Optimizing phylogenetic diversity across two trees

Newton Institute, December 18, 2007

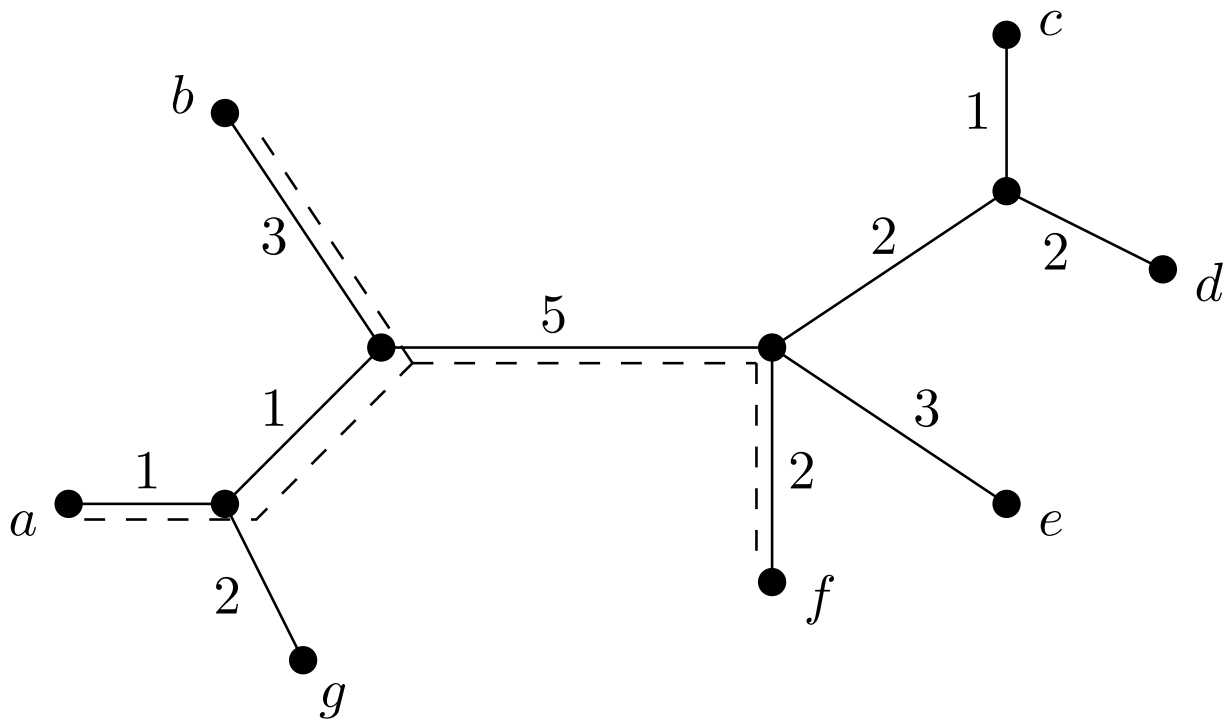
Magnus Bordewich

Joint work with Charles Semple and Andreas Spillner

Phylogenetic diversity

Instance: A phylogenetic tree T on taxa set X, and an integer k.

Question: Find a subset $Y \subseteq X$ such that $|Y| = k$ and $PD_T(Y)$ is maximum among all k-element subsets of X.



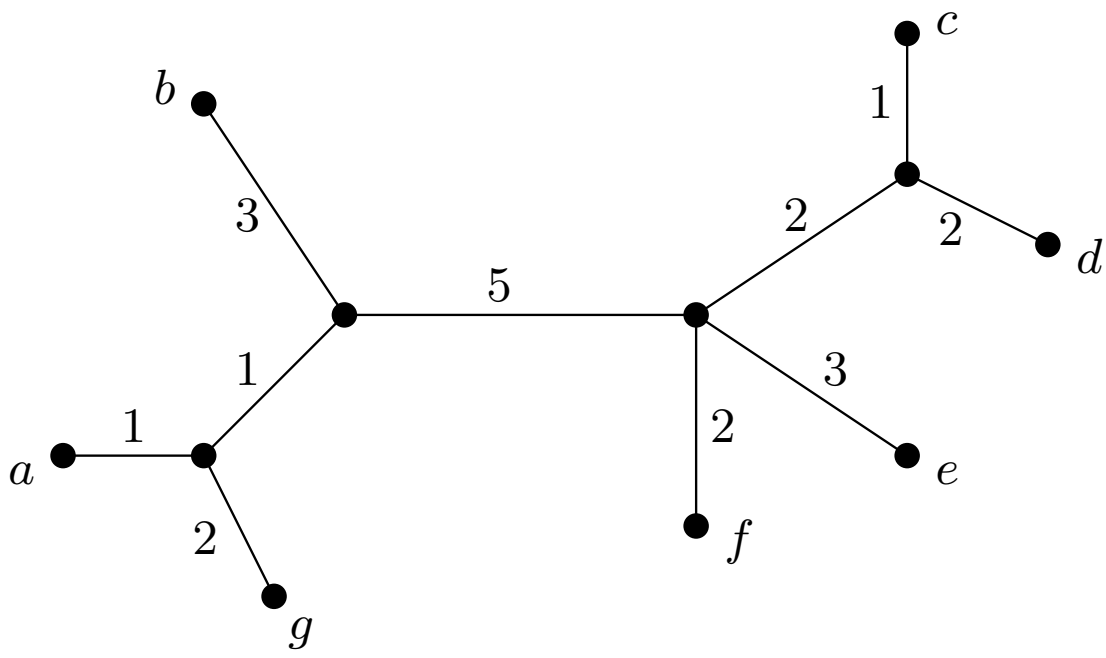
$PD_T(Y)$ is the length of the subtree connecting the taxa in Y, e.g. $PD_T(\{a, b, f\}) = 12$.

Greedy algorithm [Steel '05, Pardi and Goldman '05]

- Pick any two taxa maximally far apart: S_2.
- Form S_{i+1} from S_i by adding the taxa giving greatest PD increase.
- Return S_k, which is an optimal set of k taxa.

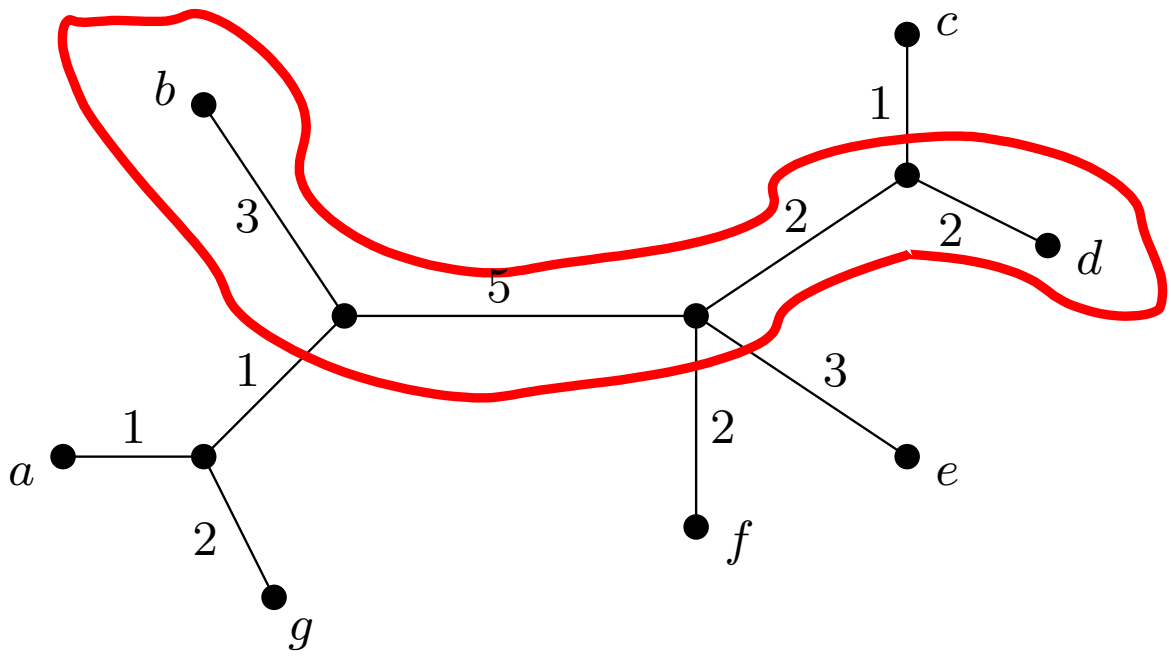
Greedy algorithm [Steel '05, Pardi and Goldman '05]

- Pick any two taxa maximally far apart: S_2.
- Form S_{i+1} from S_i by adding the taxa giving greatest PD increase.
- Return S_k, which is an optimal set of k taxa.



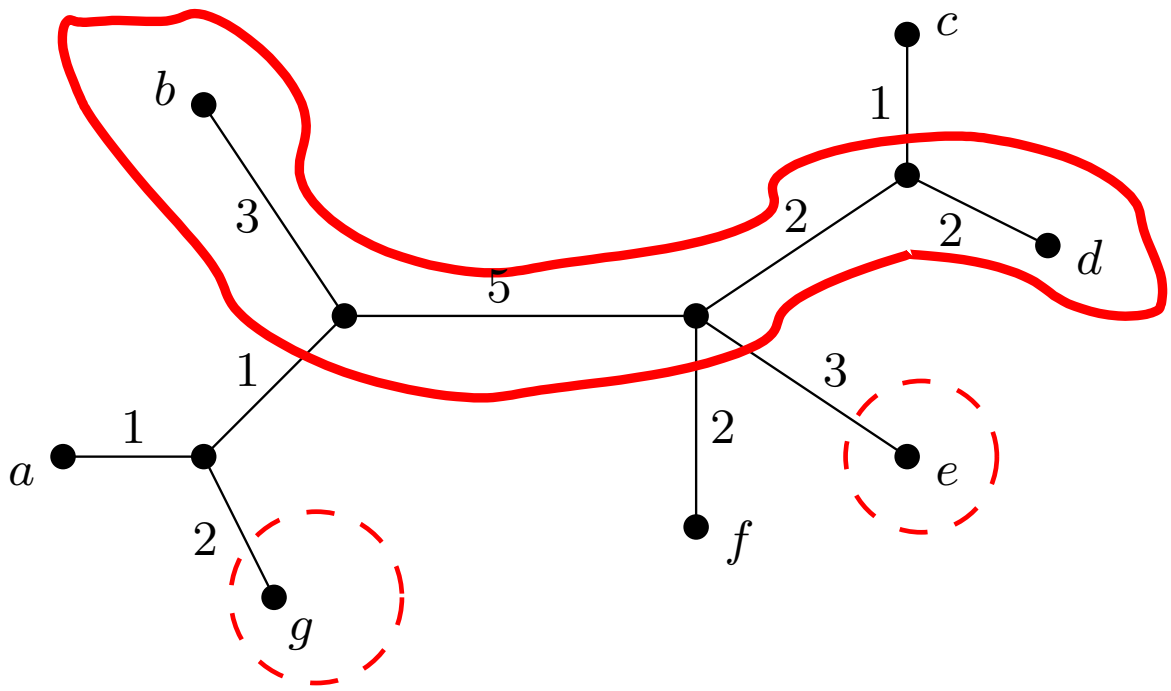
Greedy algorithm [Steel '05, Pardi and Goldman '05]

- Pick any two taxa maximally far apart: S_2.
- Form S_{i+1} from S_i by adding the taxa giving greatest PD increase.
- Return S_k, which is an optimal set of k taxa.



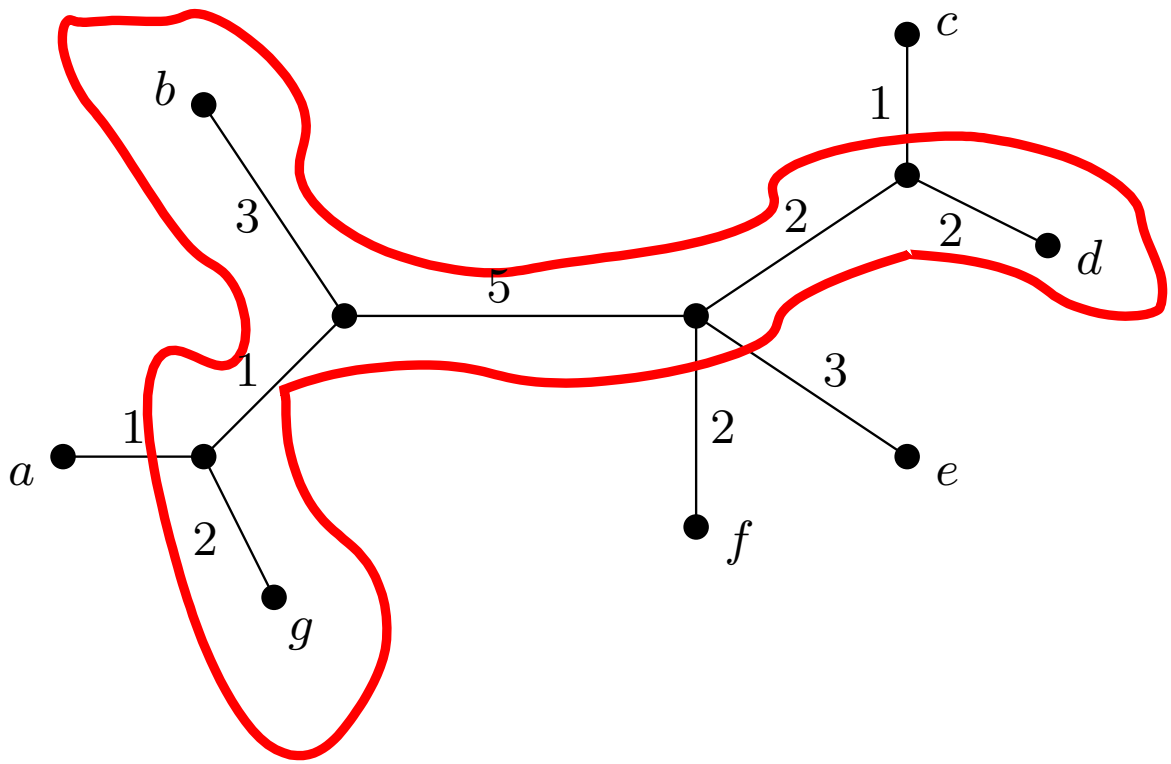
Greedy algorithm [Steel '05, Pardi and Goldman '05]

- Pick any two taxa maximally far apart: S_2.
- Form S_{i+1} from S_i by adding the taxa giving greatest PD increase.
- Return S_k, which is an optimal set of k taxa.



Greedy algorithm [Steel '05, Pardi and Goldman '05]

- Pick any two taxa maximally far apart: S_2.
- Form S_{i+1} from S_i by adding the taxa giving greatest PD increase.
- Return S_k, which is an optimal set of k taxa.



We don't always know the true tree

Analysis using different genes or methods may give different trees.

We could:

- attach weights to each gene-tree,
- find a set that maximises *weighted average* PD.

We don't always know the true tree

Analysis using different genes or methods may give different trees.

We could:

- attach weights to each gene-tree,
- find a set that maximises *weighted average* PD.

Problem: WeightedAveragePD: WAPD$_t$.

Instance: A collection $\mathcal{T} = \{T_1, \ldots, T_t\}$ of phylogenetic trees on the same set of taxa X, weights $\{\lambda_1, \ldots, \lambda_t\}$, and an integer k.

Question: Find a subset $Y \subseteq X$ such that $|Y| = k$ and $PD_{\mathcal{T}}(Y) = \lambda_1 PD_{T_1}(Y) + \cdots + \lambda_t PD_{T_t}(Y)$ is maximum among all k-element subsets of X.

We don't always know the true tree

Analysis using different genes or methods may give different trees.

We could:

- attach weights to each gene-tree,
- find a set that maximises *weighted average* PD.

Problem: WeightedAveragePD: WAPD$_t$.

Instance: A collection $\mathcal{T} = \{T_1, \ldots, T_t\}$ of phylogenetic trees on the same set of taxa X, weights $\{\lambda_1, \ldots, \lambda_t\}$, and an integer k.

Question: Find a subset $Y \subseteq X$ such that $|Y| = k$ and $PD_{\mathcal{T}}(Y) = \lambda_1 PD_{T_1}(Y) + \cdots + \lambda_t PD_{T_t}(Y)$ is maximum among all k-element subsets of X.

- WAPD$_1$ is essentially PD. Greedy algorithm works.

We don't always know the true tree

Analysis using different genes or methods may give different trees.

We could:

- attach weights to each gene-tree,
- find a set that maximises *weighted average* PD.

Problem: WeightedAveragePD: WAPD$_t$.

Instance: A collection $\mathcal{T} = \{T_1, \ldots, T_t\}$ of phylogenetic trees on the same set of taxa X, weights $\{\lambda_1, \ldots, \lambda_t\}$, and an integer k.

Question: Find a subset $Y \subseteq X$ such that $|Y| = k$ and $PD_{\mathcal{T}}(Y) = \lambda_1 PD_{T_1}(Y) + \cdots + \lambda_t PD_{T_t}(Y)$ is maximum among all k-element subsets of X.

- WAPD$_1$ is essentially PD. Greedy algorithm works.
- WAPD$_t$, for $t \geq 3$, is NP-hard.

We don't always know the true tree

Analysis using different genes or methods may give different trees.

We could:

- attach weights to each gene-tree,
- find a set that maximises *weighted average* PD.

Problem: WeightedAveragePD: WAPD$_t$.

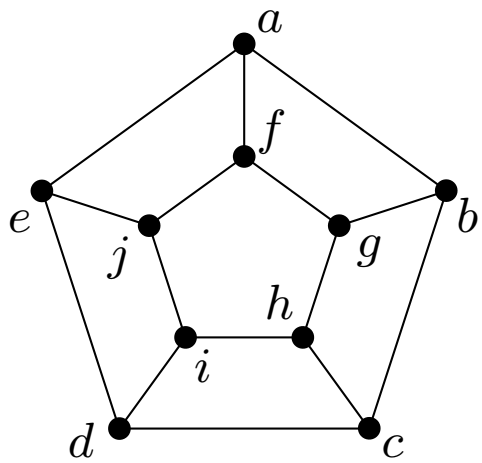
Instance: A collection $\mathcal{T} = \{T_1, \ldots, T_t\}$ of phylogenetic trees on the same set of taxa X, weights $\{\lambda_1, \ldots, \lambda_t\}$, and an integer k.

Question: Find a subset $Y \subseteq X$ such that $|Y| = k$ and $PD_{\mathcal{T}}(Y) = \lambda_1 PD_{T_1}(Y) + \cdots + \lambda_t PD_{T_t}(Y)$ is maximum among all k-element subsets of X.

- WAPD$_1$ is essentially PD. Greedy algorithm works.
- WAPD$_t$, for $t \geq 3$, is NP-hard.
- WAPD$_2$: in P or NP-hard?

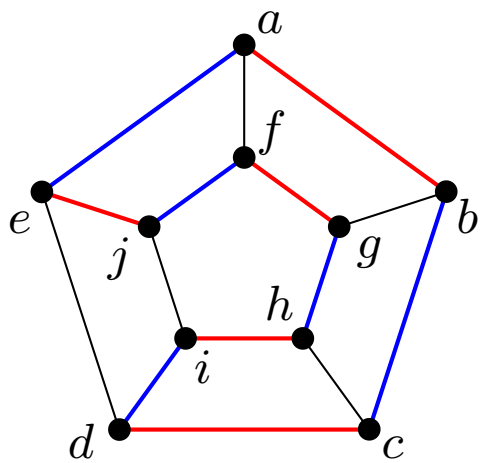
WAPD_3 is NP-hard: reduction from vertex cover

- G an instance of cubic planar VC.



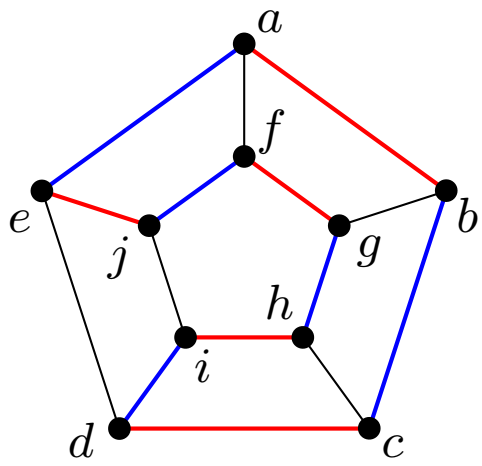
WAPD_3 is NP-hard: reduction from vertex cover

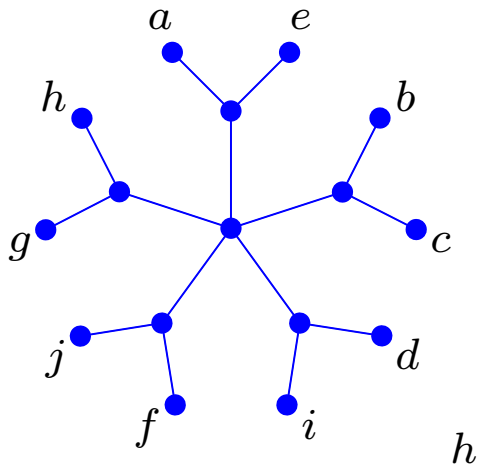
- G an instance of cubic planar VC.

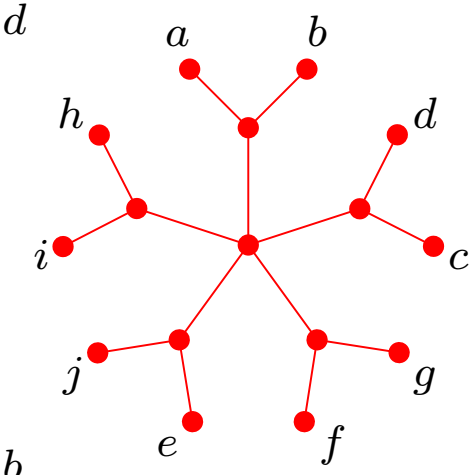


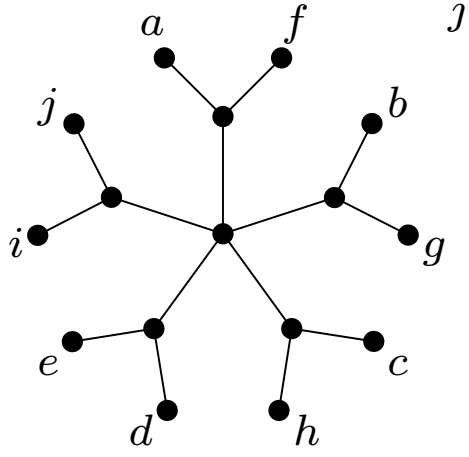
WAPD_3 is NP-hard: reduction from vertex cover

- G an instance of cubic planar VC.



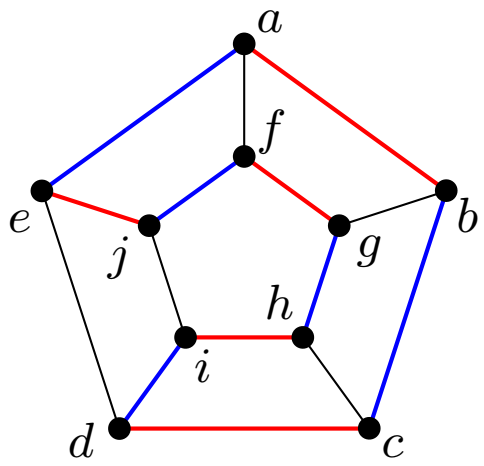


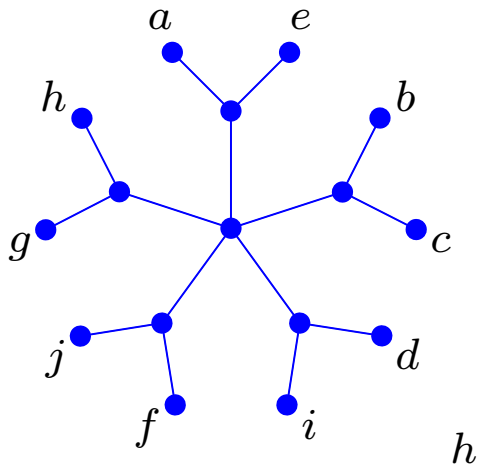


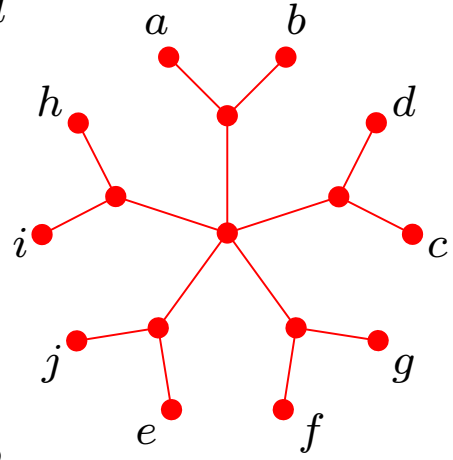


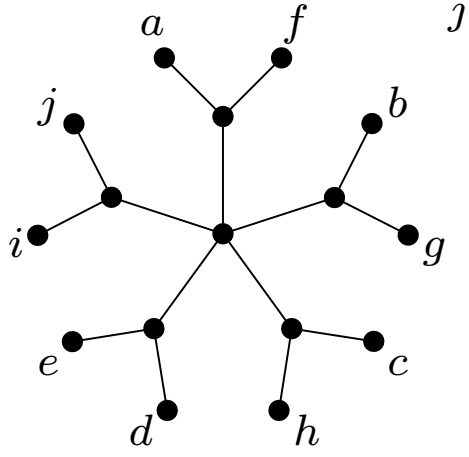
$WAPD_3$ is NP-hard: reduction from vertex cover

- G an instance of cubic planar VC.









Lemma

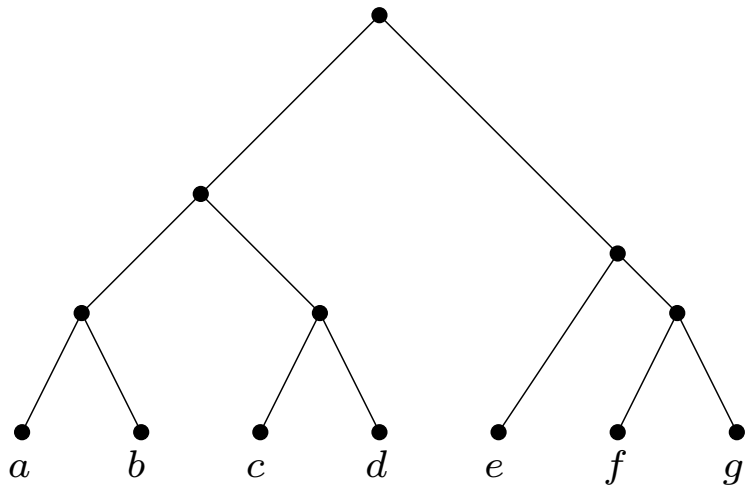
For $k \geq 3$, G has a vertex cover of size k iff the optimal solution to $WAPD_3$ is $|E(G)| + 3k$.

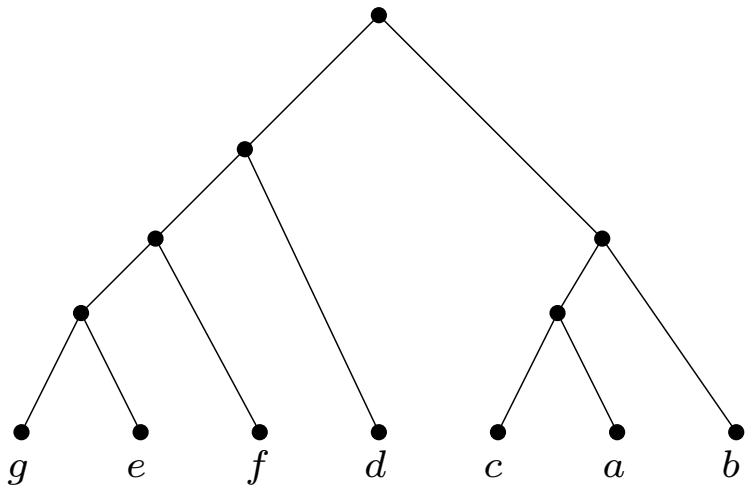
A polynomial time algorithm for WAPD_2

- Start with two trees T_1, T_2 and an integer k.
- Convert these trees into a network with capacities and costs.
- Solve a *minimum cost flow problem* on the network.
- Minimum cost of of a k-flow is exactly the maximum WAPD_2.

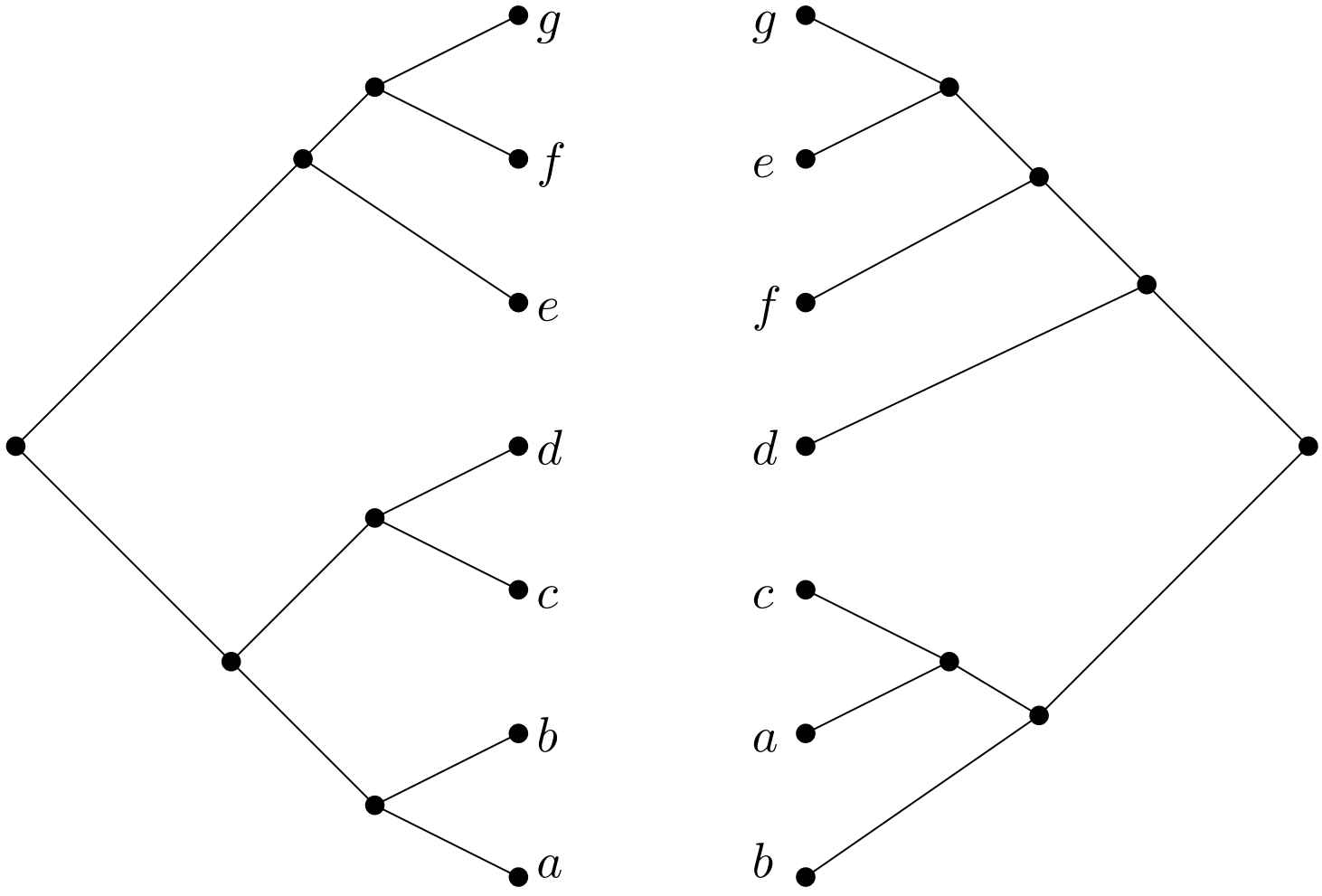
Minimum cost flow is an old and well studied problem which can be solved by simple and efficient algorithms ($O(n^2 \log^2 n)$).

The construction (rooted case)

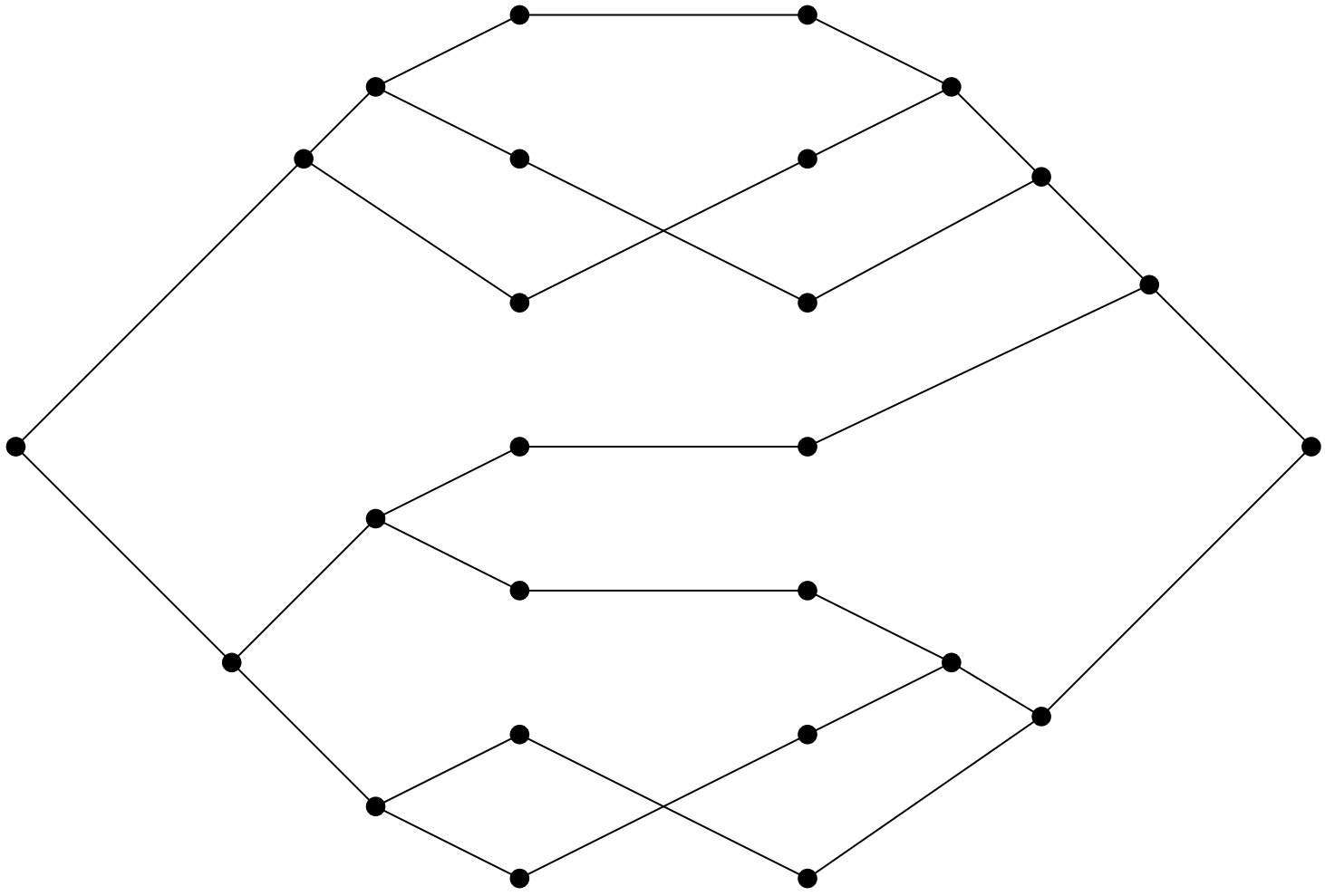




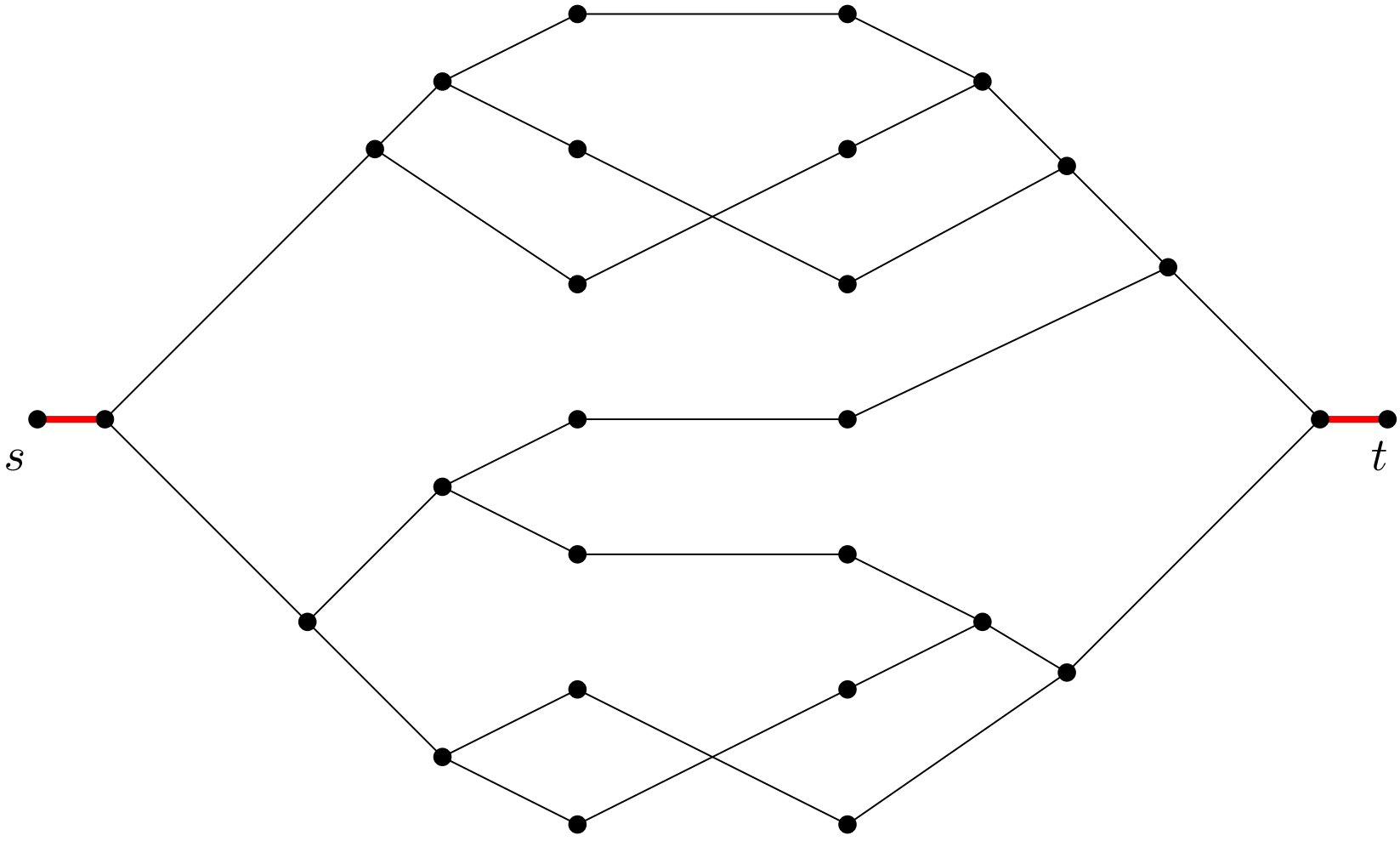
The construction (rooted case)



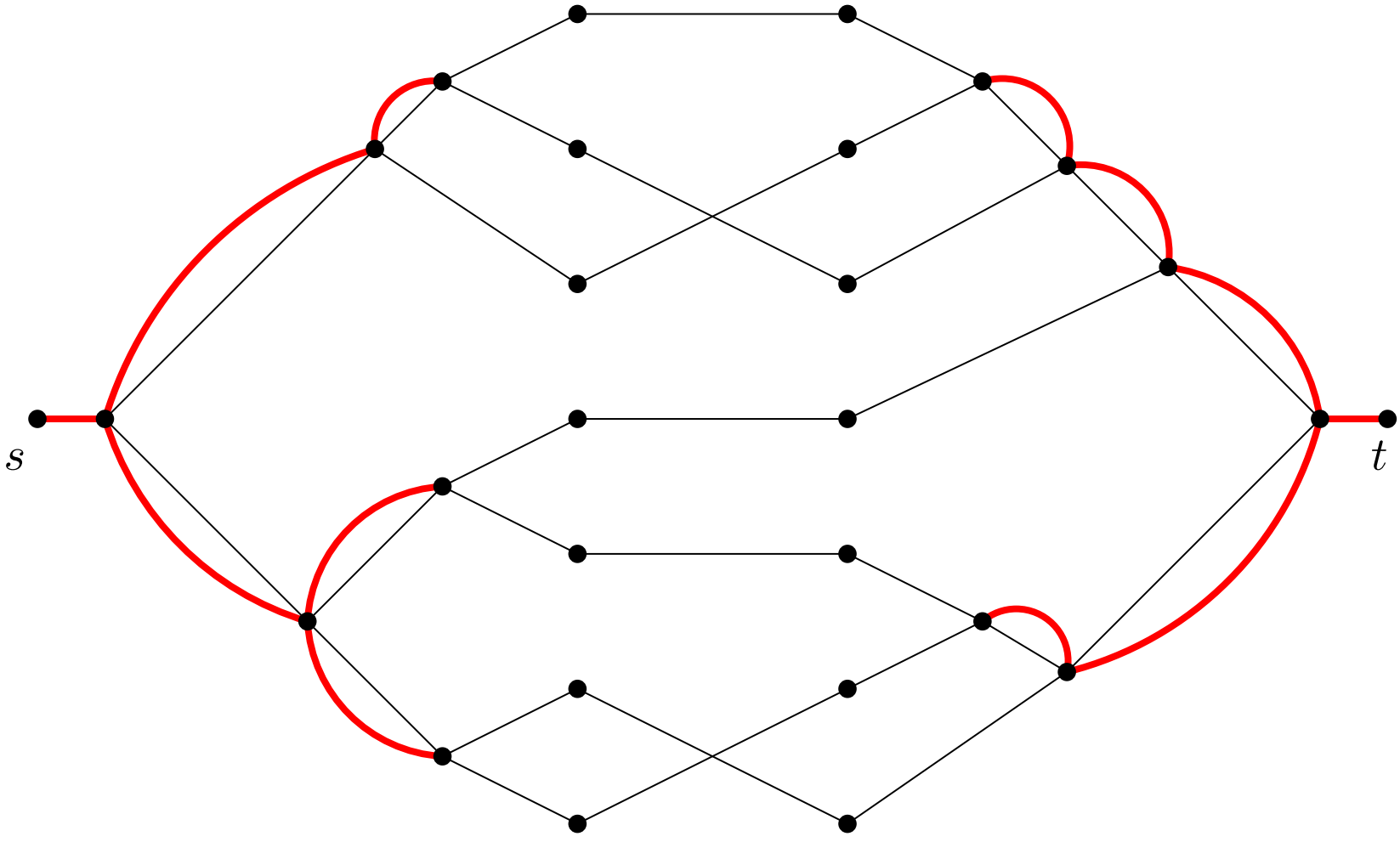
The construction (rooted case)



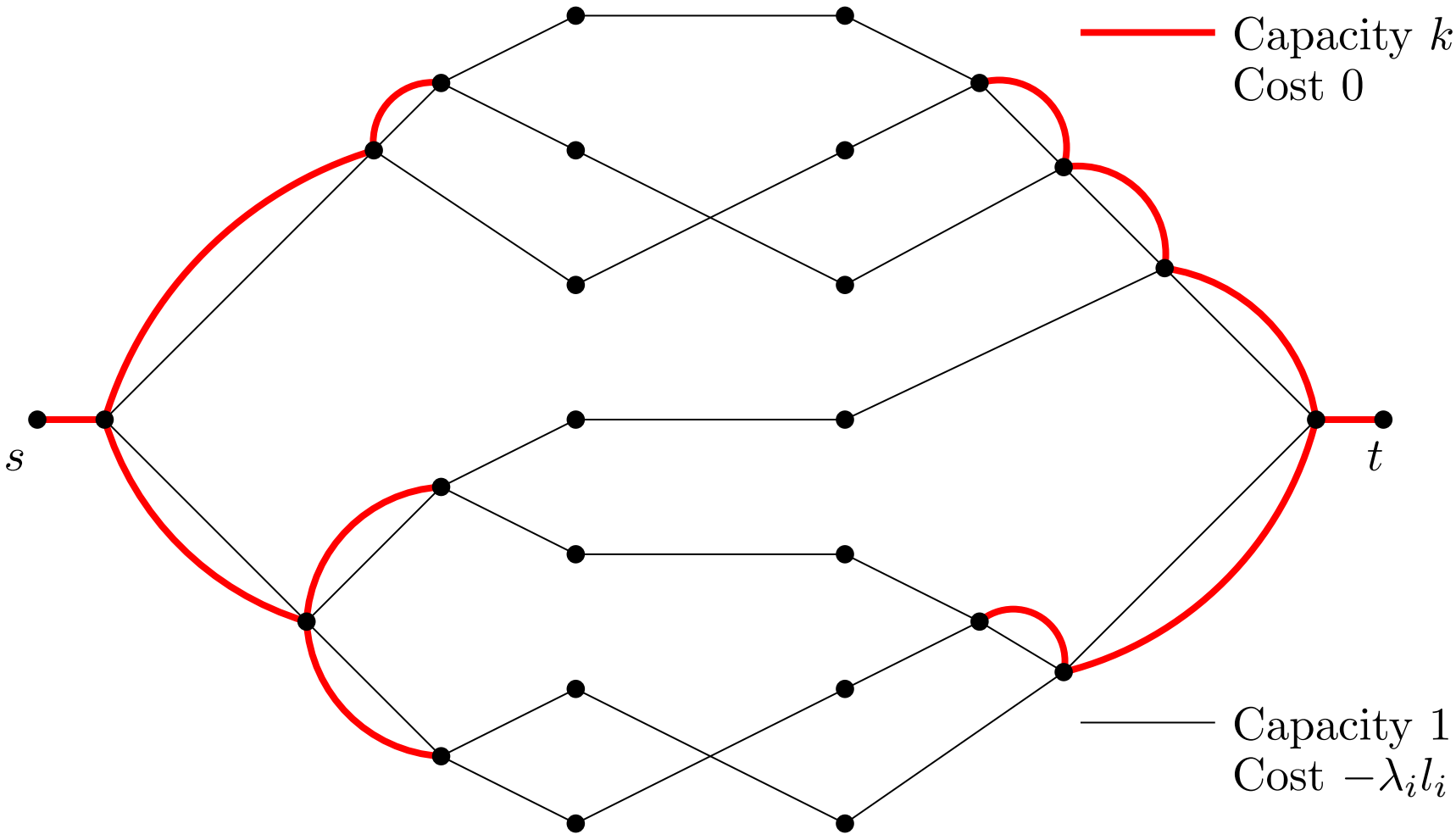
The construction (rooted case)



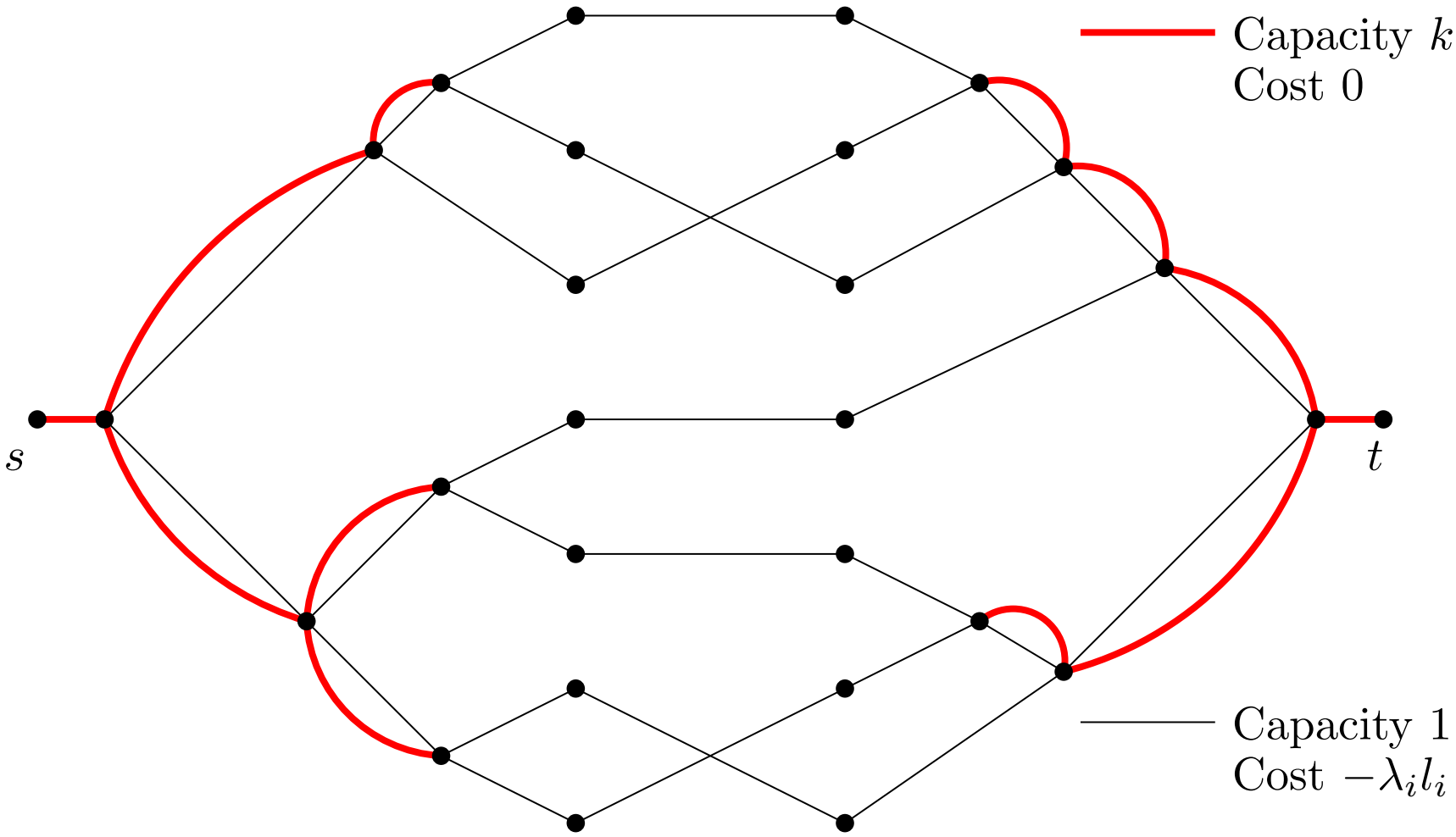
The construction (rooted case)



The construction (rooted case)

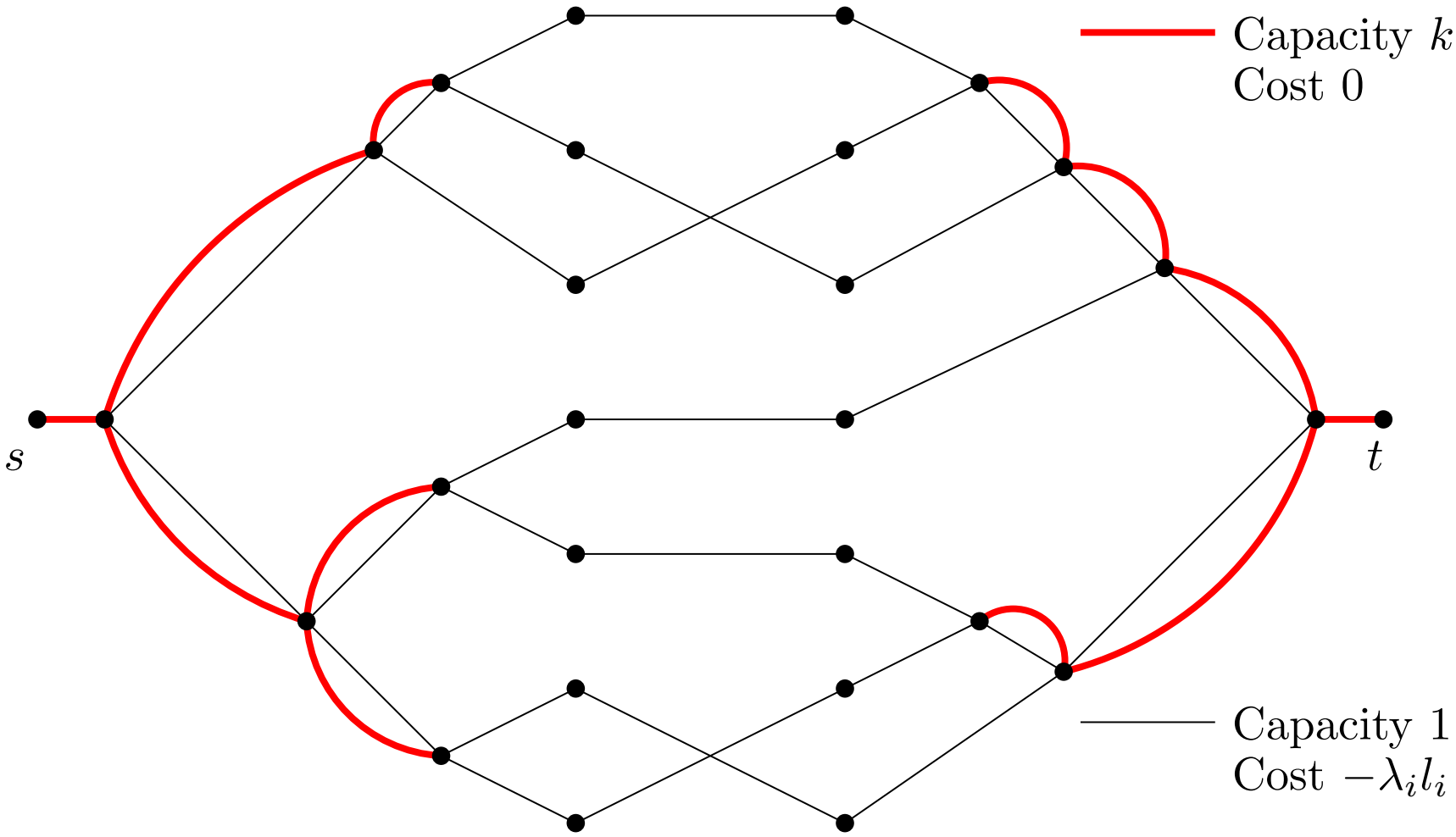


The construction (rooted case)



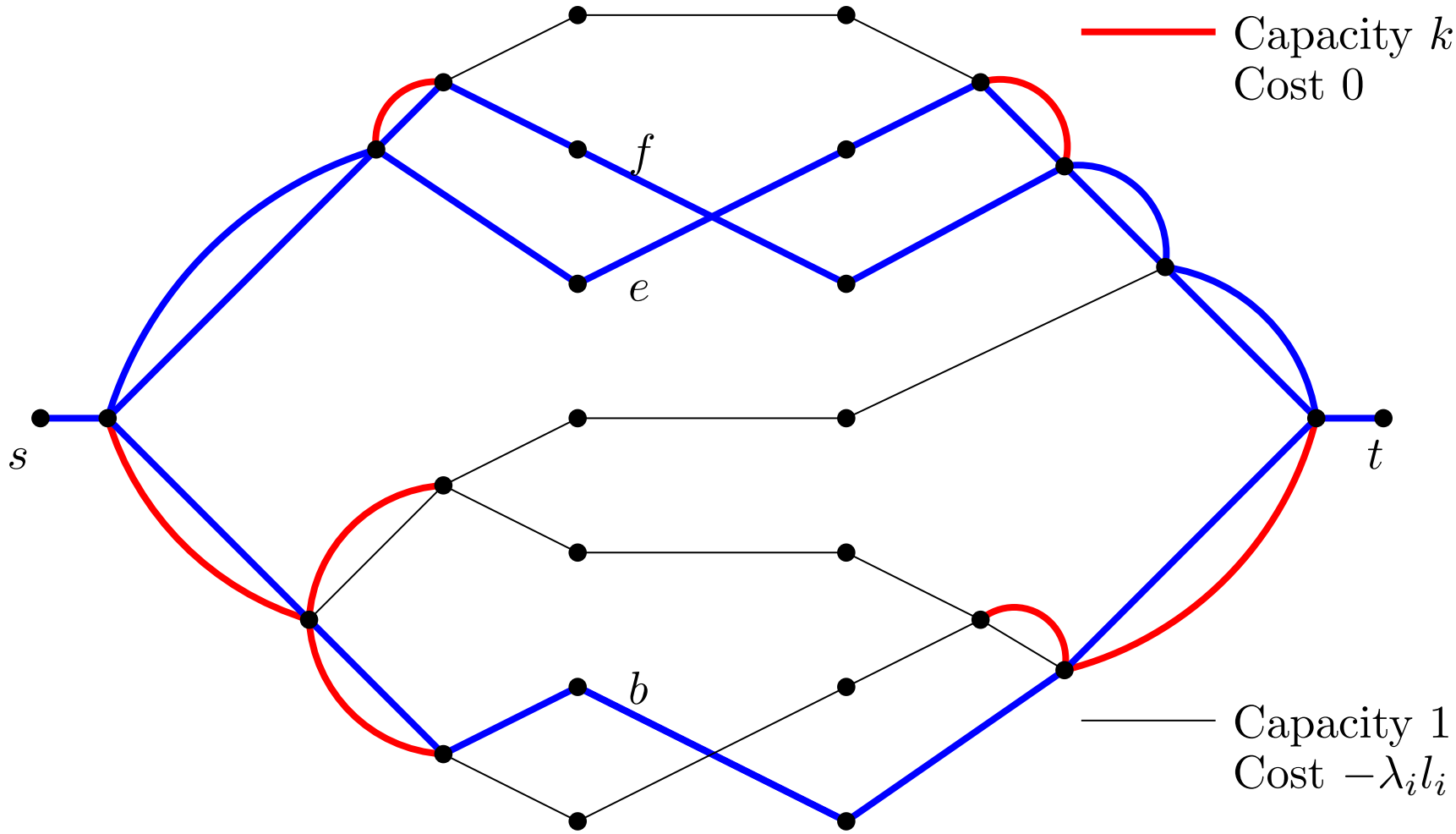
- We push a flow of k units from s to t.

The construction (rooted case)



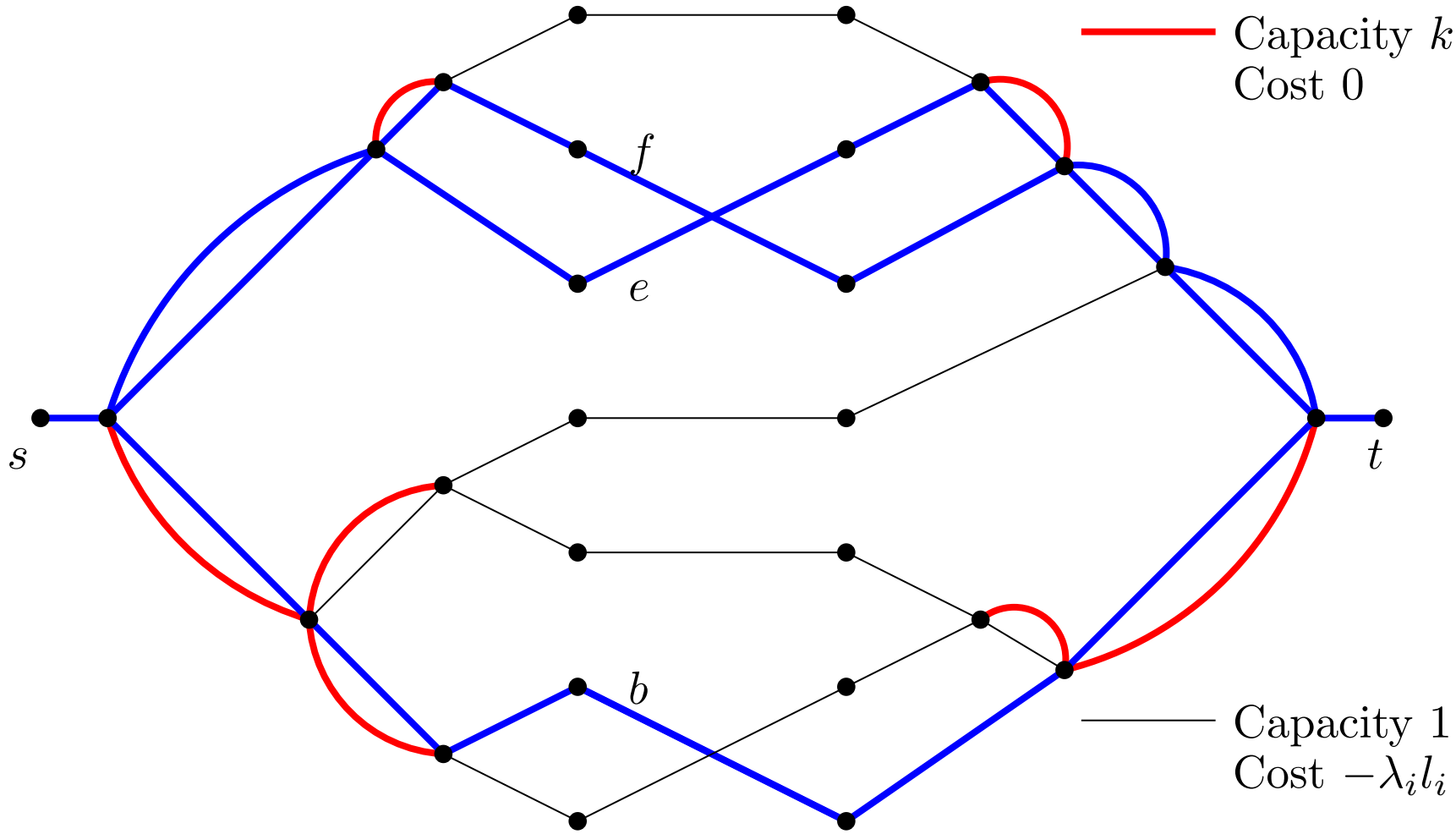
- Flow of k units 'picks out' k taxa (Y say).

The construction (rooted case)



- Flow of k units 'picks out' k taxa (Y say).

The construction (rooted case)



- Minimum cost for these taxa is $-PD_{T_1}(Y) - PD_{T_2}(Y)$

Unrooted case

For each $x \in X$:

- Root T_1 and T_2 at x.
- Use the algorithm for the rooted version and $k - 1$.
- Obtain solution S_x.

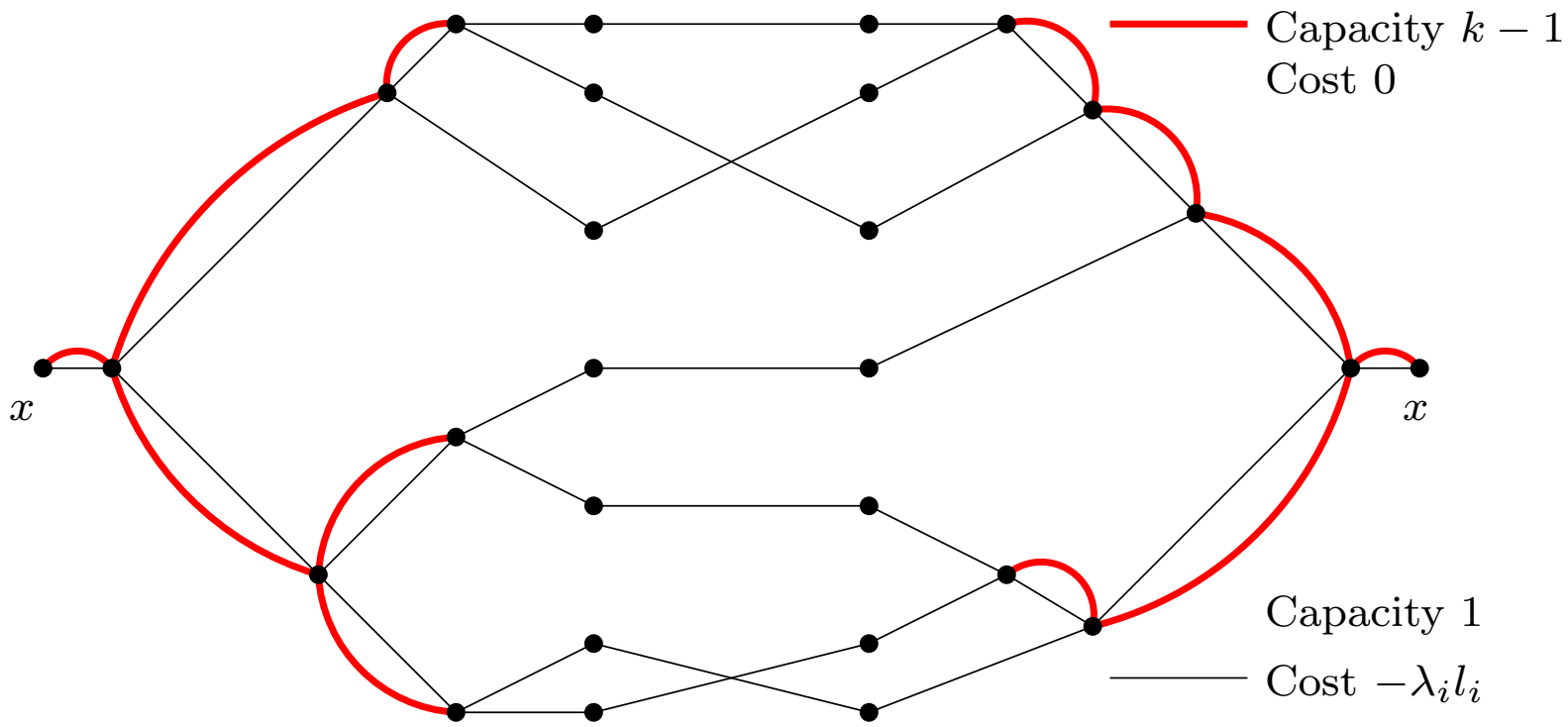
Take the best solution $\max_{x \in X} S_x$.

Unrooted case

For each $x \in X$:

- Root T_1 and T_2 at x.
- Use the algorithm for the rooted version and $k-1$.
- Obtain solution S_x.

Take the best solution $\max_{x \in X} S_x$.



Open problems

Away from multiple trees to more general split systems:

- polynomial algorithm for circular split systems [Minh *et al.*],
- polynomial algorithm for affine split systems [Spillner *et al.*],
- weakly compatible split systems?

More realism:

- survival probabilities,
- budgets and costs,
- nature reserves,
- political agenda.