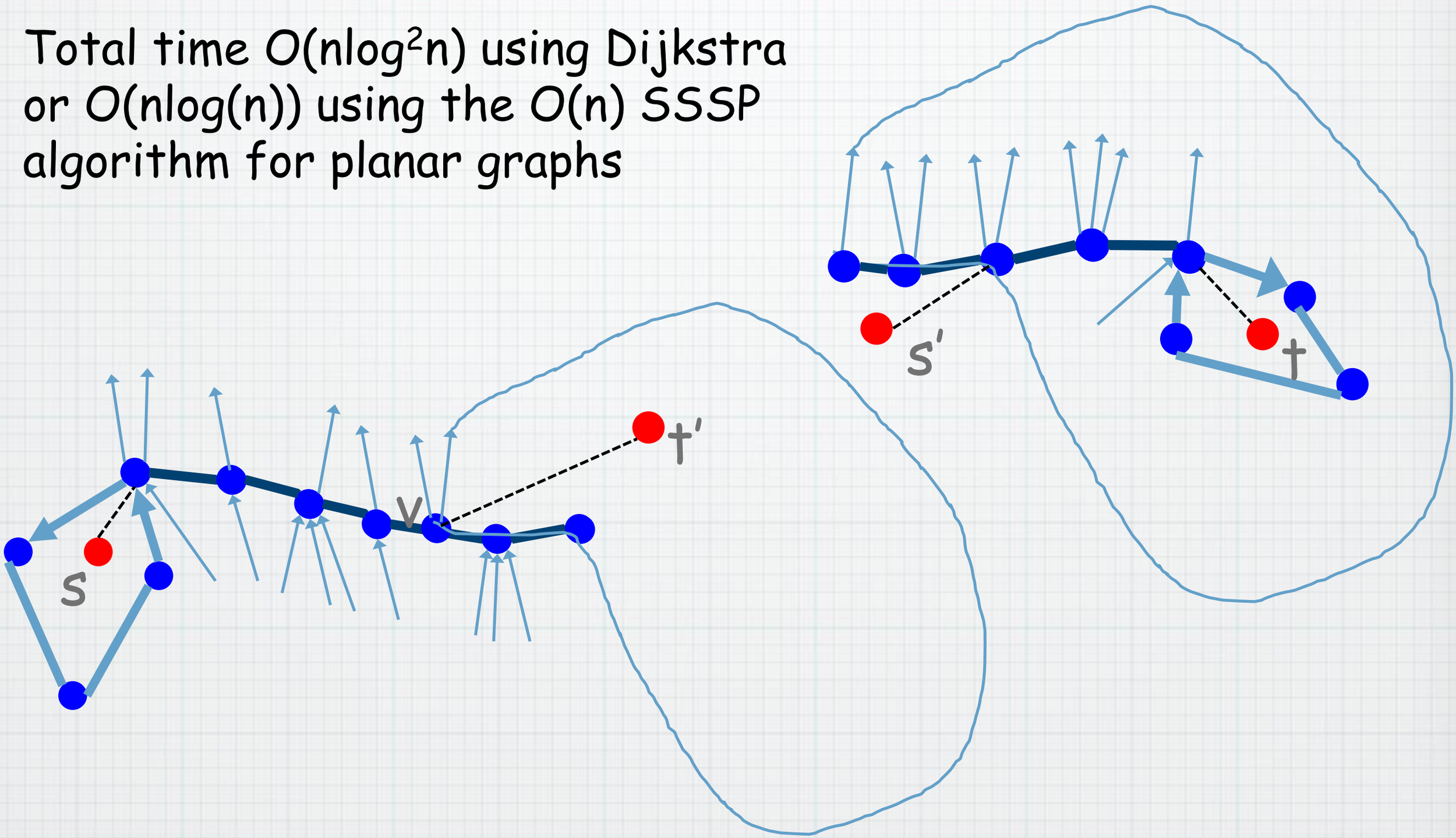
Obs2: Each red vertex is in one subproblem (+s′ and t′)

➔Total size of subproblems at level $k \leq \log(n)$ is $O(n + 2^k) = O(n)$



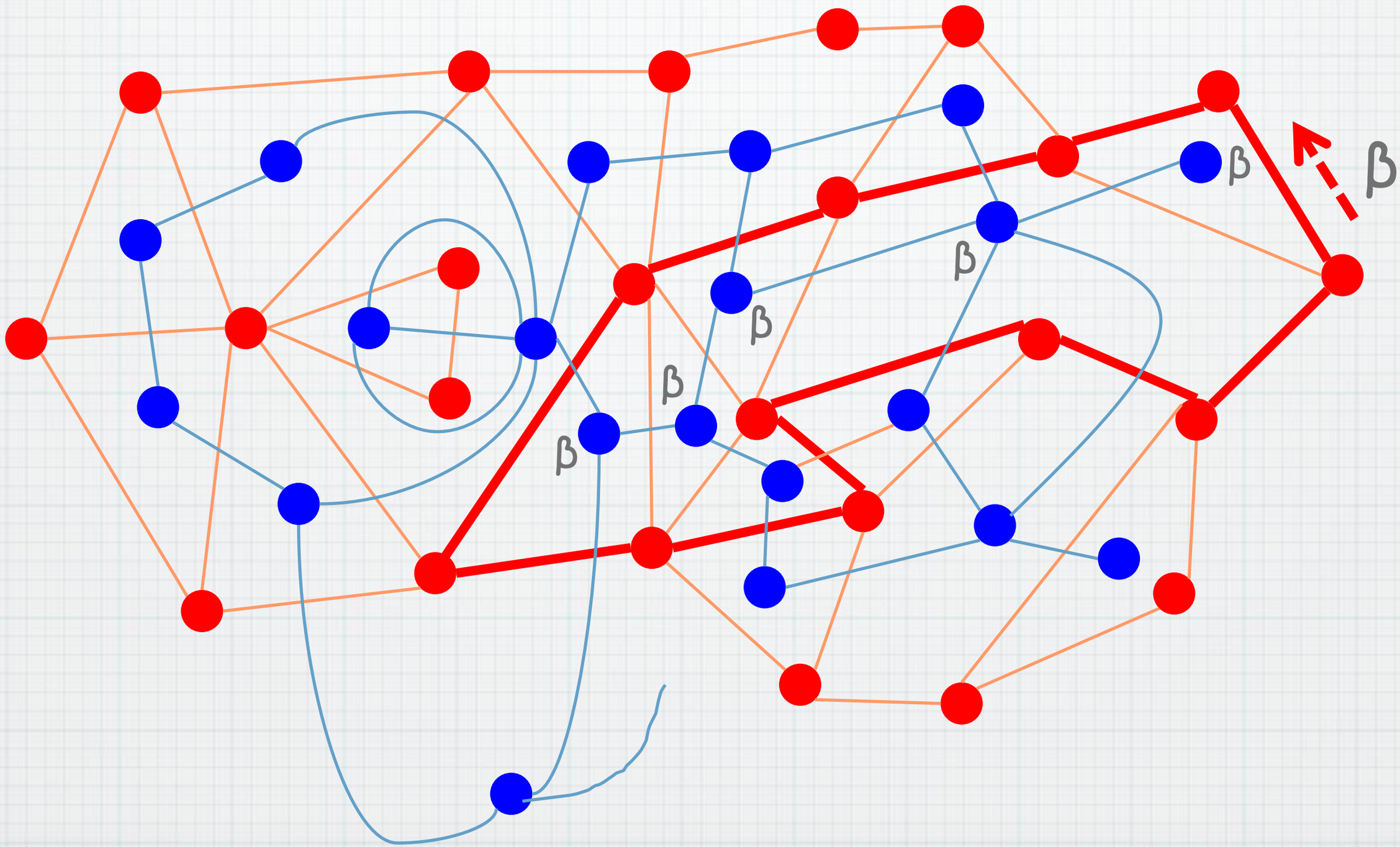
Summary

Total time $O(n\log^2 n)$ using Dijkstra or $O(n\log(n))$ using the $O(n)$ SSSP algorithm for planar graphs



Circulations and prices

Circulations and prices



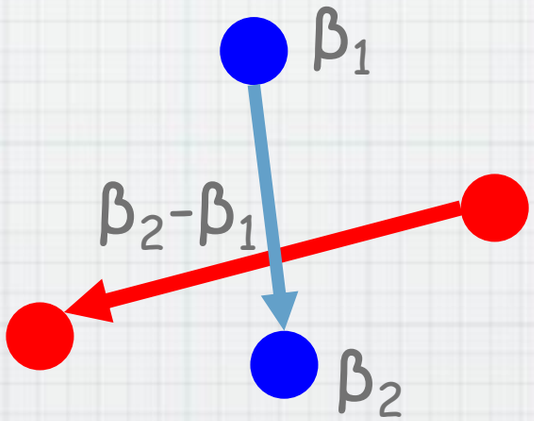
Circulations and prices

Decompose the flow into CCW cycles

Start with potentials of 0

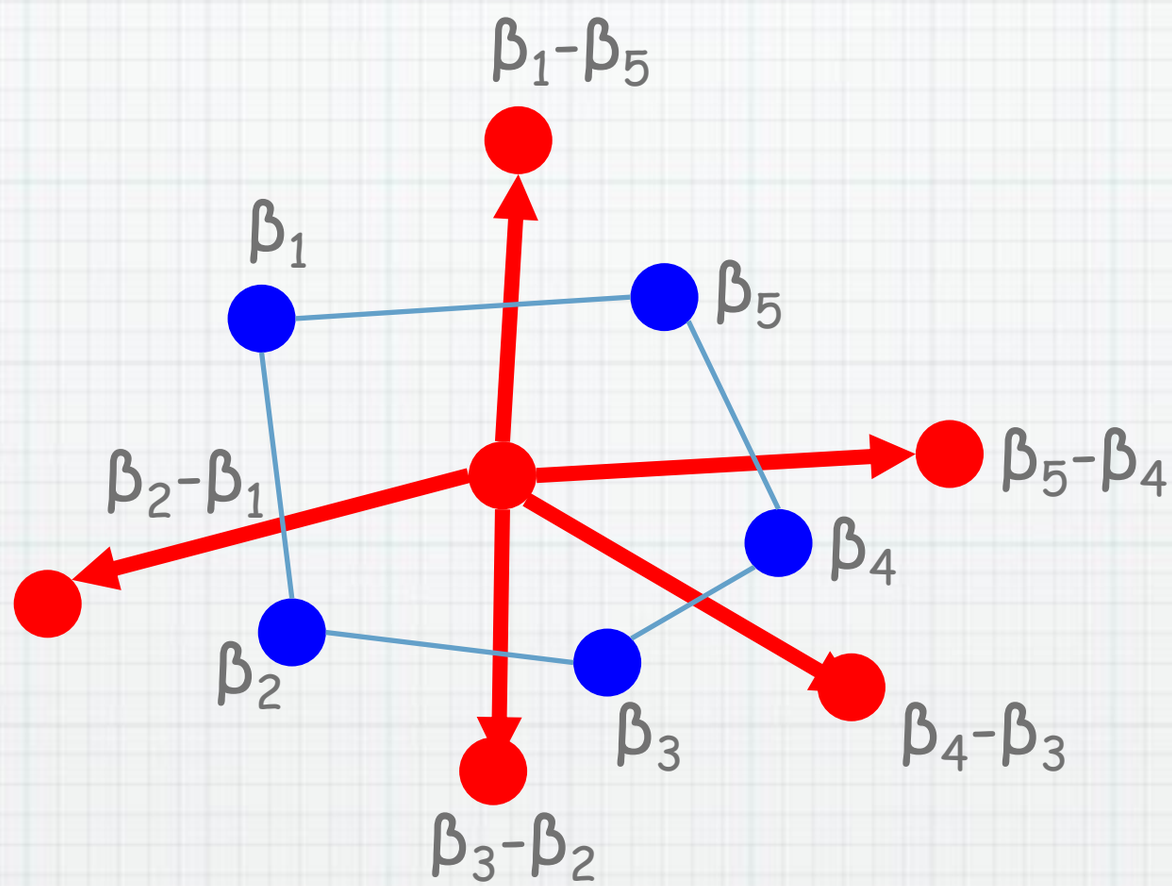
For each CCW cycle of value β, add β to the potentials of the faces inside the cycle

The flow along an edge is the difference in the potentials of its incident faces

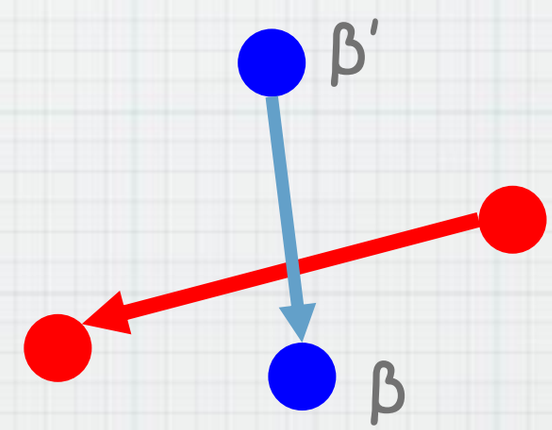


Circulations and prices

Any face prices define a circulation the same way

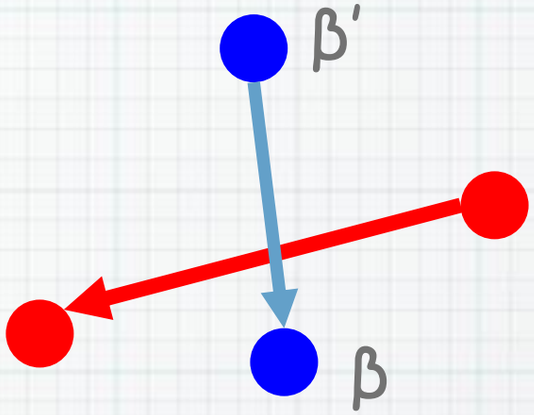


The flow is feasible iff $\beta - \beta' \leq u(e)$
Iff $\forall e$ $u(e) + \beta' - \beta \geq 0$ (nonnegative reduced costs)



Circulations and prices

Flow is feasible iff $\forall e\ u(e) + \beta' \geq \beta$



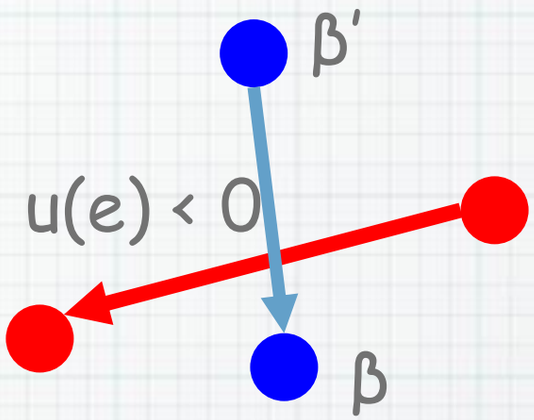
We can get potentials from any shortest path tree in the dual

The reduced costs equal the residual capacities of the corresponding flow.

2 applications for this connection

Feasible circulations

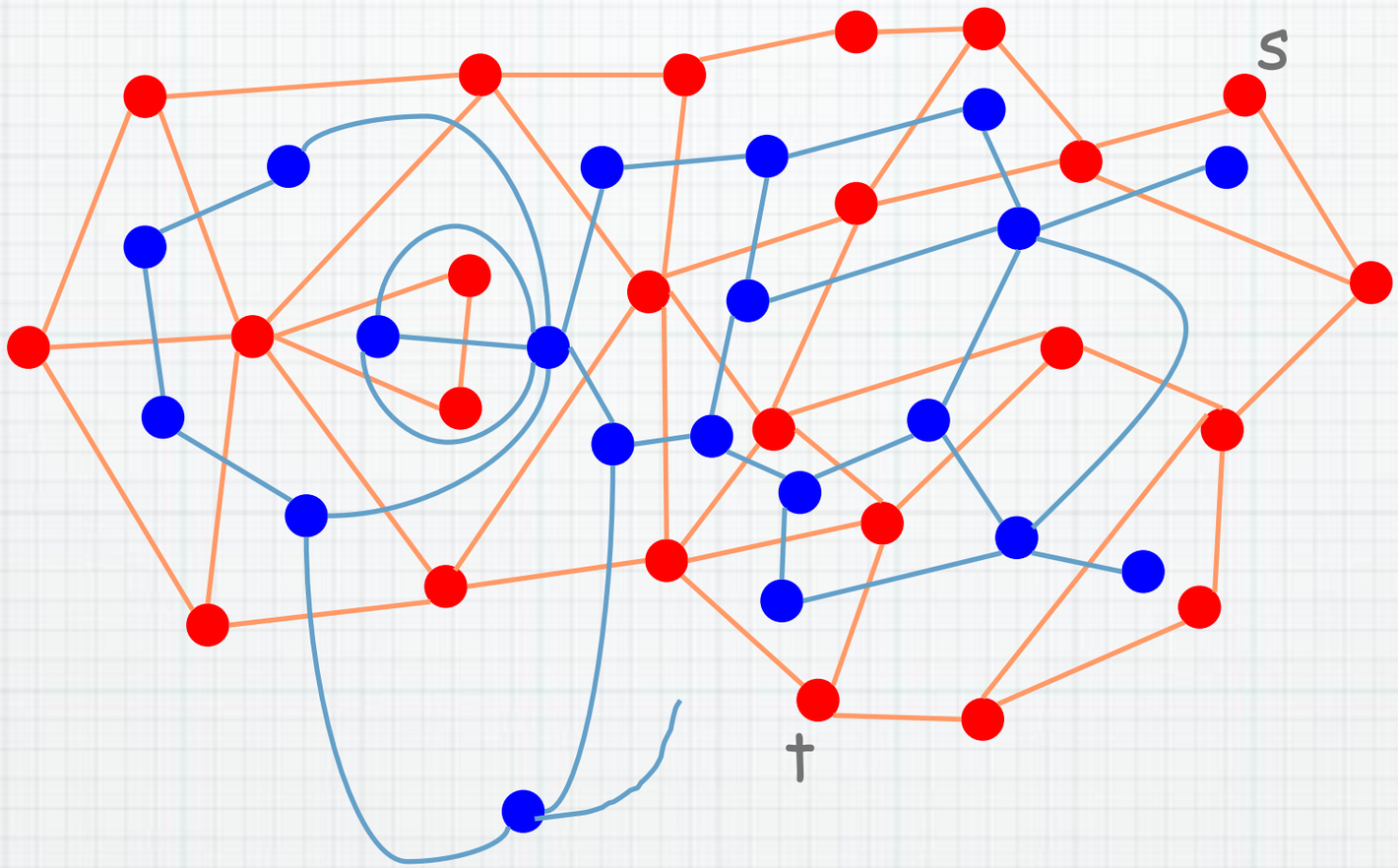
Negative capacity is a lower bound on the flow on the reverse arc



A circulation exists iff there are feasible potentials iff no negative cycles in the dual

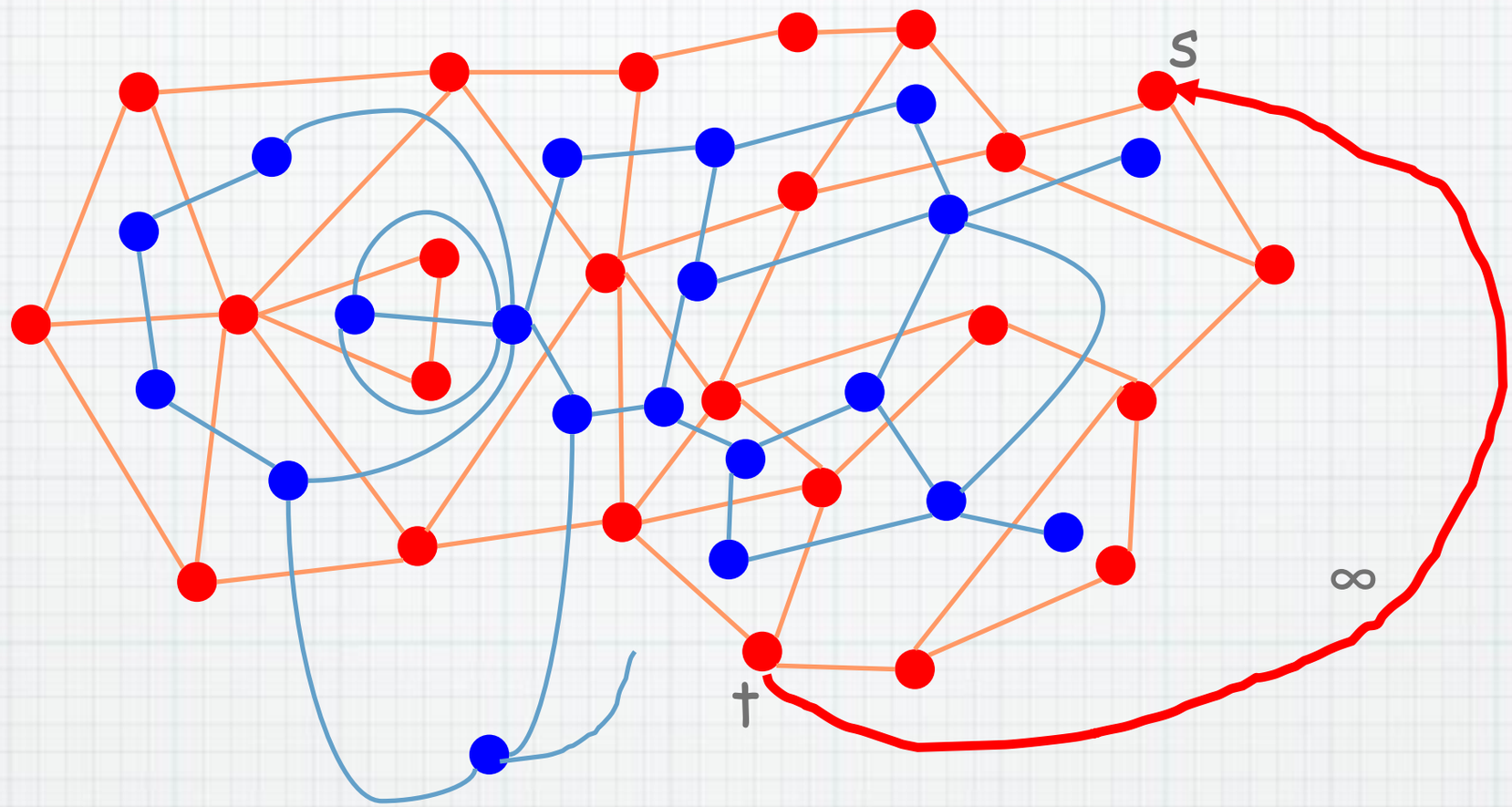
Can decide via a shortest path algorithm that can handle negative weights ➔ O(mn)

Max s-t flow when s and t are on the same face



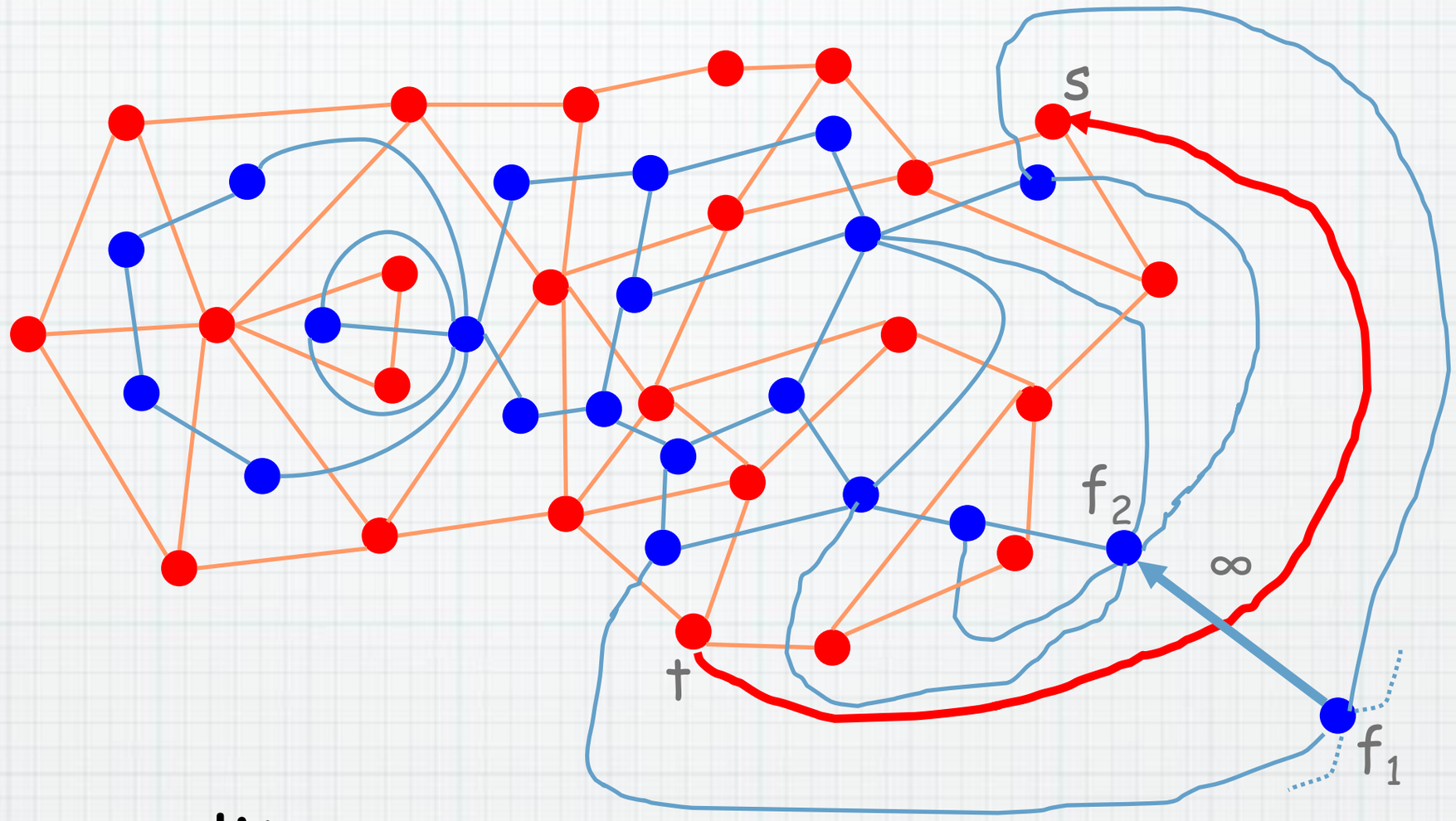
Find max flow from s to t when s and t are on the same face

Max s-t flow when s and t are on the same face



Add an edge from t to s with ∞ capacity

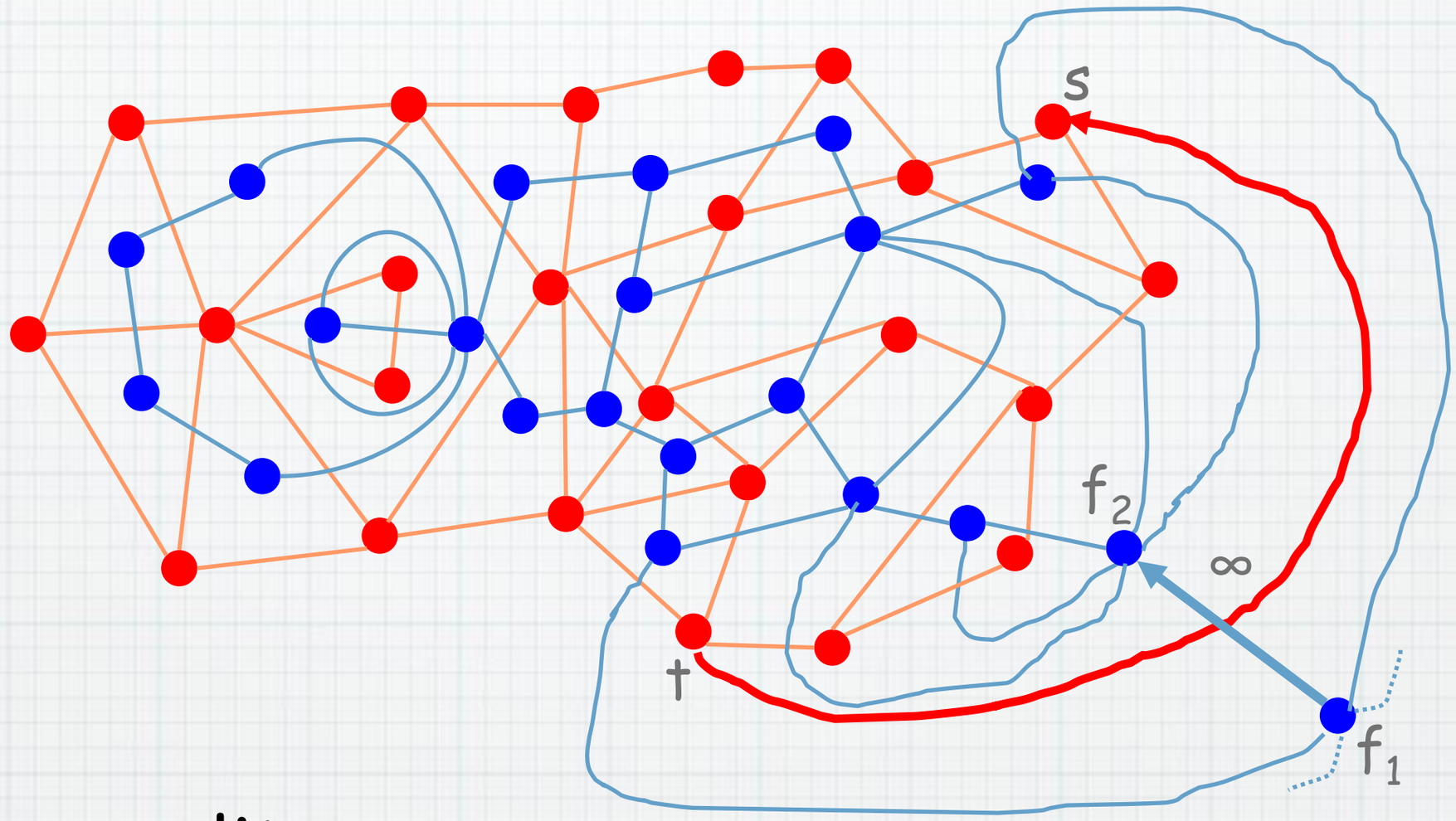
Max s-t flow when s and t are on the same face



Infinite face splits

Compute shortest paths from f_1 and define a flow according to these potentials

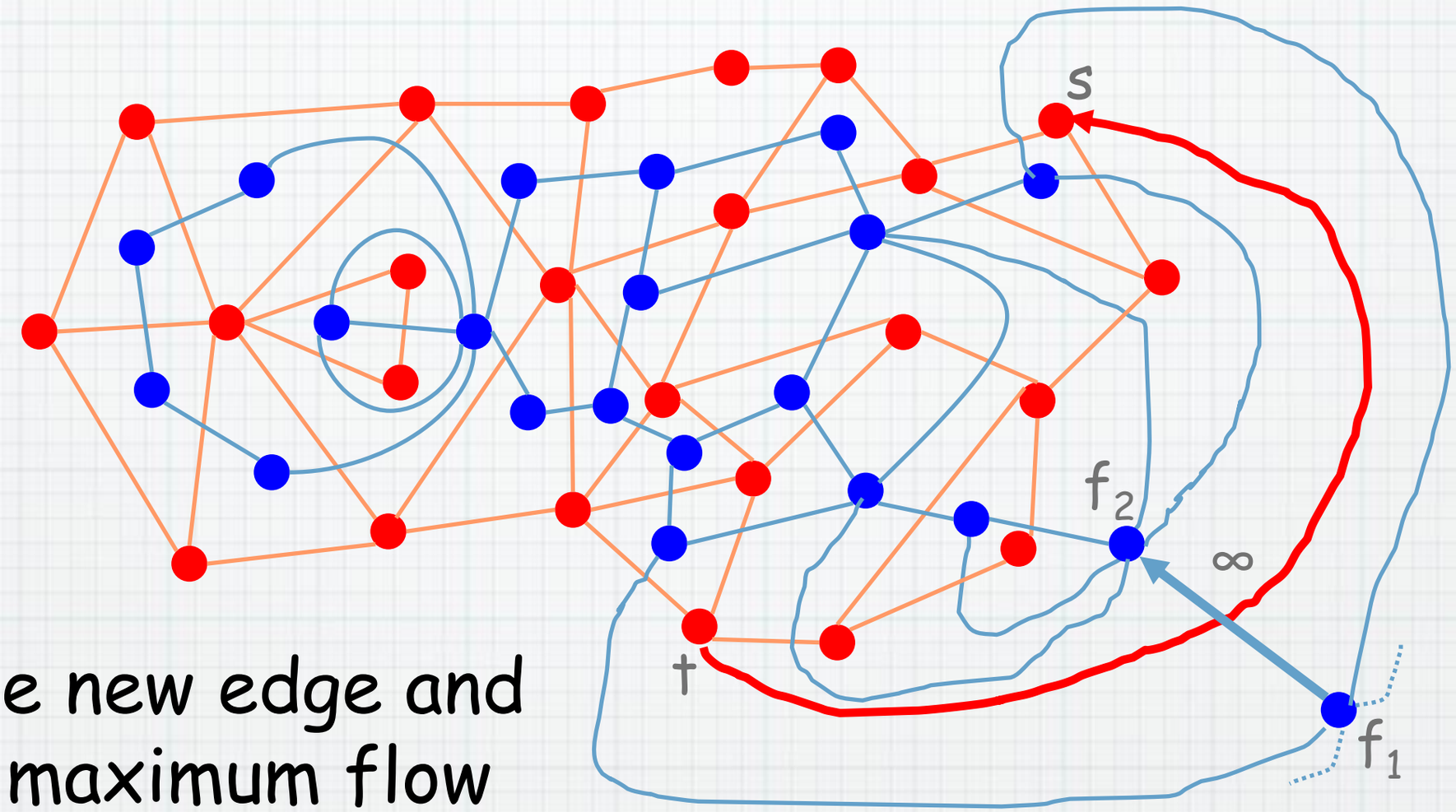
Max s-t flow when s and t are on the same face



Infinite face splits

Compute shortest paths from f_1 and define a flow according to these potentials

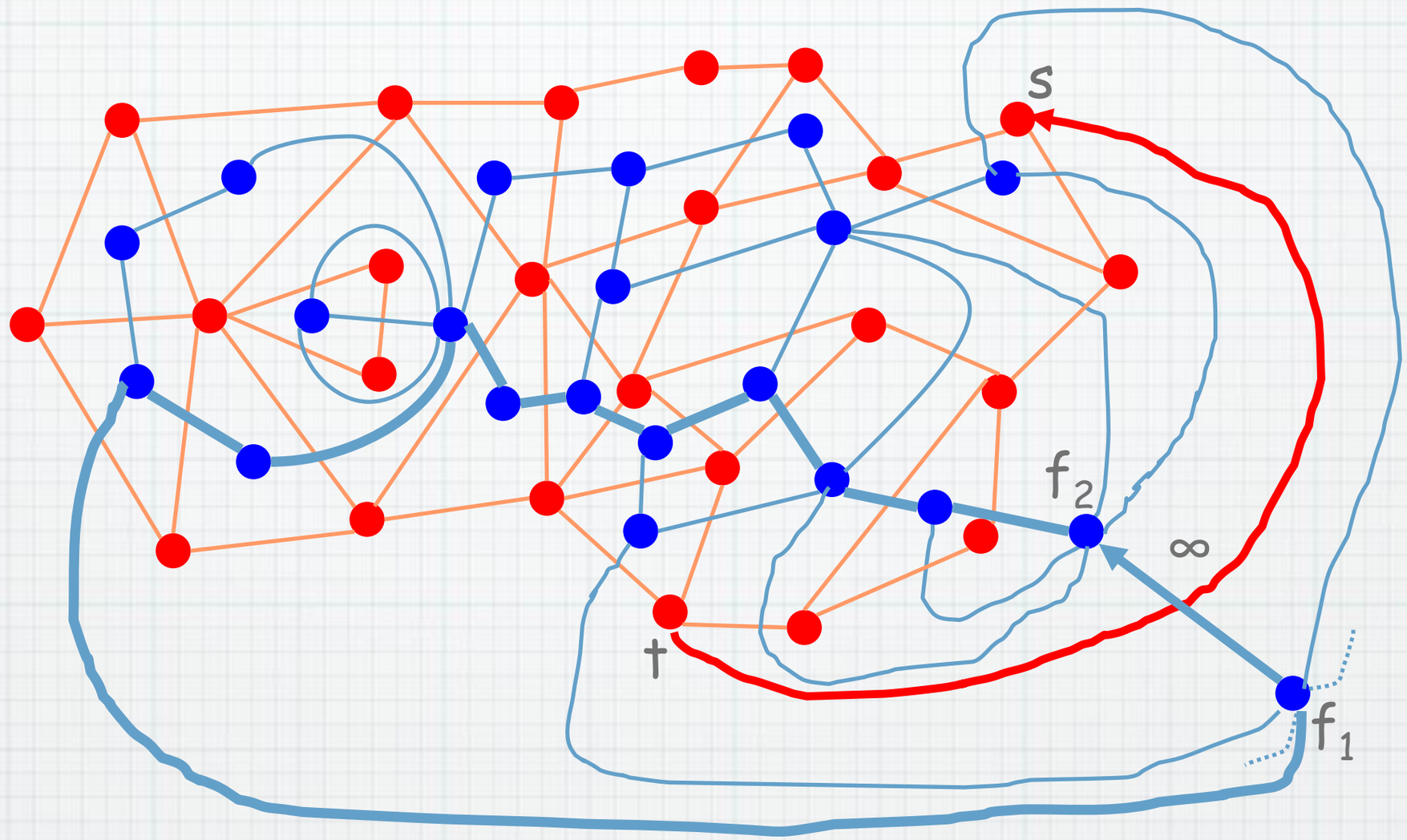
Max s-t flow when s and t are on the same face



Delete the new edge and you get a maximum flow from s to t

Proof. Its feasible (corres. to sp. in the dual), its maximum because it equals the minimum st-cut (=shortest path from f1 to f2)

Max s-t flow when s and t are on the same face



Proof. Its feasible (corres. to sp. in the dual), its maximum because it equals the minimum st-cut (=shortest path from f_1 to f_2)