Topic 7: Disjoint Paths Problem

Input: A directed graph: G

k pairs of nodes: $(s_1, t_1), (s_2, t_2), (s_3, t_3), ..., (s_k, t_k)$

* Each pair is a routing request for a path from s_i to t_i.

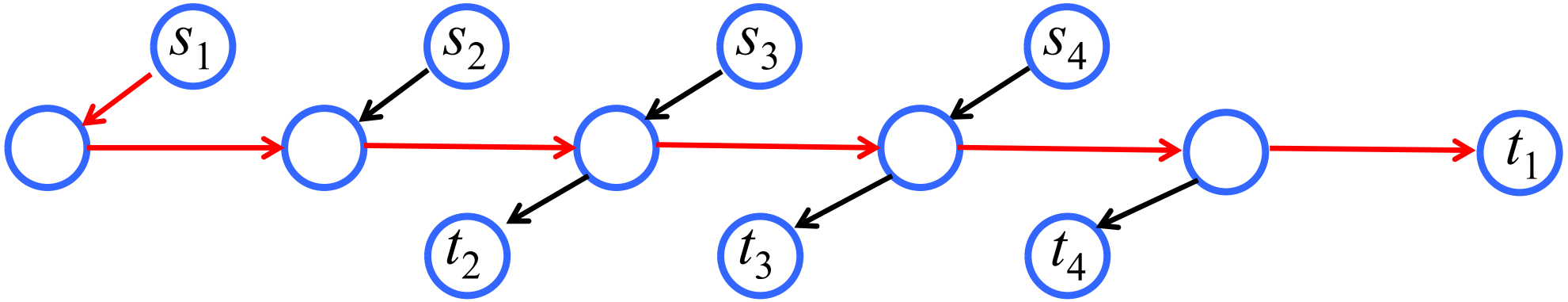
An integer capacity of each edge: c

Constraint: Each edge cannot be used by more than c paths.

Objective: Maximization of the number of satisfied paths

Output: Satisfied paths.

Let I be a set of satisfied paths: $I \subseteq \{1, 2, ..., k\}$. The objective is to maximize $|I|$. The optimal solution is denoted by I^*.



Greedy Disjoint Paths ($c = 1$)

Basic Idea ($c = 1$): Select the shortest path.

procedure GREEDY-DISJOINT-PATHS
  Set $I = \emptyset$
  until no new path can be found
    Let P_i be the shortest path (if one exists) that is edge-disjoint from previously selected paths, and connects some unconnected (s_i, t_i) pair
    Add i to I and select path P_i to connect s_i to t_i
  end until
end procedure

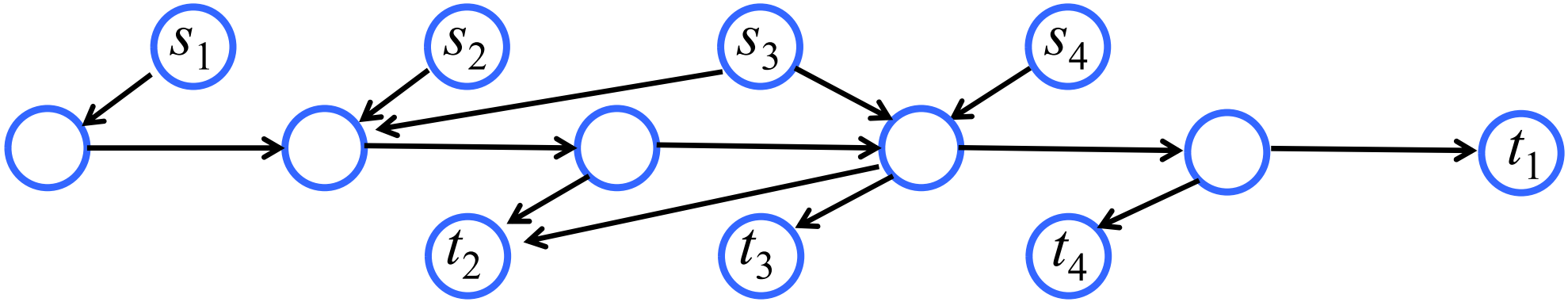
satisfies the capacity condition

The greedy algorithm is a $(2\sqrt{m}+1)$- approximation algorithm where m is the number of edges ($m = |E|$).

$$|I^*| \leq (2\sqrt{m}+1)\,|I|$$

$$\frac{|I^*|}{(2\sqrt{m}+1)} \leq |I|$$

Example 1: $|I^*| =$ ____ (Optimal), $|I| =$ ____ (Greedy)



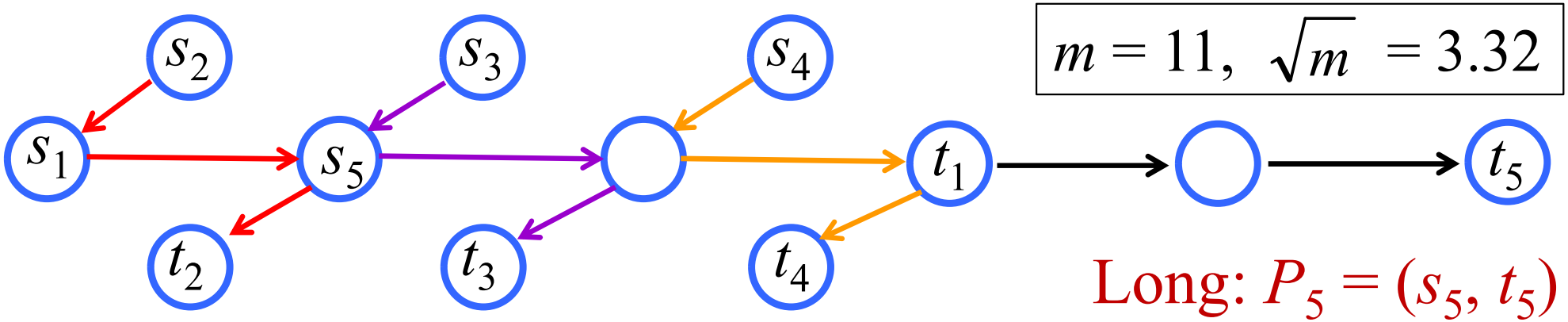
The greedy algorithm is a $(2\sqrt{m}+1)$- approximation algorithm where m is the number of edges ($m = |E|$). $|I^*| \leq (2\sqrt{m}+1)|I|$

Basic Idea of the proof (I^*: Optimal solution, I: Obtained solution)

1. Divide the paths in I into I_S (short paths) and $I - I_S$ (long paths).
2. Divide the paths in I^* into I_S^* (short paths) and $I - I_S^*$ (long paths).

A path P is short when $|P| < \sqrt{m}$,
i.e., P is short when P has less than $\sqrt{m}$ edges.

If $|P| \geq \sqrt{m}$, P is long.

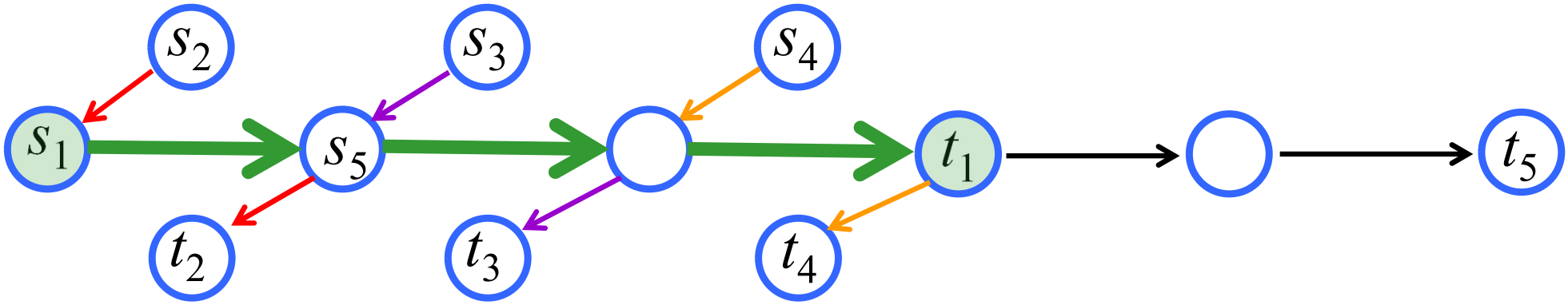


The greedy algorithm is a $(2\sqrt{m}+1)$- approximation algorithm where m is the number of edges ($m = |E|$). $|I^*| \leq (2\sqrt{m}+1)|I|$

Basic Idea of the proof (I^*: Optimal solution, I: Obtained solution)

1. Divide the paths in I into I_S (short paths) and $I - I_S$ (long paths).
2. Divide the paths in I^* into I_S^* (short paths) and $I - I_S^*$ (long paths).
3. Consider the paths in I_S^* but not in I_S. Those paths are blocked by some other short paths in I_S.

$$I = I_S = \{P_1\}$$
$$I^* = I_S^* = \{P_2, P_3, P_4\}$$



The greedy algorithm is a $(2\sqrt{m}+1)$- approximation algorithm where m is the number of edges ($m = |E|$). $|I^*| \leq (2\sqrt{m}+1)|I|$

Basic Idea of the proof (I^*: Optimal solution, I: Obtained solution)

1. Divide the paths in I into I_S (short paths) and $I - I_S$ (long paths).
2. Divide the paths in I^* into I_S^* (short paths) and $I - I_S^*$ (long paths).
3. Consider the paths in I_S^* but not in I_S. Those paths are blocked by some other short paths in I_S.
4. The total number of long paths is not larger than $\sqrt{m}$ (e.g., if $m = |E| = 36$, each long path needs at least 6 edges. Thus, we cannot have more than 6 long paths).

A path P is short when $|P| < \sqrt{m}$,
i.e., P is short when P has less than $\sqrt{m}$ edges.

If $|P| \geq \sqrt{m}$, P is long.

The greedy algorithm is a $(2\sqrt{m}+1)$- approximation algorithm where m is the number of edges ($m = |E|$). $|I^*| \leq (2\sqrt{m}+1)|I|$

Proof.

Let I^* = set of pairs connected in an optimal solution, and P_i^* for $i \in I^*$ be the selected paths.

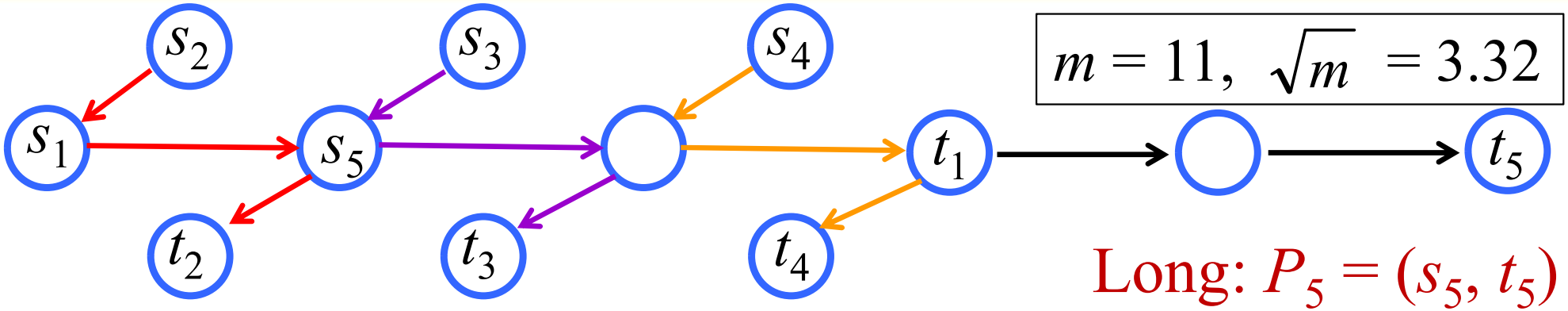
Let I = the set of pairs selected by the algo, and P_i for $i \in I$ be the corresponding paths.

Call a path *long* if it has at least $\sqrt{m}$ edges, and *short* otherwise.

Let I_s^* be indices in I^* with short P_i^*, and similarly define I_s for I.

G has m edges, and long paths use at least $\sqrt{m}$ edges, so there can be at most $\sqrt{m}$ long paths in I^*.

Now consider short paths in I^*. In order for I^* to be much larger than I, there must be many connected pairs that are in I^* but not in I.



Proof.

Consider pairs connected by the optimum by a short path, but not connected by our algo.

Since P_i^* connecting s_i and t_i in I^* is short, the greedy algo would have picked it before picking long paths if it was available.

Since the algo did not pick it, one of the edges e along P_i^* must occur in a path P_j selected earlier by the algo.

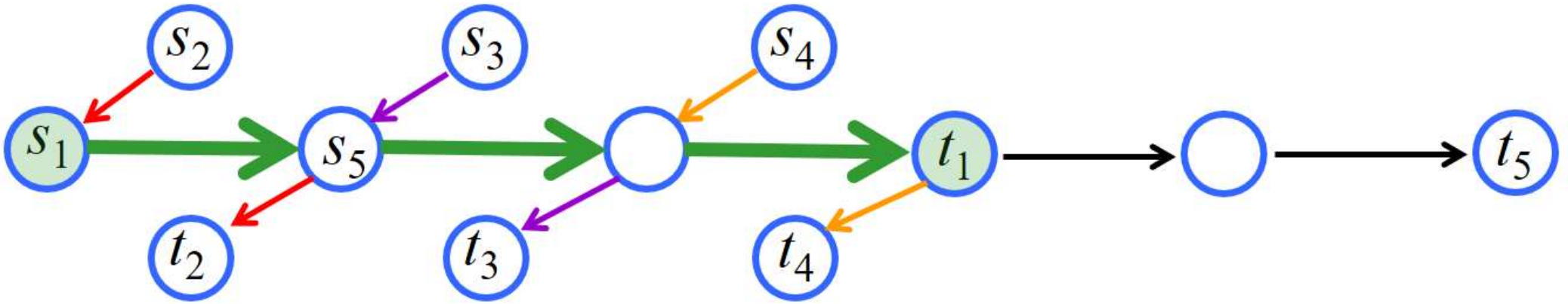
We say edge e blocks path P_i^*.

Lengths of paths selected by the algo are monotone increasing.

P_j was selected before P_i^*, and so must be shorter: $|P_j| \leq |P_i^*| \leq \sqrt{m}$, so P_j is short.

Paths are edge-disjoint, so each edge in a path P_j can block at most one path P_i^*.

So each short path P_j blocks at most $\sqrt{m}$ paths in the optimal solution,



Proof.

Consider pairs connected by the optimum by a short path, but not connected by our algo.

Since P_i^* connecting s_i and t_i in I^* is short, the greedy algo would have picked it before picking long paths if it was available.

Since the algo did not pick it, one of the edges e along P_i^* must occur in a path P_j selected earlier by the algo.

We say edge e blocks path P_i^*.

Lengths of paths selected by the algo are monotone increasing.

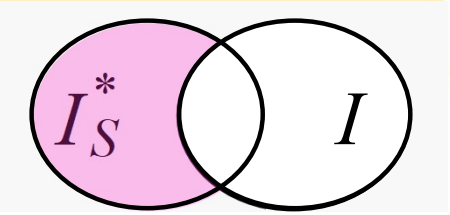
P_j was selected before P_i^*, and so must be shorter: $|P_j| \leq |P_i^*| \leq \sqrt{m}$, so P_j is short.

Paths are edge-disjoint, so each edge in a path P_j can block at most one path P_i^*.

So each short path P_j blocks at most $\sqrt{m}$ paths in the optimal solution, and so

$$|I_s^* - I| \leq \sum_{j \in I_s} |P_j| \leq |I_s|\sqrt{m} \qquad (*)$$

$I_S^* - I := I_S^* - (I_S^* \cap I)$



Proof.

I^* consists of three kinds of paths:

- long paths, of which there are at most $\sqrt{m}$;
- paths that are also in I; and
- short paths that are not in I, fewer than $|I_s|\sqrt{m}$ by (*).

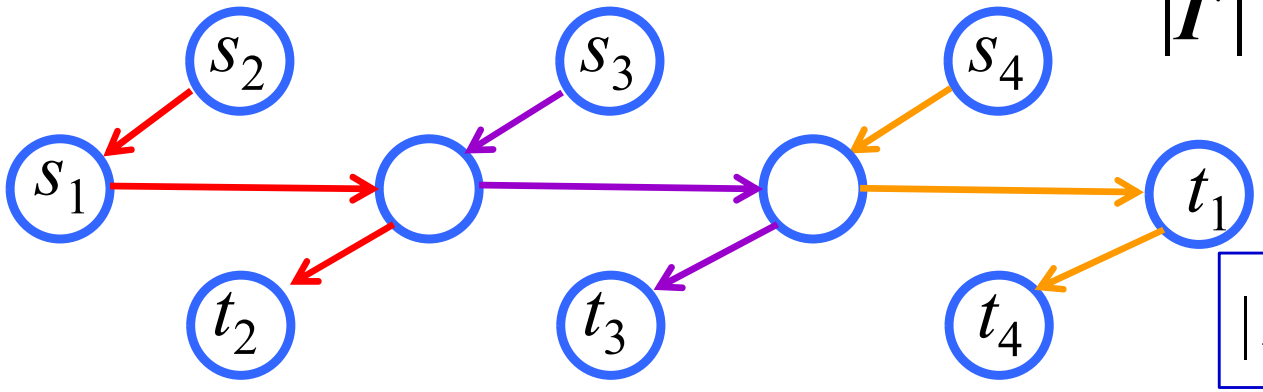
Note that $|I| \geq 1$ whenever at least one pair can be connected.
So $|I^*| \leq \sqrt{m} + |I| + |I_s^* - I| \leq \sqrt{m} + |I| + \sqrt{m}|I_s| \leq$
$\sqrt{m}|I| + |I| + \sqrt{m}|I| = (2\sqrt{m} + 1)|I|$. □

Exercise 13-1 (*c*=1):

(1) Create an example where the optimal solution is always obtained by the greedy algorithm independent of the selection order of the shortest paths. (2) Create another example where $|I^*| \geq \sqrt{m}\,|I|$ always holds independent of the selection order of the shortest paths. (3) Create another example where $|I^*| > \sqrt{m}\,|I|$ holds depending on the selection order of the shortest paths.

[Example: $m = 9$] This example does not satisfy the exercise conditions. All paths have the same length (length 3).

$|I^*| = 3$, $|I| = 1$ or $|I| = 3$.



Depending on the order,

$$|I^*| = \sqrt{m}\,|I| = \sqrt{9}\,|I| = 3|I|$$

Topic 7: Disjoint Paths Problem (c > 1)

Input: A directed graph: $G = (V, E)$, $\boldsymbol{m} = |E|$

k pairs of nodes: $(s_1, t_1), (s_2, t_2), (s_3, t_3), ..., (s_k, t_k)$

* Each pair is a routing request for a path from s_i to t_i.

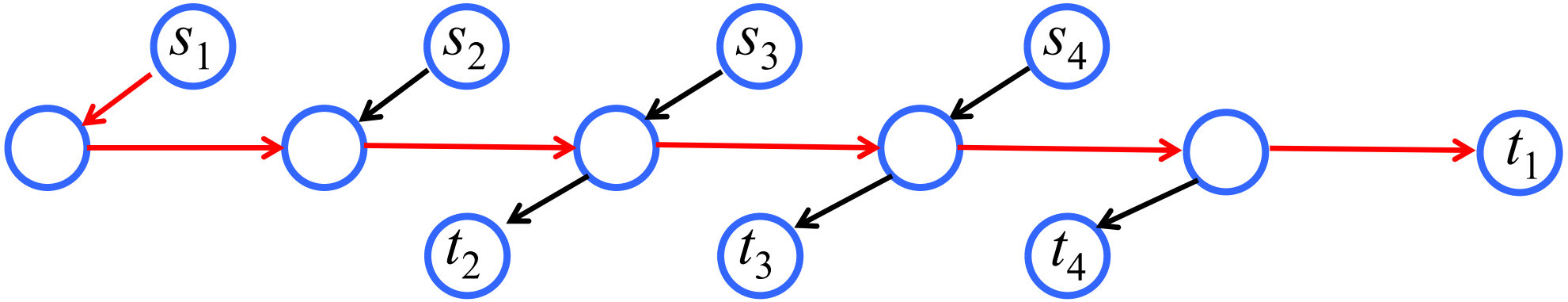
An integer capacity of each edge: c

Constraint: Each edge cannot be used by more than ***c* paths**.

Objective: Maximization of the number of satisfied paths

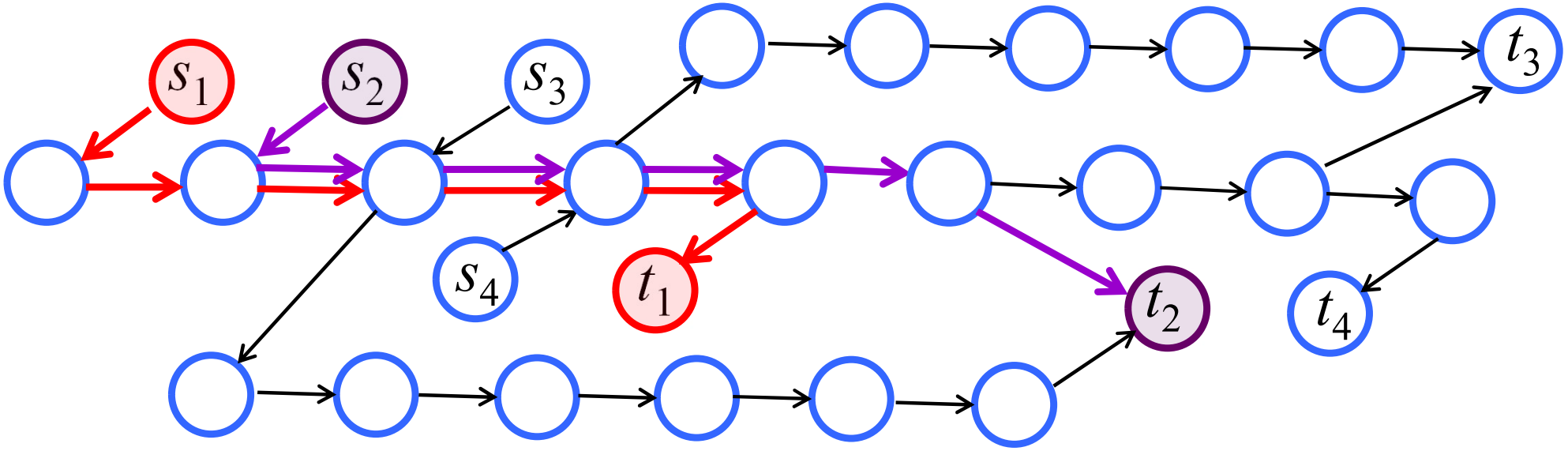
Output: Satisfied paths.

Let I be a set of satisfied paths: $I \subseteq \{1, 2, ..., k\}$. The objective is to maximize $|I|$. The optimal solution is denoted by I^*.

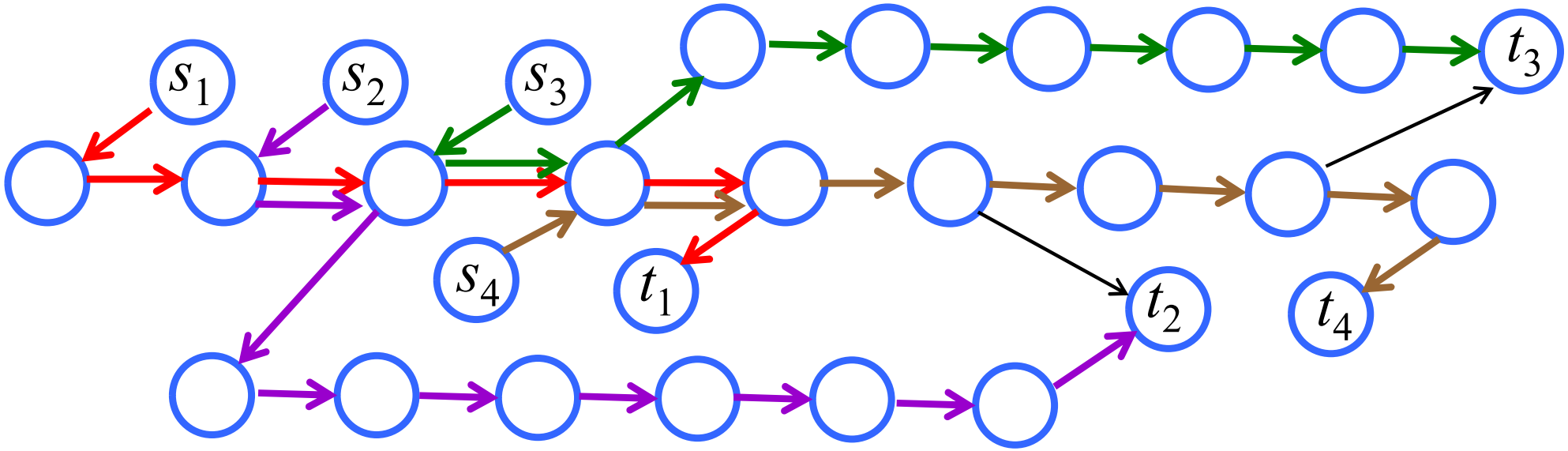


Greedy Disjoint Paths with $c > 1$

Example: $c = 2$ (Each edge can be used twice).



Point: To avoid concentrating on any single edge.



Greedy Disjoint Paths with $c > 1$ $(m = |E|)$

Pricing Method ($c > 1$) ($\beta = m^{1/(c+1)}$)

Edges are viewed as more expensive if they have already been used (since they have less capacity). That is, the cost of each edge is increased when it is used by a path. Its initial cost is 1. After the use by a single path, its cost is β. After the second use (i.e., after the use by two paths), its cost is β^2. After the third ...

The cost of each edge is handled as its length.

ℓ_e: the length of an edge e.

$\ell(P)$: the length of a path P : $\ell(P) = \sum_{e \in P} \ell_e$.

The length is updated whenever a new path is selected.

Greedy Paths with Capacity

Basic Idea ($c > 1$): Increase the length of each edge used by a newly added path and select the shortest path. $\beta = m^{1/(c+1)}$

procedure GREEDY-PATHS-WITH-CAPACITY
Set $I = \emptyset$
Set edge length $\ell_e = 1$ for all $e \in E$
until no new path can be found
Let P_i be the shortest path (if one exists) such that adding P_i to the selected set of paths does not use any edge more than c times, and P_i connects some unconnected (s_i, t_i) pair
Add i to I and select P_i to connect s_i to t_i
Multiply the length of all edges along P_i by β
end until
end procedure

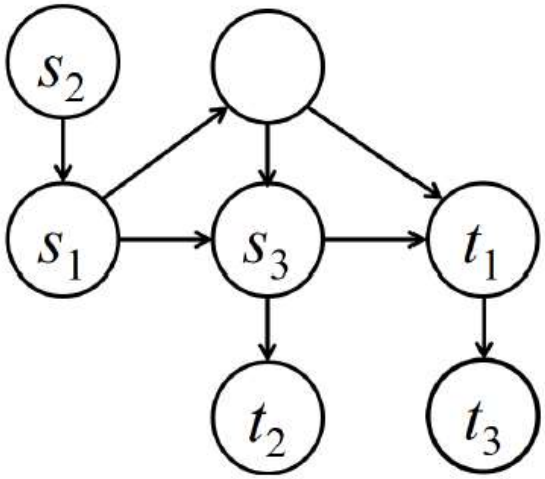
satisfies the capacity condition

I : The set of satisfied paths by this algorithm
I^* : Optimal solution

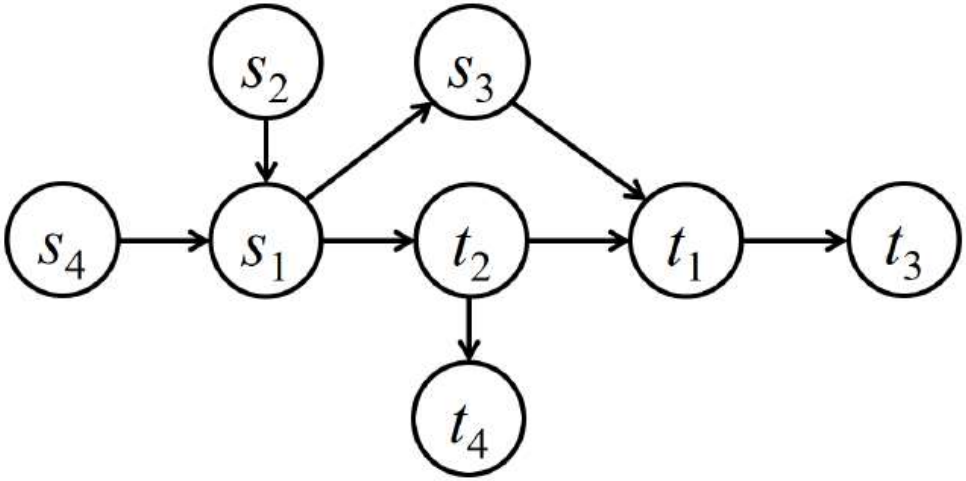
(4.1) Show all solutions of Graph 4.1 which can be obtained by the greedy algorithm for $c = 1$. Note that each solution is a set of paths: $\{(s_i, t_i), \ldots, (s_j, t_j)\}$.

(4.2) Show all solutions of Graph 4.2 which can be obtained by the greedy algorithm for $c = 2$. Note that each solution is a set of paths: $\{(s_i, t_i), \ldots, (s_j, t_j)\}$.

(4.1) Graph 4.1 ($c = 1$)



(4.2) Graph 4.2 ($c = 2$)



Solution Quality

The greedy algorithm with $\beta = m^{1/3}$ is a $(4m^{1/3}+1)$-approximation algorithm for $c = 2$ where m is the number of edges ($m = |E|$).

The greedy algorithm with $\beta = m^{1/(c+1)}$ is a $(2cm^{1/(c+1)}+1)$-approximation algorithm for $c > 2$.

When $c = 1$, $|I^*| \leq (2\sqrt{m}+1)|I|$

Exercise 13-2 ($c = 2$):

(1) Create an example where the optimal solution is always obtained by the greedy algorithm (independent of the choice of the shortest paths: independent of a tie-breaking method).

(2) Create another example where the greedy algorithm solution $|I|$ is close to $|I^*|/(4m^{1/3}+1)$ depending on the choice of the shortest paths (i.e., depending on a tie-breaking method).

The greedy algorithm with $\beta = m^{1/3}$ is a $(4m^{1/3}+1)$-approximation algorithm when $c = 2$ where m is the number of edges ($m = |E|$).

Basic Idea of the proof

1. The proof uses the length of each edge at the time when the algorithm first encounters the situation where any other short path cannot be added. The length of each edge e **at this time** is denoted by $\overline{\ell}_e$.
2. Divide the paths in I into I_S (short paths) and $I - I_S$ (long paths).
3. Divide the paths in I^* into I_S^* (short paths) and $I^* - I_S^*$ (long paths).
4. Consider a path P_i^* which is in I^* but not in I. The length of P_i^* is not shorter than β^2. If it is shorter than β^2, P_i^* can be selected. $\overline{\ell}(P_i^*) \geq \beta^2$
5. In the initial stage, the total length in the graph is m since the edge length is 1. Adding a single short path P increases the total length by increasing the edge length from 1 to β of $|P|$ edges ($< \beta \times \beta^2$).

If $|P| < \beta^2$, P is short. Otherwise P is long.

The key to the analysis in the disjoint case was to distinguish "short" and "long" paths. For the case when $c = 2$, we will consider a path P_i selected by the algorithm to be *short* if the length is less than β^2. Let I_s denote the set of short paths selected by the algorithm.

Next we want to compare the number of paths selected with the maximum possible. Let I^* be an optimal solution and P^*_i be the set of paths used in this solution. As before, the key to the analysis is to consider the edges that block the selection of paths in I^*. Long paths can block a lot of other paths, so for now we will focus on the short paths in I_s. As we try to continue following what we did in the disjoint case, we immediately run into a difficulty, however. In that case, the length of a path in I^* was simply the number of edges it contained; but here, the lengths are changing as the algorithm runs, and so it is not clear how to define the length of a path in I^* for purposes of the analysis. In other words, for the analysis, when should we measure this length? (At the beginning of the execution? At the end?)

It turns out that the crucial moment in the algorithm, for purposes of our analysis, is the first point at which there are no short paths left to choose. Let $\bar{\ell}$ be the length function at this point in the execution of the algorithm; we'll use $\bar{\ell}$ to measure the length of paths in I^*. For a path P, we use $\bar{\ell}(P)$ to denote its length, $\sum_{e \in P} \bar{\ell}_e$. We consider a path P^*_i in the optimal solution I^* *short* if $\bar{\ell}(P^*_i) < \beta^2$, and *long* otherwise. Let I^*_s denote the set of short paths in I^*. The first step is to show that there are no short paths connecting pairs that are not connected by the approximation algorithm.

we do not have to worry about explicitly enforcing the requirement that each edge be used by at most $c = 2$ paths:

Theorem

(11.17) Consider a source-sink pair $i \in I^*$ that is not connected by the approx algo; i.e. $i \notin I$. Then $\bar{\ell}(P_i^*) \geq \beta^2$.

Proof.

As long as short paths are being selected, any edge e considered for selection by a third path would already have length $\ell_e = \beta^2$ and hence be long.

Consider the state of the algo with length $\bar{\ell}$. We can imagine the algo running up to this point without caring about the limit of c.

Since s_i, t_i of P_i^* are not connected by the algo, and since there are no short paths left when the length function reaches $\bar{\ell}$, it must be that path P_i^* has length of at least β^2 as measured by $\bar{\ell}$. □

(11.17) *Consider a source-sink pair $i \in I^*$ that is not connected by the approximation algorithm; that is, $i \notin I$. Then $\bar{\ell}(P^*_i) \geq \beta^2$.*

Proof. As long as short paths are being selected, we do not have to worry about explicitly enforcing the requirement that each edge be used by at most $c = 2$ paths: any edge e considered for selection by a third path would already have length $\ell_e = \beta^2$, and hence be long. Consider the state of the algorithm with length $\bar{\ell}$. By the argument in the previous paragraph, we can imagine the algorithm having run up to this point without caring about the limit of c; it just selected a short path whenever it could find one. Since the endpoints s_i, t_i of P^*_i are not connected by the greedy algorithm, and since there are no short paths left when the length function reaches $\bar{\ell}$, it must be the case that path P^*_i has length at least β^2 as measured by $\bar{\ell}$. ▪

The analysis in the disjoint case used the fact that there are only m edges to limit the number of long paths. Here we consider length $\bar{\ell}$, rather than the number of edges, as the quantity that is being consumed by paths. Hence, to be able to reason about this, we will need a bound on the total length in the graph $\sum_e \bar{\ell}_e$. The sum of the lengths over all edges $\sum_e \ell_e$ starts out at m (length 1 for each edge). Adding a short path to the solution I_s can increase the length by at most β^3, as the selected path has length at most β^2, and the lengths of the edges are increased by a β factor along the path. This gives us a useful comparison between the number of short paths selected and the total length.

Finding the total length in the graph $\sum_e \bar{\ell}_e$

$\sum_e \bar{\ell}_e$ starts out at m (length 1 for each edge). Adding a short path to the solution I_s can increase the length by at most β^3, as the selected path has length at most β^2 (for $c = 2$), and the lengths of the edges are increased by a factor of β along the path. Thus,

Theorem

(11.18) The set I_s of short paths selected by the approx algo, and the lengths $\bar{\ell}$, satisfy the relation $\sum_e \bar{\ell}_e \leq \beta^3 |I_s| + m$.

Finally, we prove an approximation bound for this algorithm. We will find that even though we have simply increased the number of paths allowed on each edge from 1 to 2, the approximation guarantee drops by a significant amount that essentially incorporates this change into the exponent: from $O(m^{1/2})$ down to $O(m^{1/3})$.

Theorem

(11.19) GREEDY-PATHS-WITH-CAPACITY, *using* $\beta = m^{\frac{1}{3}}$, *is a* $(4m^{\frac{1}{3}} + 1)$*-approx algo for capacity* $c = 2$.

Proof.

By (11.17), we have $\bar{\ell}(P_i^*) \geq \beta^2 \; \forall i \in I^* - I$.

$$|I^*| \leq (2\sqrt{m}+1)\,|I|$$

Summing over all paths in $I^* - I$, $\sum_{i \in I^* - I} \bar{\ell}(P_i^*) \geq \beta^2 |I^* - I|$.

On the other hand, each edge is used by at most two paths in I^*, so $\sum_{i \in I^* - I} \bar{\ell}(P_i^*) \leq \sum_{e \in E} 2\bar{\ell}_e$.

3

Combining these with (11.18): $\beta^2 |I^*| \leq \beta^2 |I^* - I| + \beta^2 |I| \leq \sum_{i \in I^* - I} \bar{\ell}(P_i^*) + \beta^2 |I| \leq \sum_{e \in E} 2\bar{\ell}_e + \beta^2 |I| \leq 2(\beta^2 |I| + m) + \beta^2 |I|$.

Dividing throughout by β^2, using $|I| \geq 1$ and setting $\beta = m^{\frac{1}{3}}$, $m = \beta^3$

$|I^*| \leq (4m^{\frac{1}{3}} + 1)|I|$.

$$\beta^2 |I^*| \leq 2(\beta^3 |I| + m|I|) + \beta^2 |I|$$

□

Exercise 13-1 (c=1):

(1) Create an example where the optimal solution is always obtained by the greedy algorithm independent of the selection order of the shortest paths. (2) Create another example where $|I^*| \geq \sqrt{m}\ |I|$ always holds independent of the selection order of the shortest paths. (3) Create another example where $|I^*| > \sqrt{m}\ |I|$ holds depending on the selection order of the shortest paths.

Exercise 13-2 (c = 2):

(1) Create an example where the optimal solution is always obtained by the greedy algorithm (independent of the choice of the shortest paths: independent of a tie-breaking method).

(2) Create another example where the greedy algorithm solution $|I|$ is close to $|I^*|/(4m^{1/3}+1)$ depending on the choice of the shortest paths (i.e., depending on a tie-breaking method).