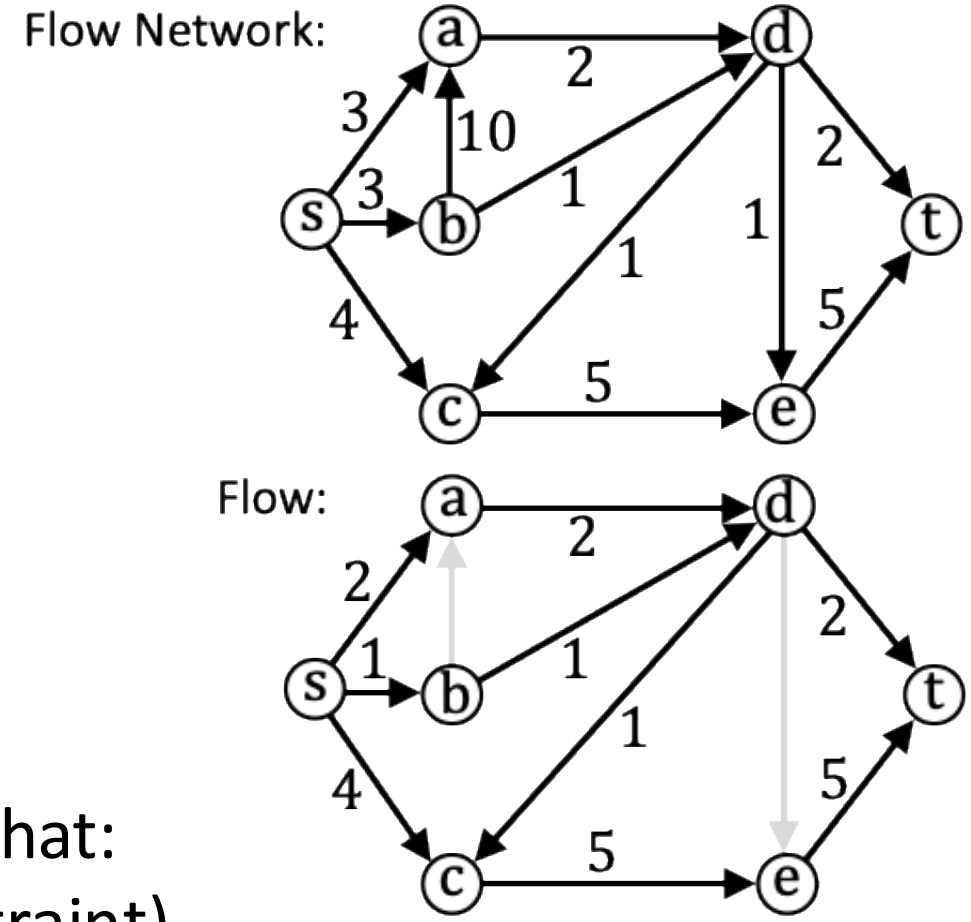
Flow Networks
CSCI 432

Maximum Flow Problem:
Given a flow network, find the maximum possible value of flow.



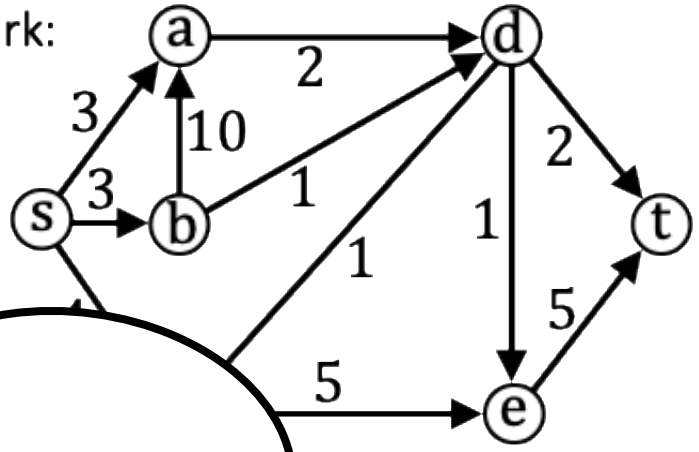
Flow Network:

- Directed-edge graph, $G = (V, E)$.
- Non-negative edge capacity, c_e.
- Single source, s, without input edges.
- Single sink, t, without output edges.

An $s - t$ flow is a function f: $E \rightarrow \mathbb{R}^+$ such that:

- $0 \leq f(e) \leq c_e, \forall e \in E$. (capacity constraint)
- $\sum_{e \epsilon \text{input}(v)} f(e) = \sum_{e \epsilon \text{output}(v)} f(e)\,, \forall v \in V \setminus \{s, t\}$. (conservation of flow constraint: "Everything that goes into a node has to come out, except for s and t")
- Value of flow = $val(f)$ = $\sum_{e \epsilon \text{output}(s)} f(e)$ = $\sum_{e \epsilon \text{input}(t)} f(e)$

Maximum Flow Problem:
Given a flow network, find the maximum possible value of flow.



Flow Network:

- Direct
- Non
-
-

An s -

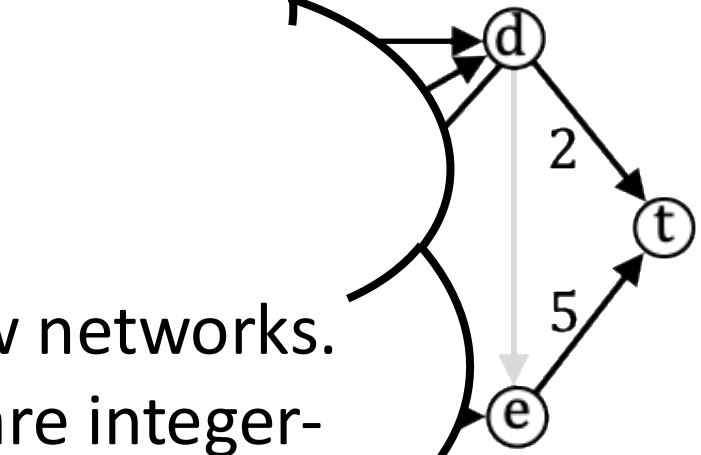
- 0
- $\sum_{e \epsilon \text{inpu}}$ (conservation of flow
es into a node has to come out, exc and t")
- Value of flow = $val(f) = \sum_{e \epsilon \text{output}(s)} f(e) = \sum_{e \epsilon \text{input}(t)} f(e)$

How do we find max flows?

We'll get to that.

Assumptions for now:

1. We can find max flows for valid flow networks.
2. If the capacities of a flow network are integer-valued, the max flow will be too.



Reductions

Reductions are a technique to solve a problem by turning into a problem we already have an algorithm for.



Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

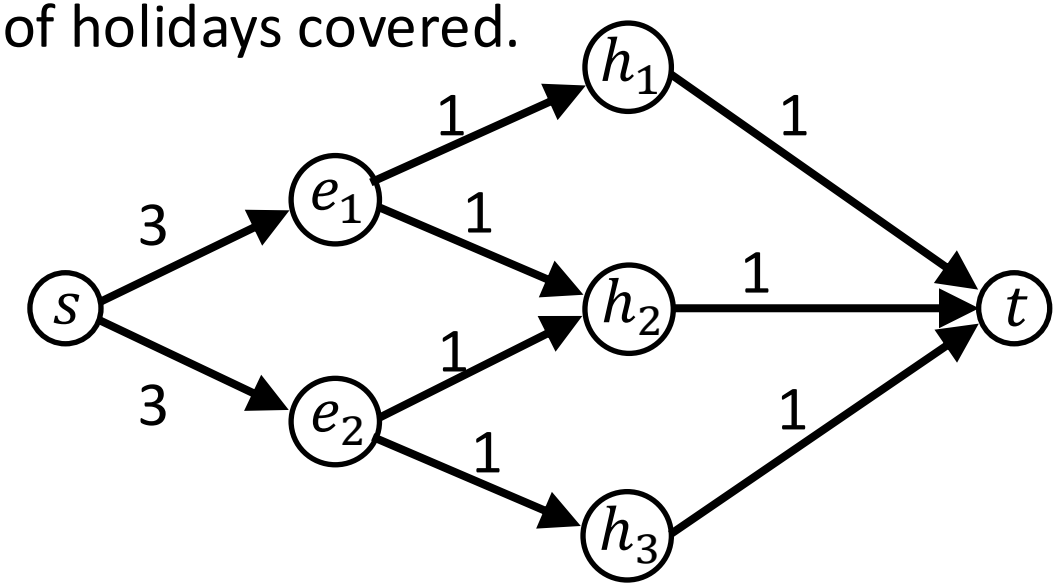
1. Build flow network:

a) Make a node for each employee, a node for each holiday, a source, and a sink.

b) Connect the source to each employee node with a capacity of 3.

c) Connect each holiday node to the sink with a capacity of 1.

d) If an employee is able to work a holiday, connect them with a capacity of 1.



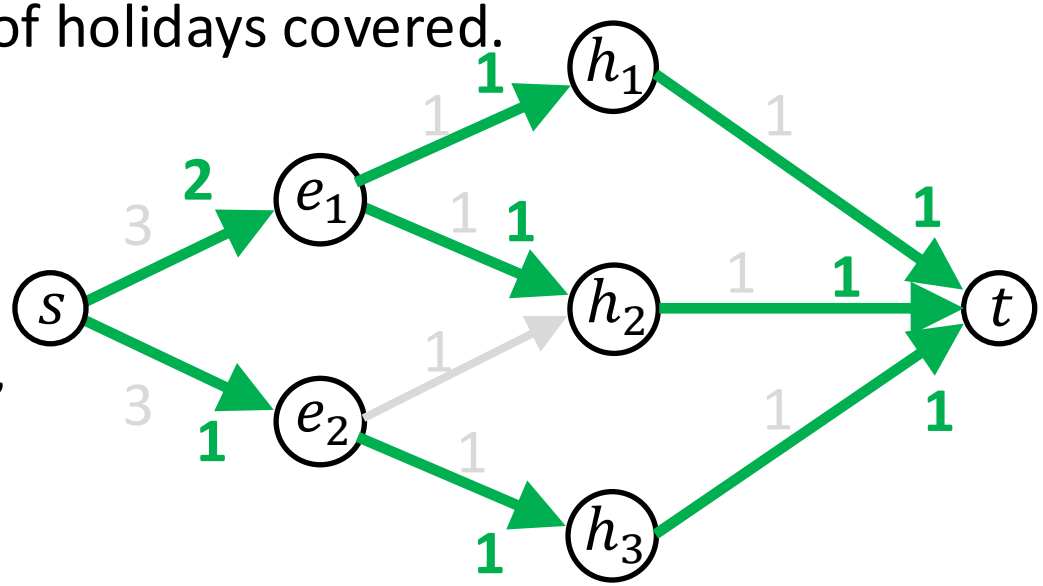
Employee	Holiday(s) they can work
e_1	h_1, h_2
e_2	h_2, h_3

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



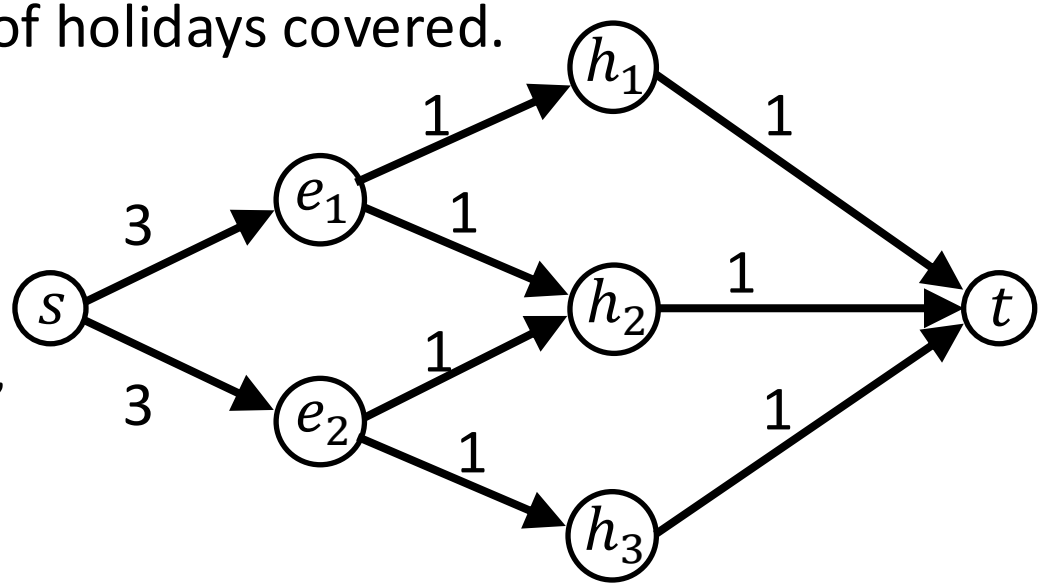
$$e_1 \rightarrow h_1$$
$$e_1 \rightarrow h_2$$
$$e_2 \rightarrow h_3$$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Running Time:

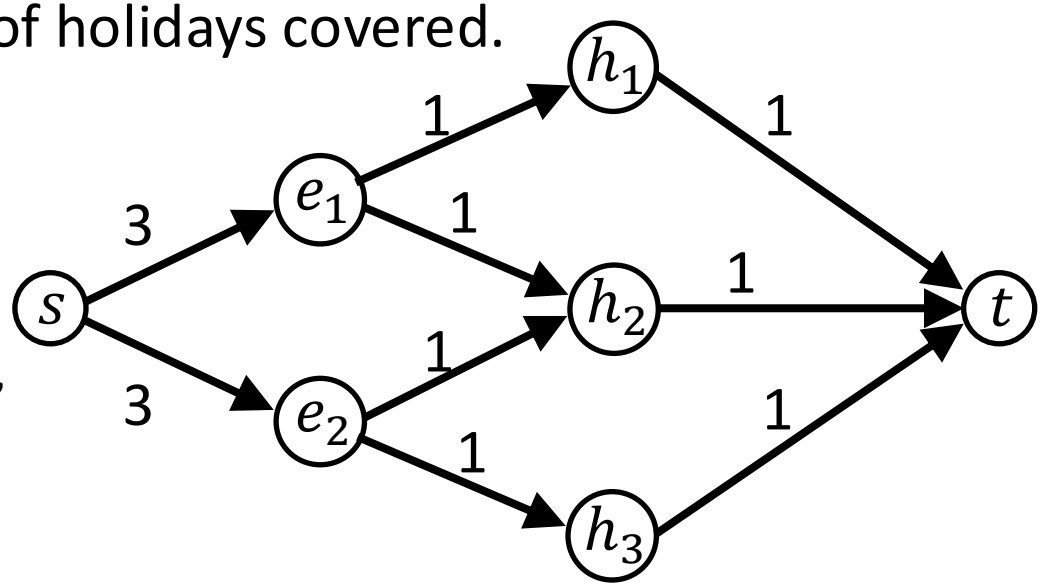
$$O(\text{time to build G}) + O(\text{Max Flow}) + O(\text{time to translate solution})$$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Running Time:

n = # employees, m = # holidays

For $G = (V, E)$, $|V| = n + m + 2$, $|E| = n + nm + m$

So, building $G \in O(nm)$.

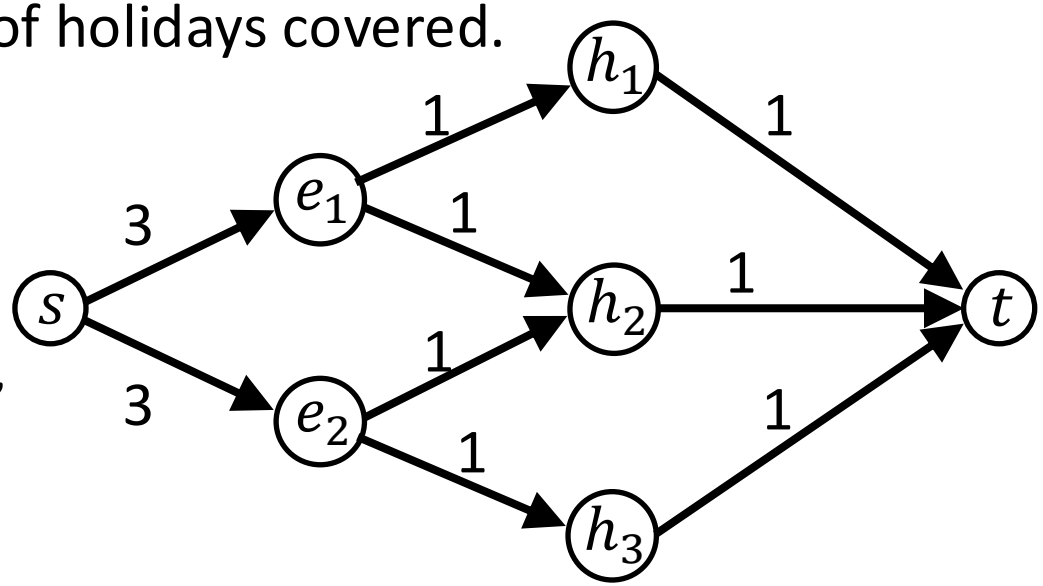
Total running time $\in O(nm) + O(\text{Max Flow}) + O(nm) \in O(nm) + O(\text{Max Flow})$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Validity:

??

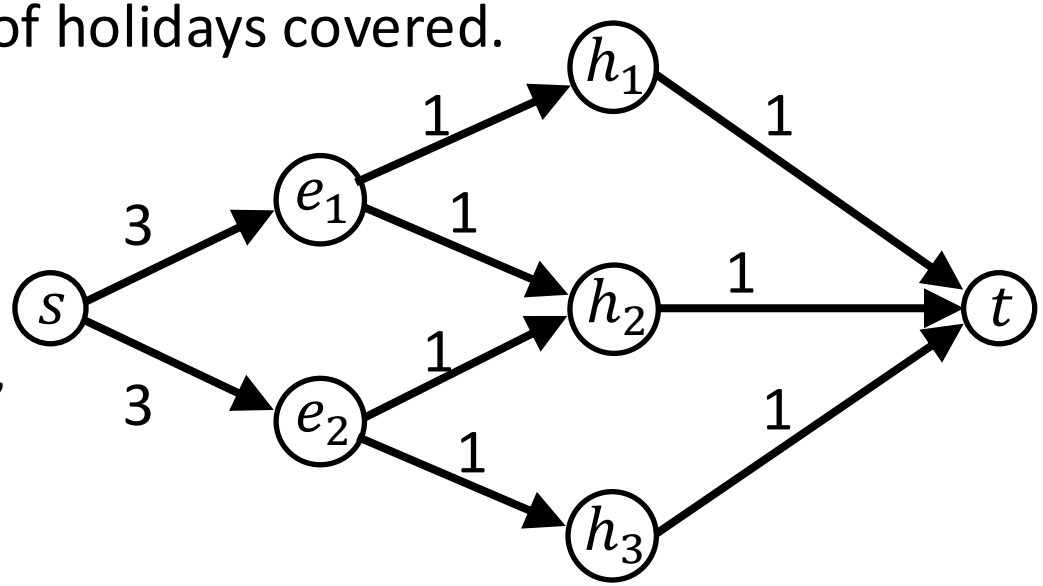
We need to show the algorithm's answer is valid for the problem, not that the max flow found is valid for the flow network we made.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Validity:

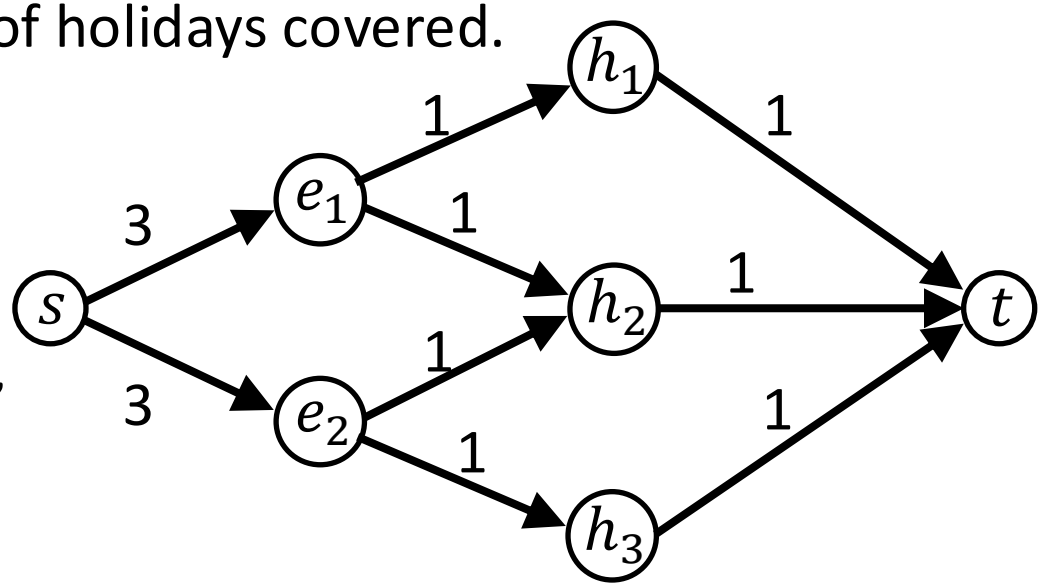
Employees cannot be assigned a holiday they are not able to work (no edges exist between incompatible pairs).

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Validity:

Employees cannot be assigned a holiday they are not able to work (no edges exist between incompatible pairs). Employees cannot be assigned to work more than 3 holidays (Why?).

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

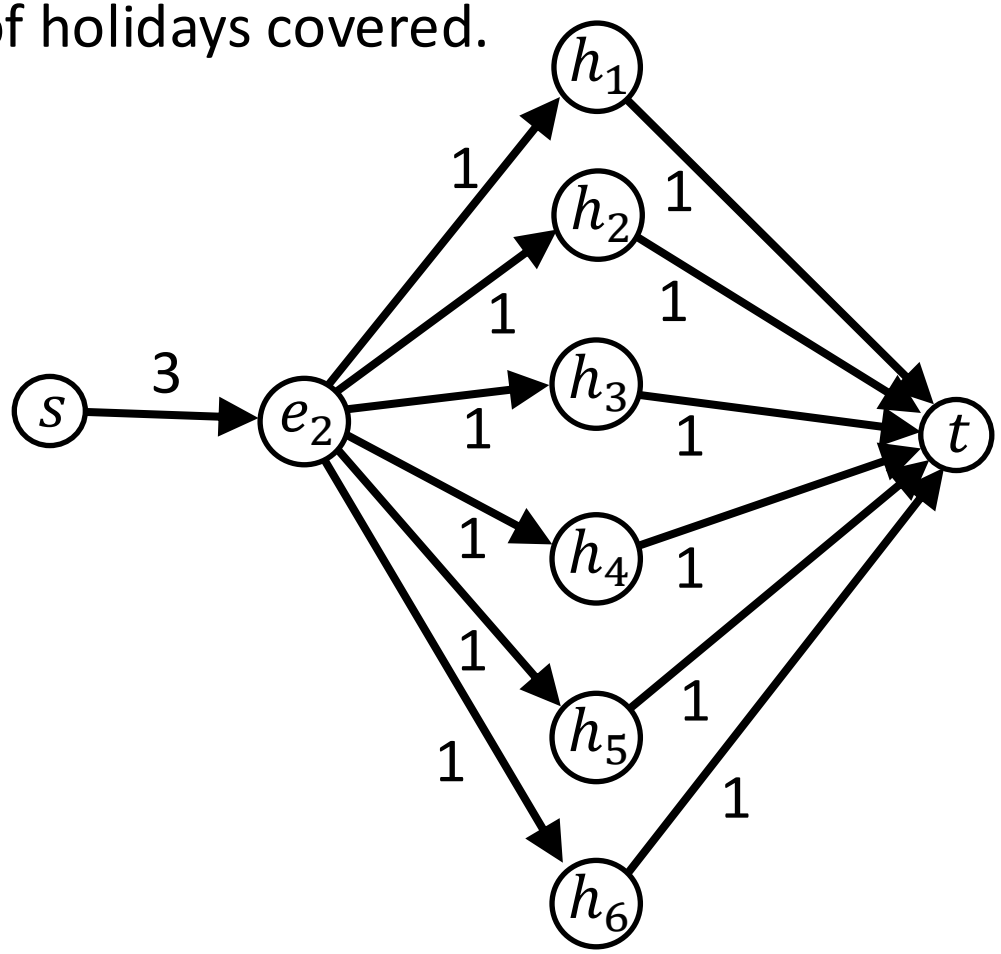
1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.

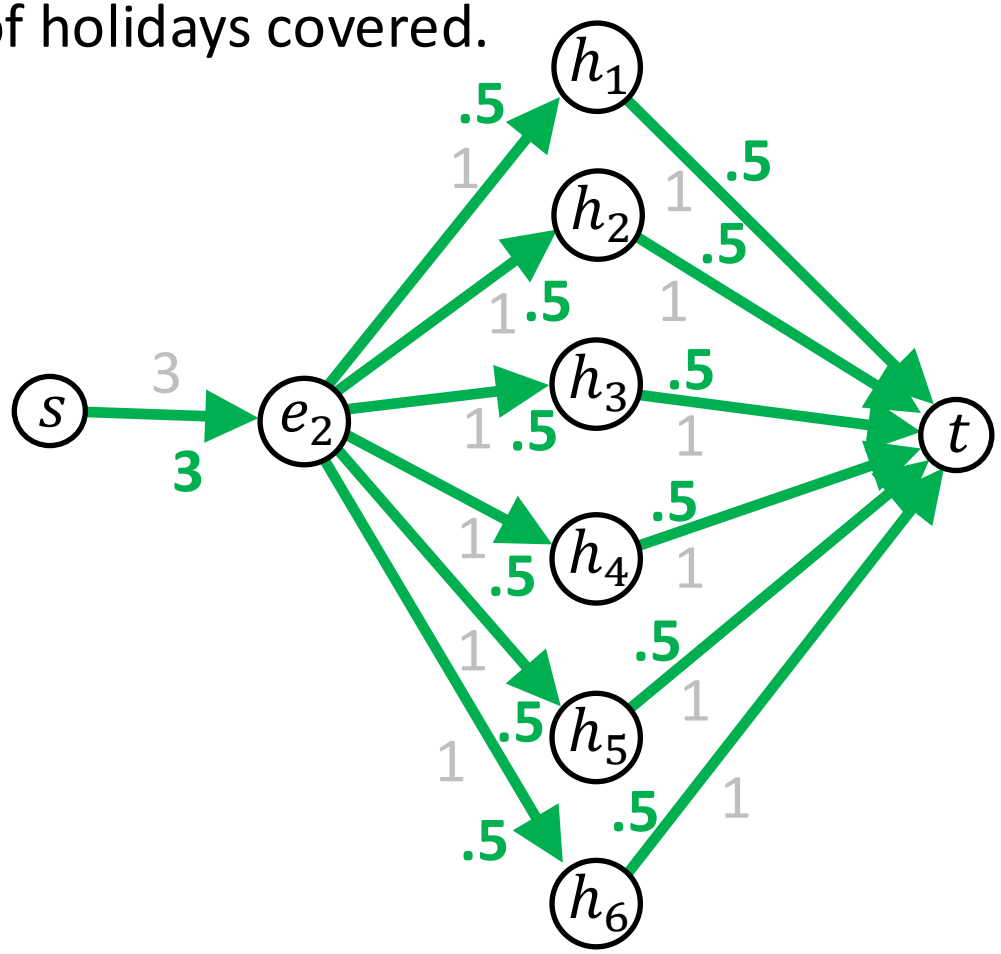


Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Work Sched

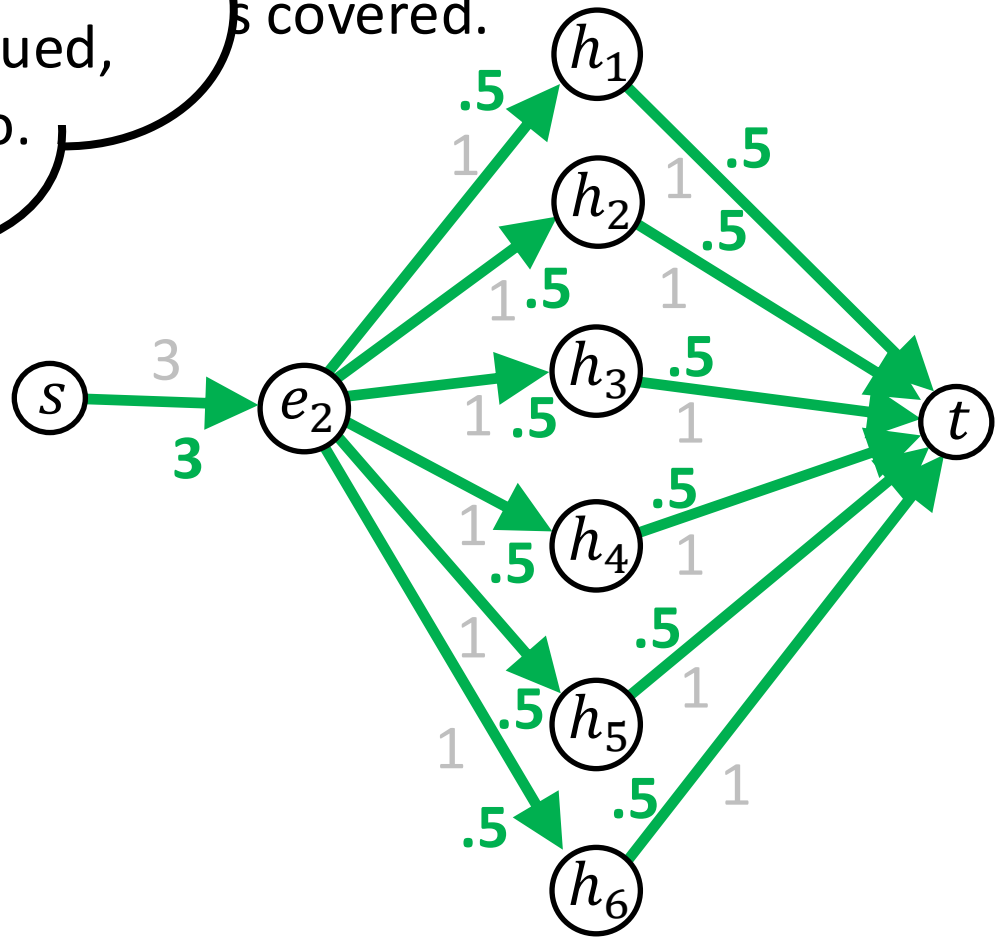
Assumptions for now:

1. We can find max flows for valid flow networks.
2. If the capacities of a flow network are integer-valued, the max flow will be too.

<u>Problem:</u> We nee ees. Each employee has a set of holida ee should work at most 3 holidays. V s covered.

<u>Algorithm:</u>

1. Build flow networ
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.

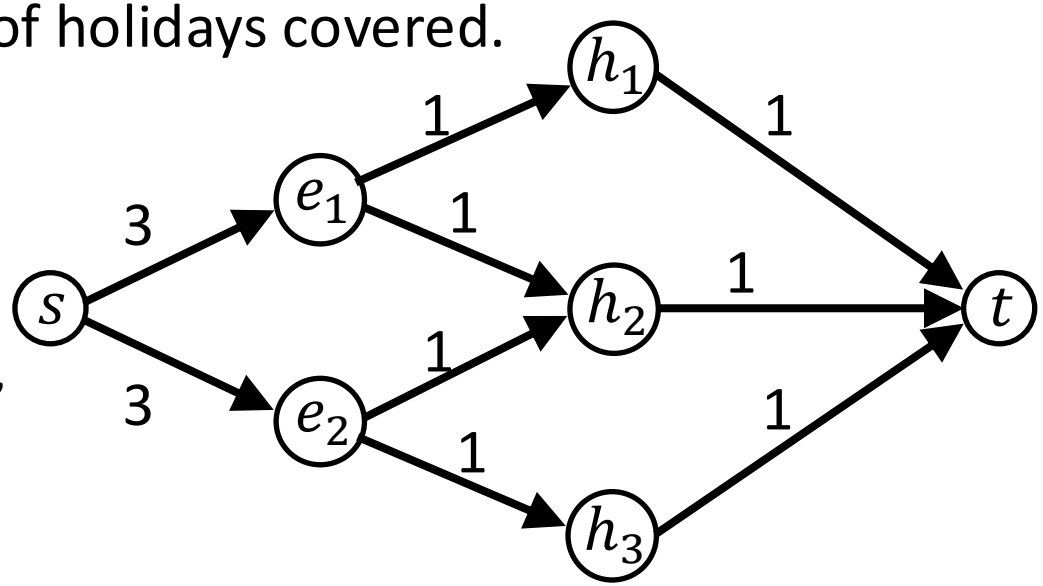


Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Validity:

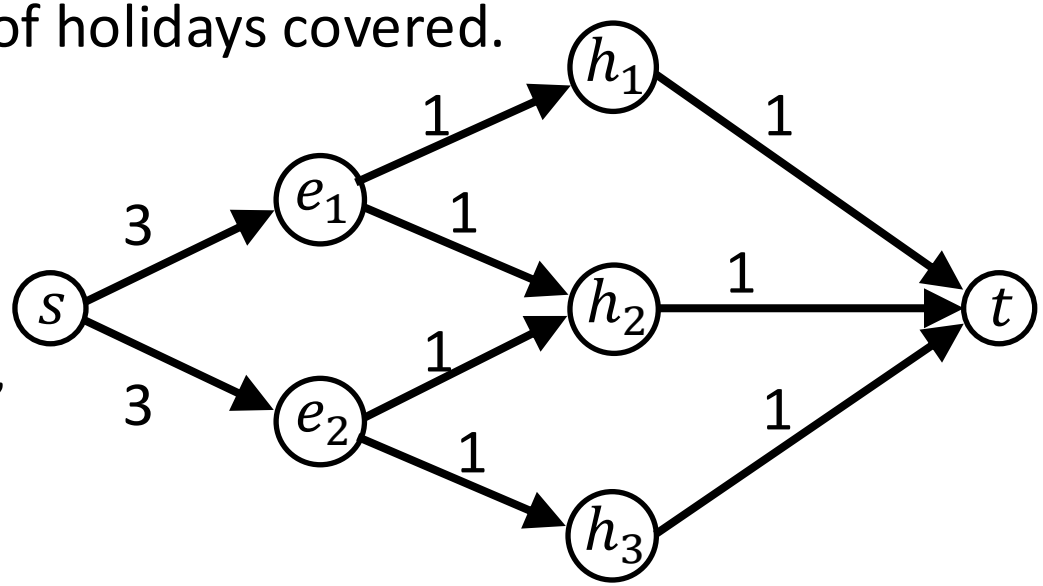
Employees cannot be assigned a holiday they are not able to work (no edges exist between incompatible pairs). Employees cannot be assigned to work more than 3 holidays (integer edge capacities yield integer flow values and total available flow is 3).

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

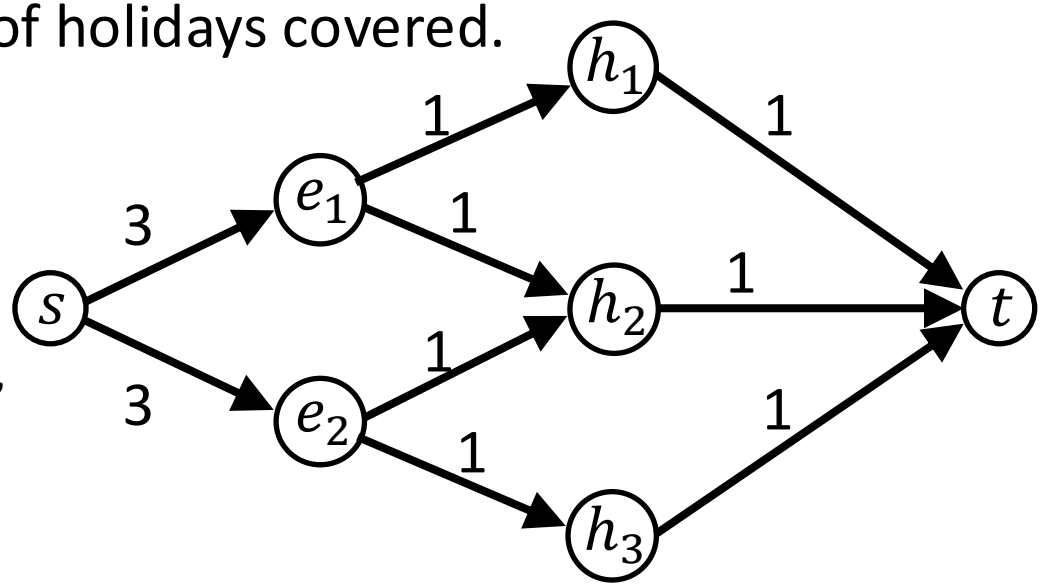
We need to show the algorithm's answer is optimal for the problem, not that the max flow found is optimal for the flow network we made.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.

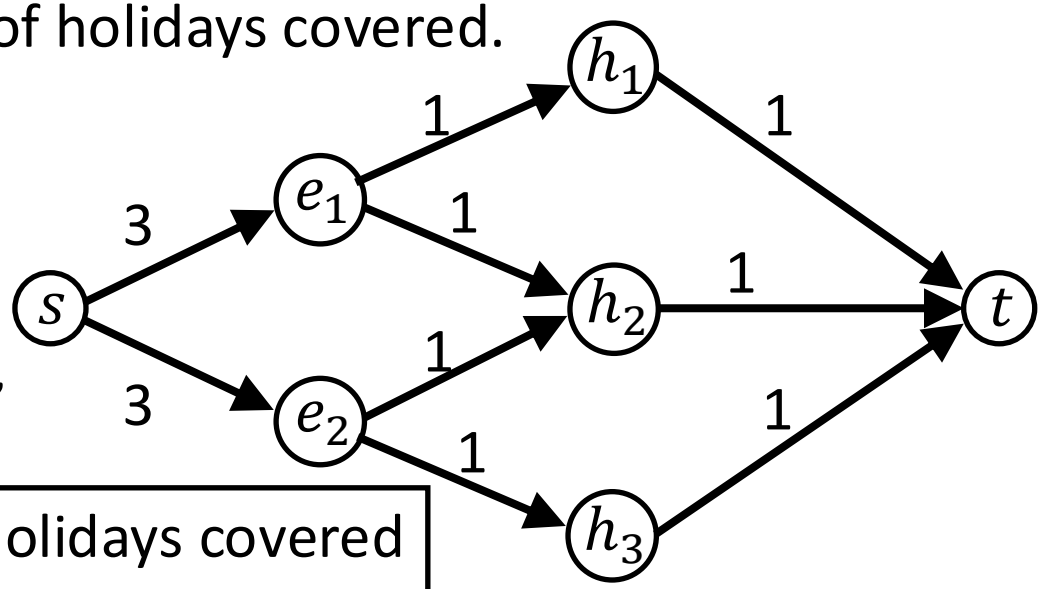


Optimality.

Road Map: Assume the schedule is not optimal. Consider a better one. Turn it into a flow that is larger than the maximum flow. Contradiction!

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.



Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.

Optimality.

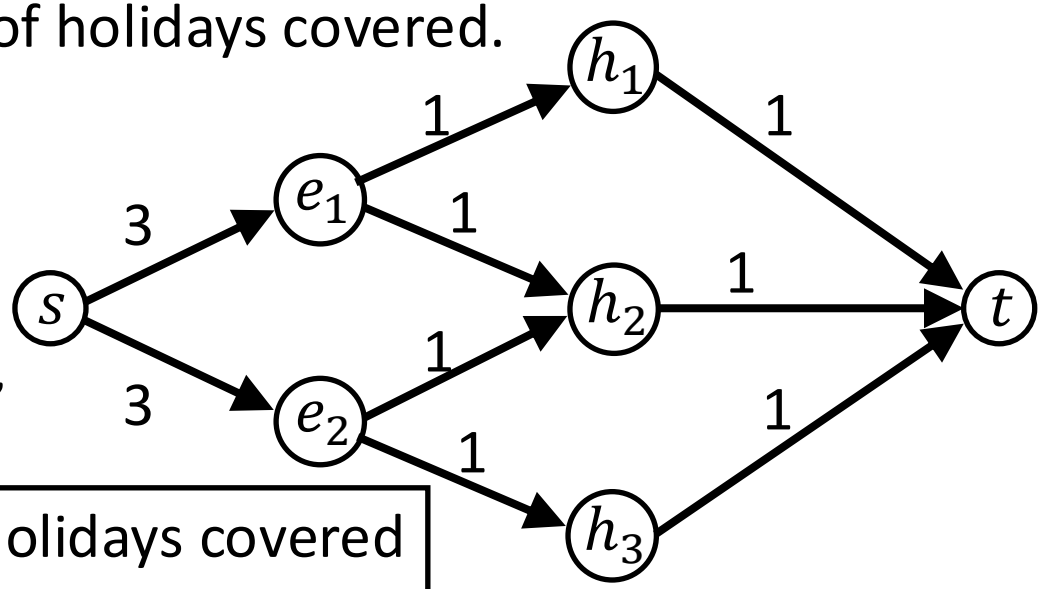
Suppose:

# holidays covered by algorithm solution	$<$	# holidays covered by optimal solution

Road Map: Assume the schedule is not optimal. Consider a better one. Turn it into a flow that is larger than the maximum flow. Contradiction!

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.



Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.

Optimality. Suppose:

$$\begin{array}{c}\text{max} \\ \text{flow } F\end{array} = \begin{array}{c}\text{\# holidays covered} \\ \text{by algorithm solution}\end{array} < \begin{array}{c}\text{\# holidays covered} \\ \text{by optimal solution}\end{array}$$

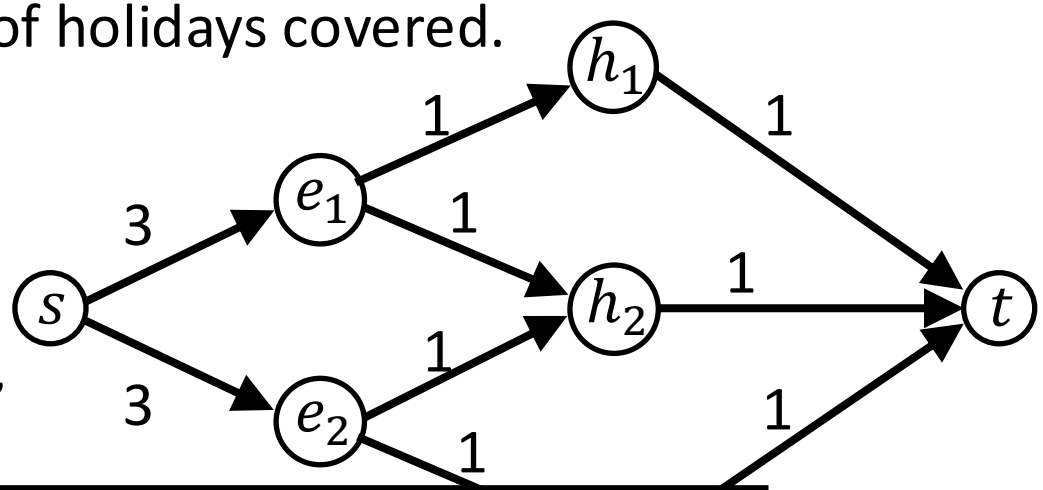
Road Map: Assume the schedule is not optimal. Consider a better one. Turn it into a flow that is larger than the maximum flow. Contradiction!

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality. Suppose:

$$\begin{array}{c|c|c|c|c} \text{max} & & \text{\# holidays covered} & & \text{valid} \\ \text{flow } F & = & \text{by algorithm solution} & < & \text{flow } F' \end{array} \quad \begin{array}{c|c} \text{\# holidays covered} & = \\ \text{by optimal solution} & \end{array}$$

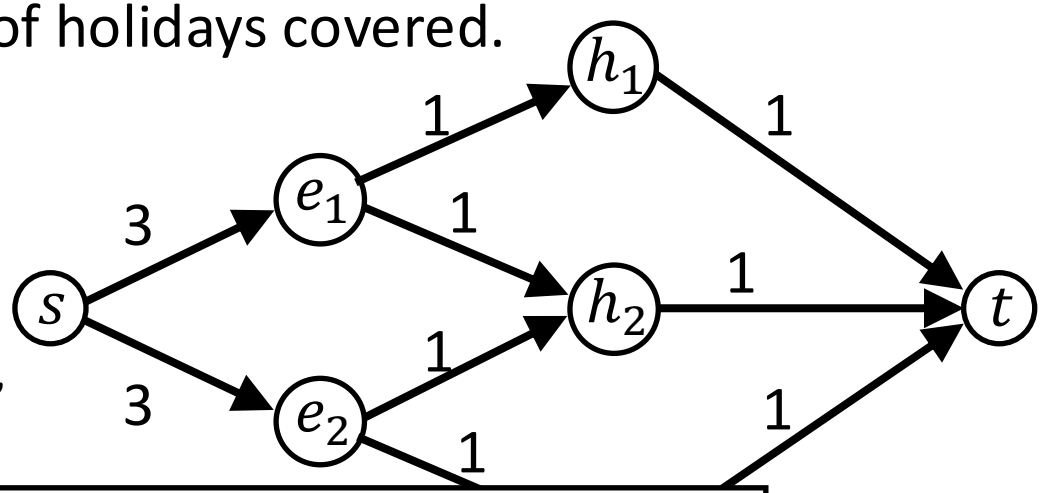
Road Map: Assume the schedule is not optimal. Consider a better one. Turn it into a flow that is larger than the maximum flow. Contradiction!

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality. Suppose:

$$\text{max flow } F = \text{\# holidays covered by algorithm solution} < \text{\# holidays covered by optimal solution} = \text{valid flow } F'$$

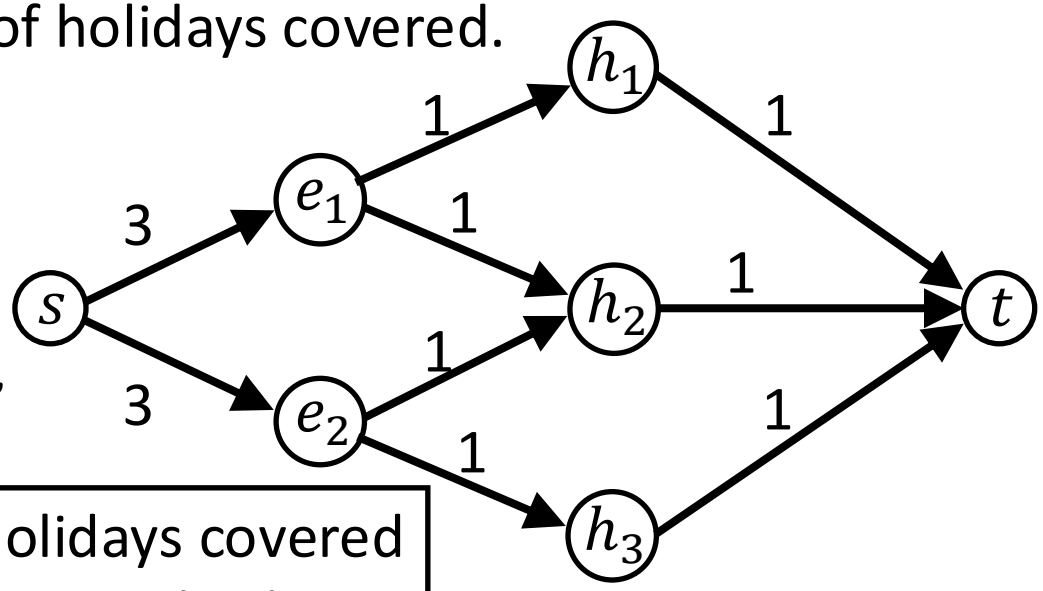
Road Map: Assume the schedule is not optimal. Consider a better one. Turn it into a flow that is larger than the maximum flow. Contradiction!

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

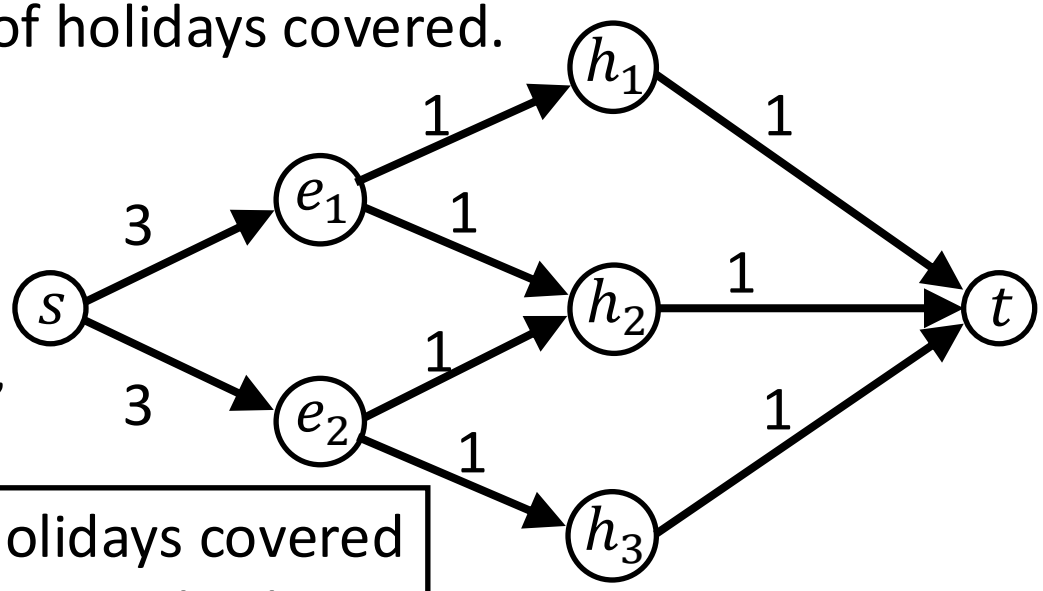
# holidays covered by algorithm solution	$<$	# holidays covered by optimal solution

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	$<$	# holidays covered by optimal solution

How should we turn a holiday schedule into a flow?

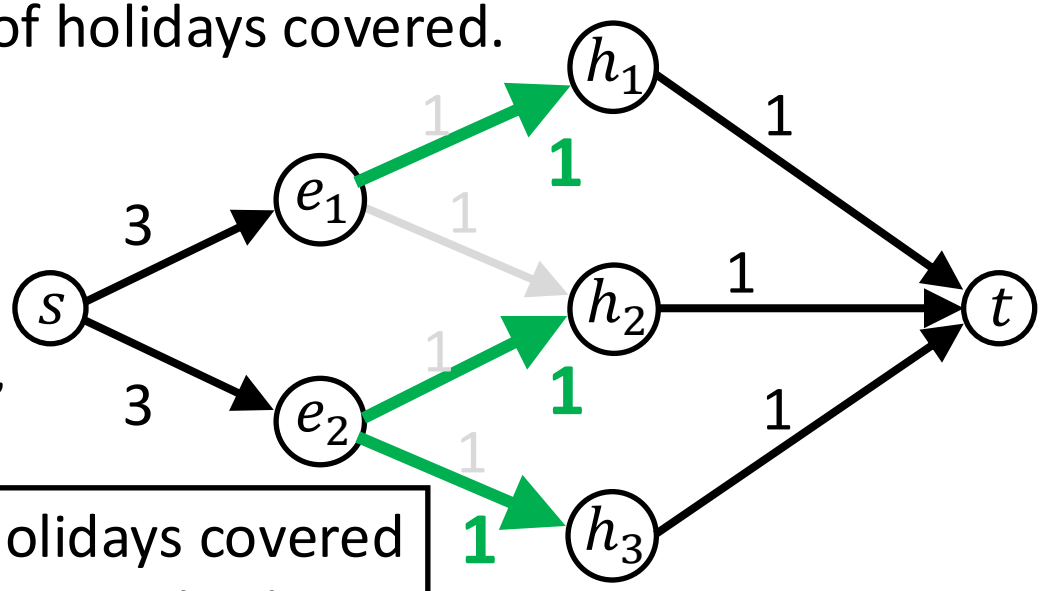
$$e_1 \to h_1, e_2 \to h_2, \ e_2 \to h_3$$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1.

How should we turn a holiday schedule into a flow?

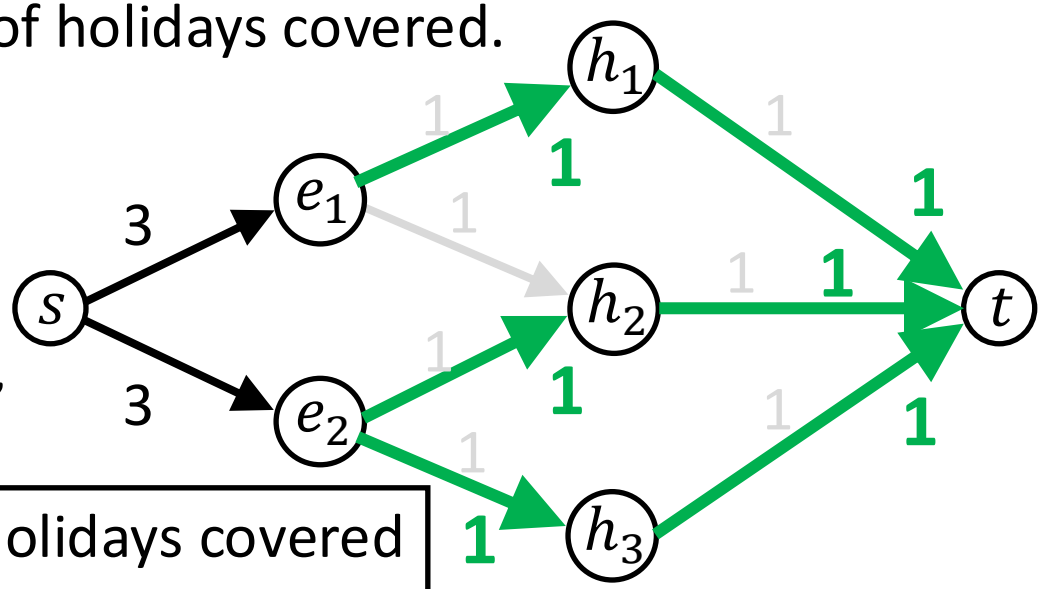
$e_1 \to h_1, e_2 \to h_2, e_2 \to h_3$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1.

How should we turn a holiday schedule into a flow?

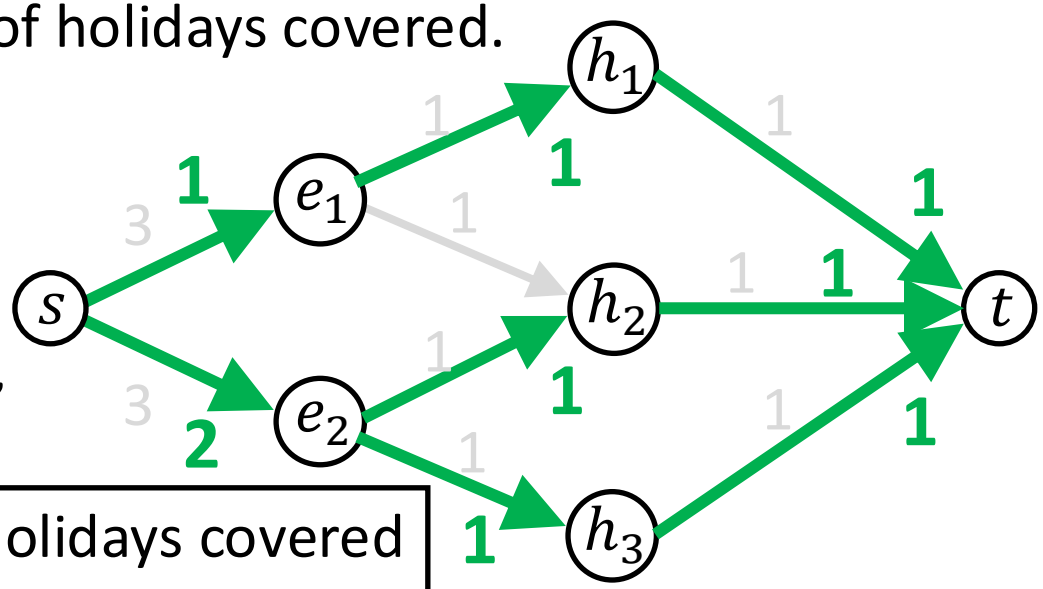
$e_1 \to h_1, e_2 \to h_2, \ e_2 \to h_3$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving.

How should we turn a holiday schedule into a flow?

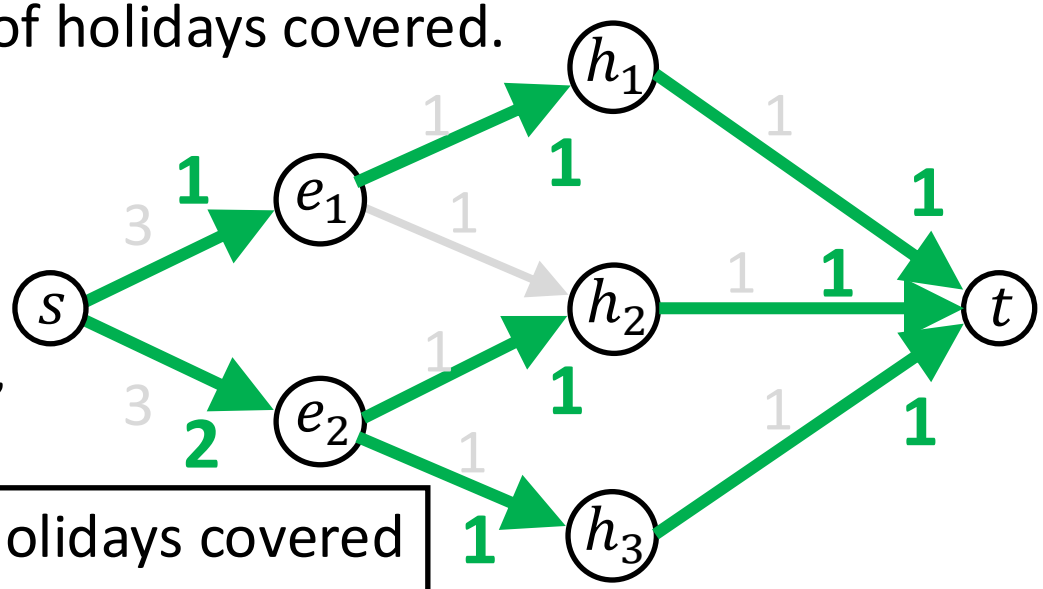
$$e_1 \to h_1, e_2 \to h_2, \ e_2 \to h_3$$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

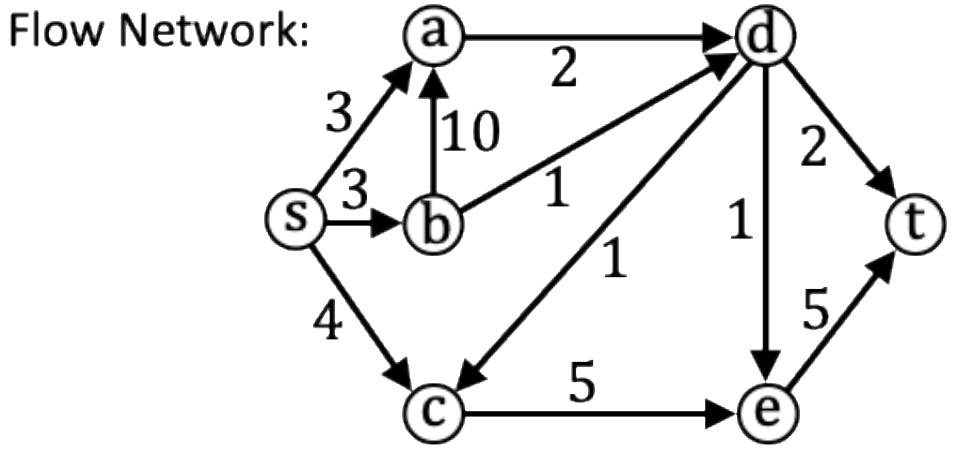
# holidays covered by algorithm solution	$<$	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving.

Now we need to argue this flow is valid. How?

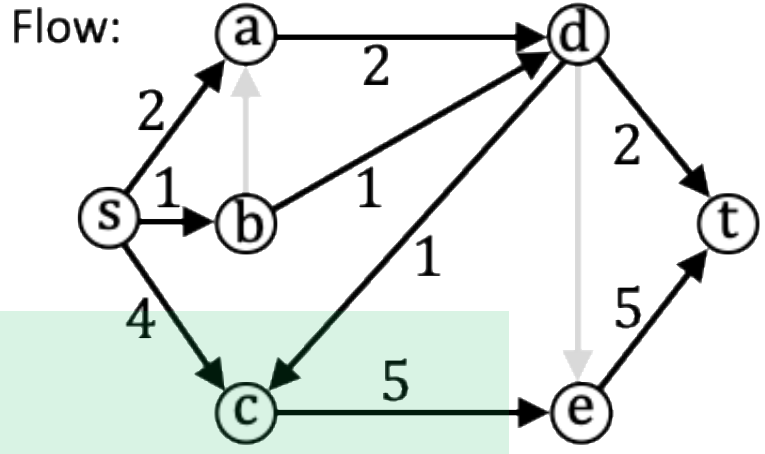
$$e_1 \to h_1, e_2 \to h_2, \ e_2 \to h_3$$

Maximum Flow Problem:
Given a flow network, find the maximum possible value of flow.



Flow Network:

- Directed-edge graph, $G = (V, E)$.
- Non-negative edge capacity, c_e.
- Single source, s, without input edges.
- Single sink, t, without output edges.



An $s - t$ flow is a function $f: E \rightarrow \mathbb{R}^+$ such that:

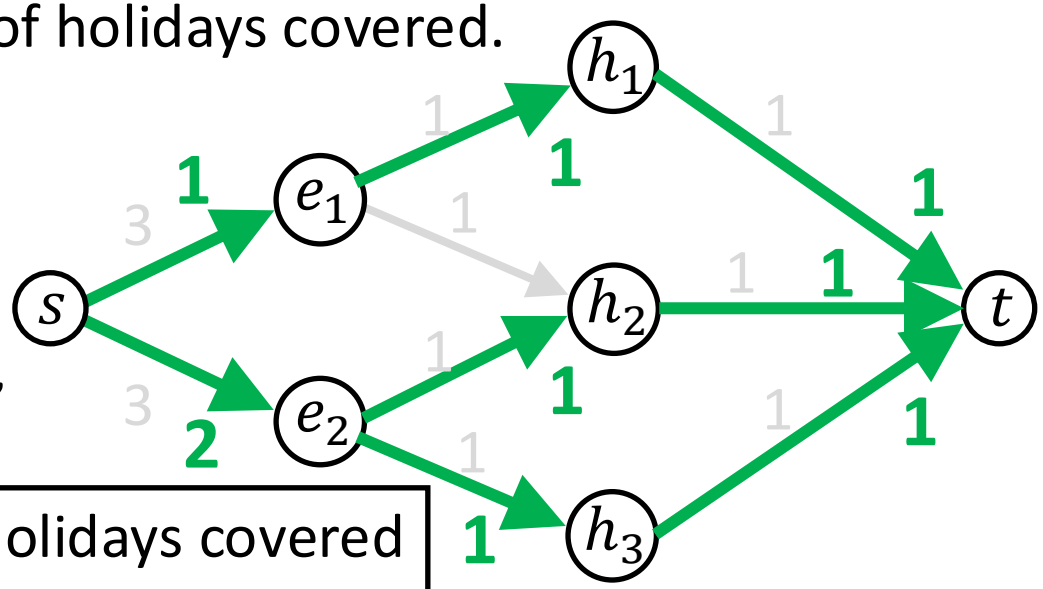
- $0 \leq f(e) \leq c_e, \forall e \in E$. (capacity constraint)
- $\sum_{e \epsilon \text{input}(v)} f(e) = \sum_{e \epsilon \text{output}(v)} f(e), \forall v \in V \setminus \{s, t\}$. (conservation of flow constraint: "Everything that goes into a node has to come out, except for s and t")
- Value of flow = $val(f)$ = $\sum_{e \epsilon \text{output}(s)} f(e)$ = $\sum_{e \epsilon \text{input}(t)} f(e)$

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

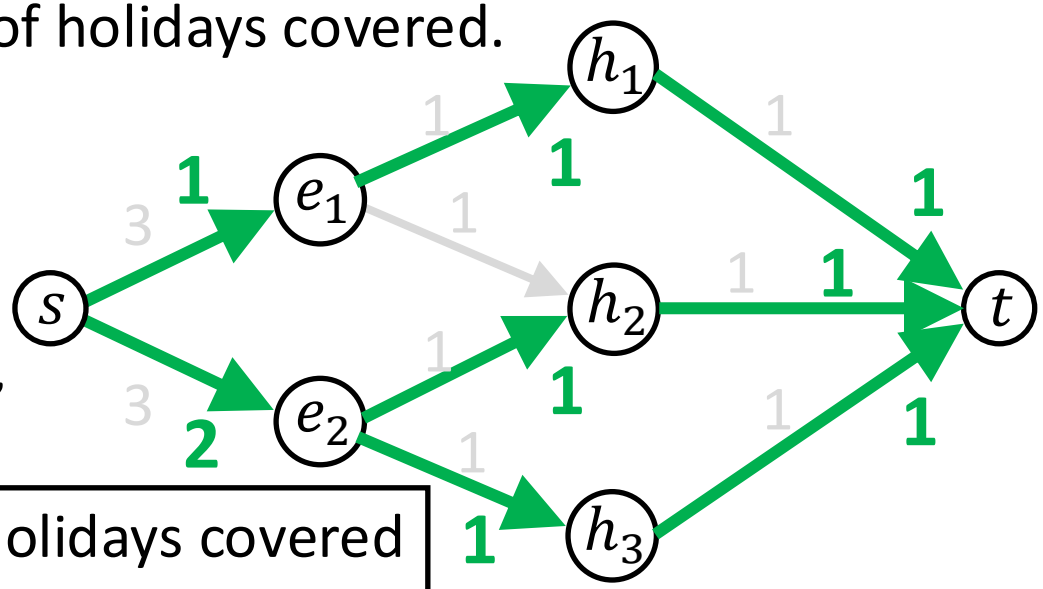
Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because…

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

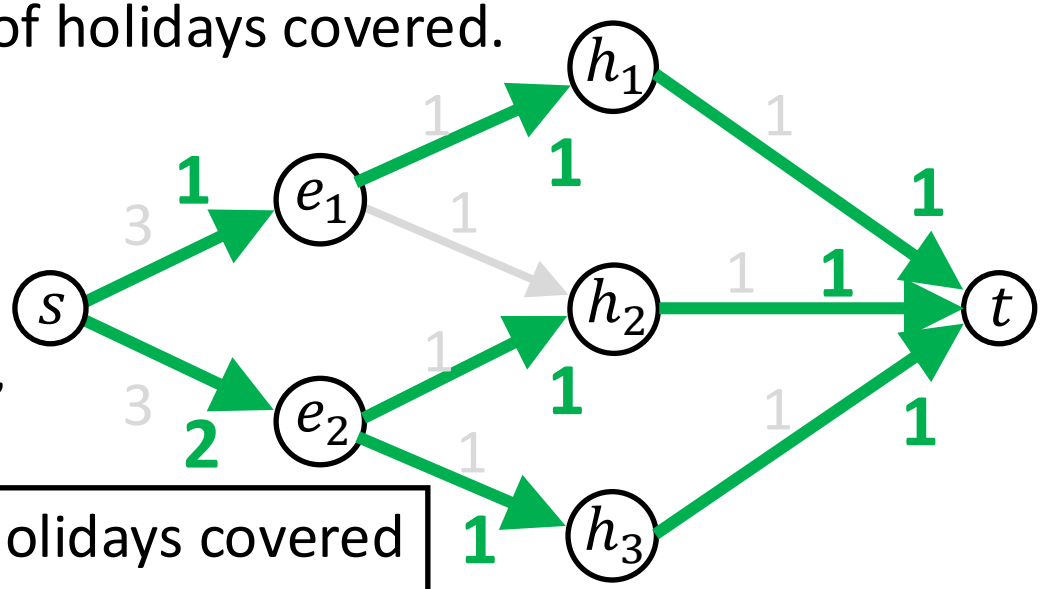
Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because…

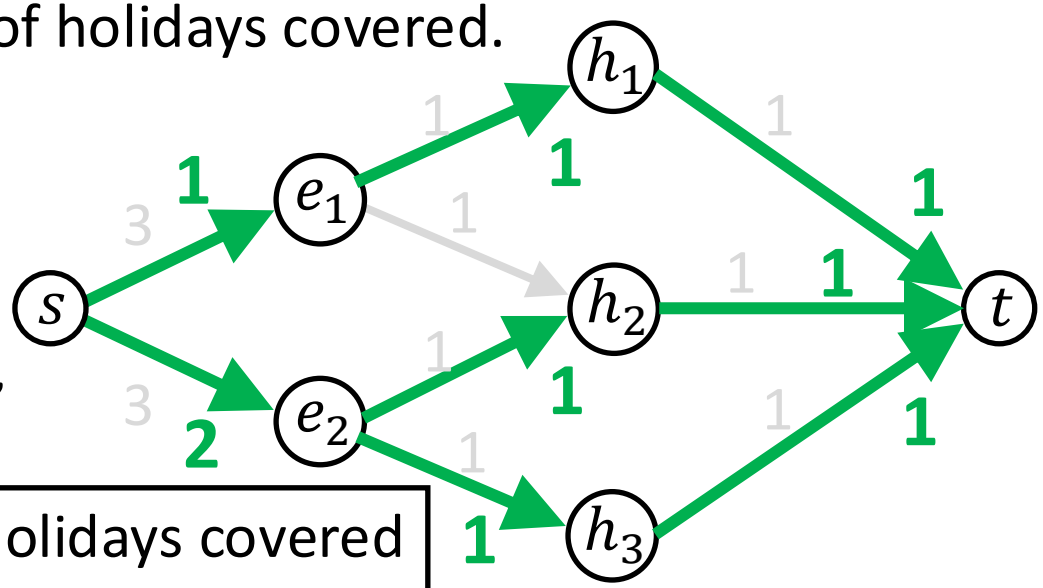
Work Scheduling

$e_1 \to h_2, e_2 \to h_2$ – Valid Schedule?

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because…

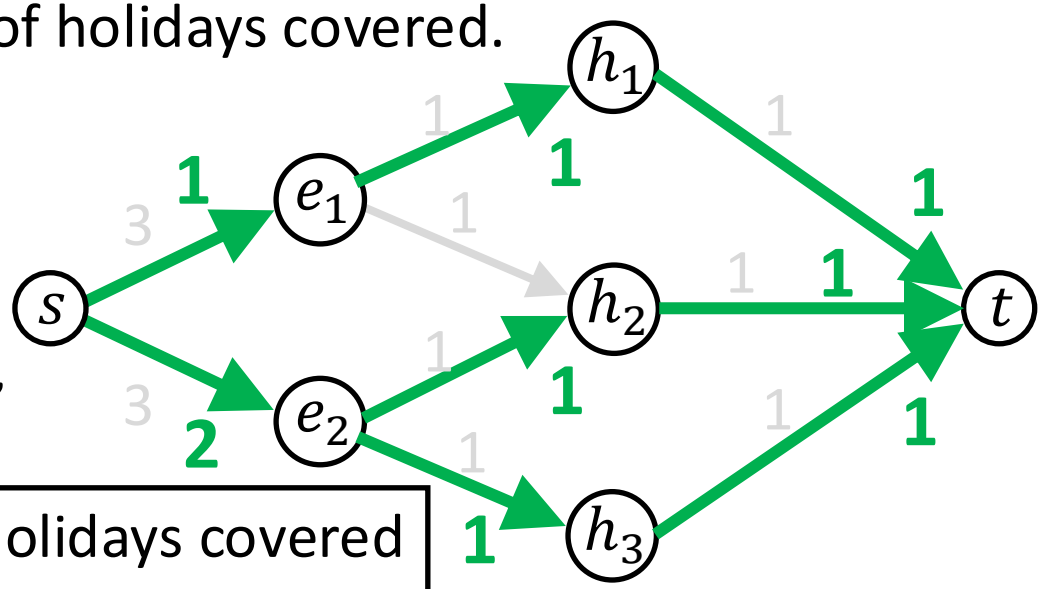
Work Scheduling

$e_1 \to h_2, e_2 \to h_2$ – Valid Schedule? ✓

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because…

Work Scheduling

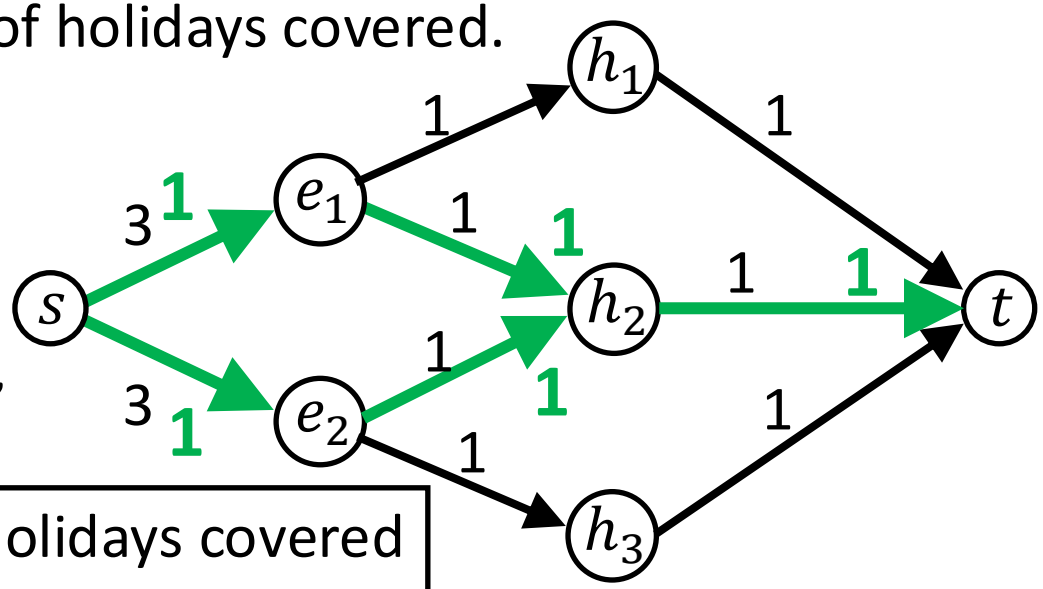
$e_1 \to h_2, e_2 \to h_2$ – Valid Schedule? ✓

Valid Flow?

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because…

Work Scheduling

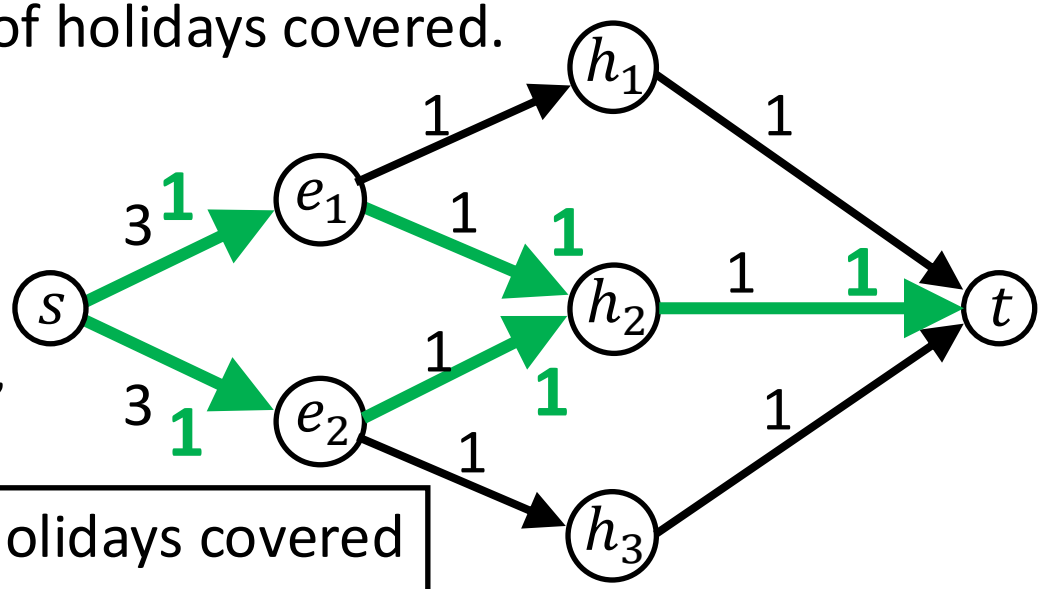
$e_1 \to h_2, e_2 \to h_2$ – Valid Schedule? ✓

Valid Flow? ✗

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because…

Work Scheduling

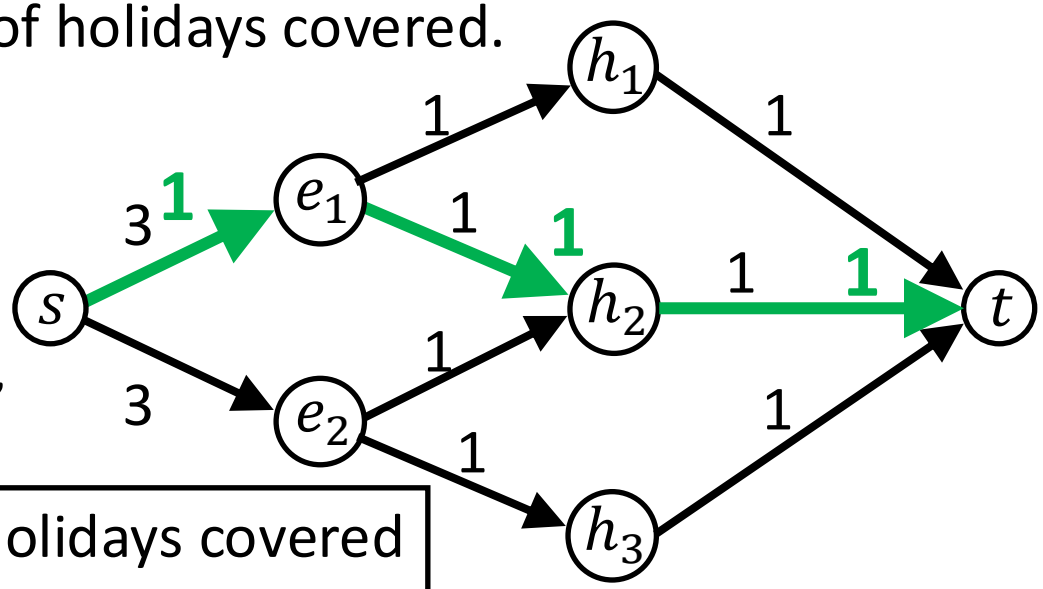
$e_1 \to h_2$, ~~$e_2 \to h_2$~~ – Valid Schedule? ✓

Valid Flow? ✗

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

First, remove redundant coverage. Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into ea

Conservation of flow holds at employees becaus

at holidays hold because…

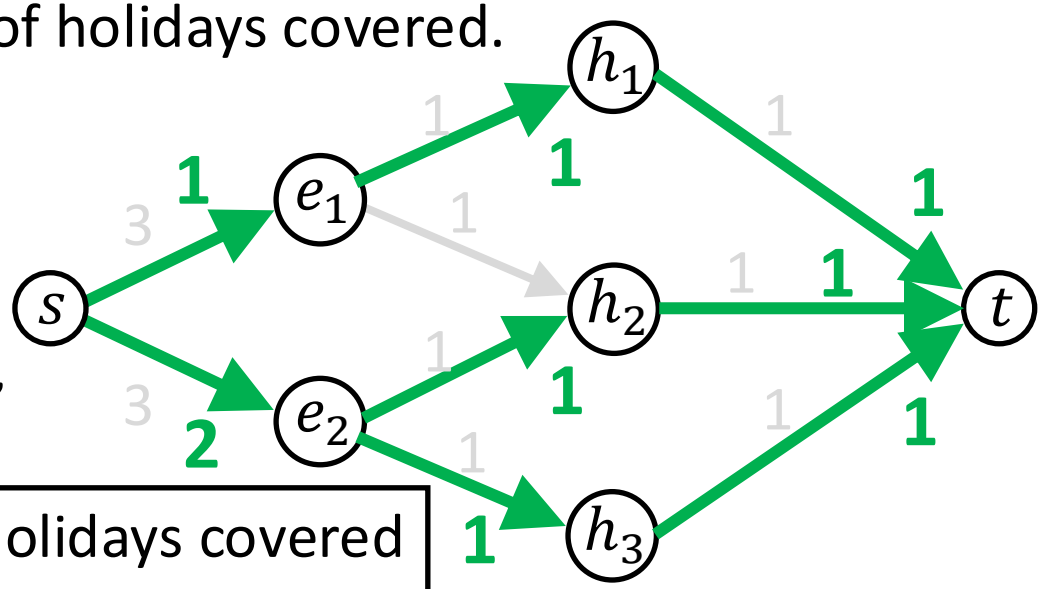
This translates a valid problem solution that is invalid on the flow network to an identical cost solution that is valid for both.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

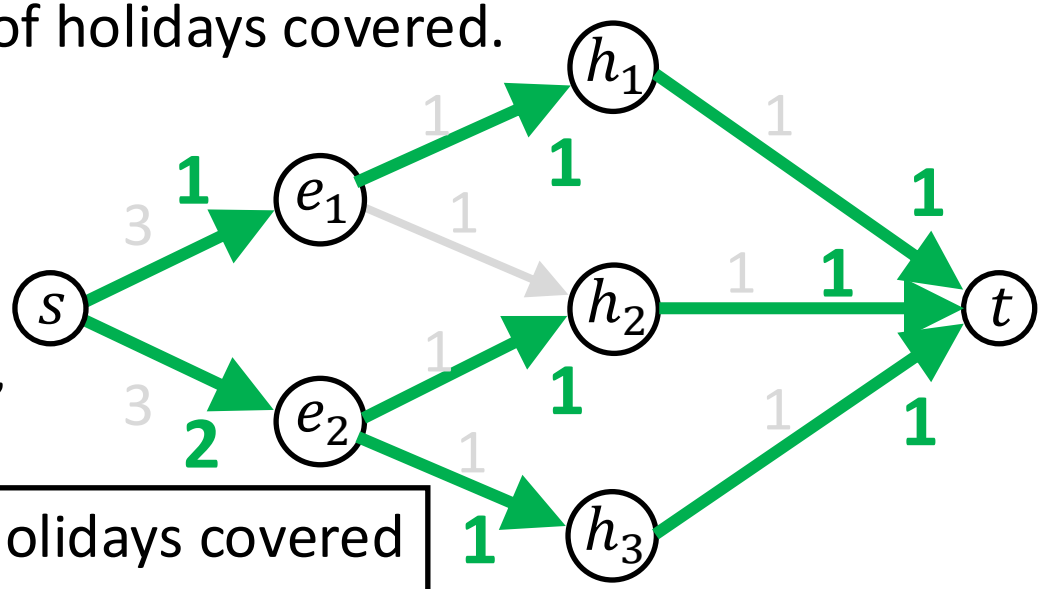
First, remove redundant coverage. Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because if a holiday is covered, it's done with a single employee.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality.

Suppose:

# holidays covered by algorithm solution	<	# holidays covered by optimal solution

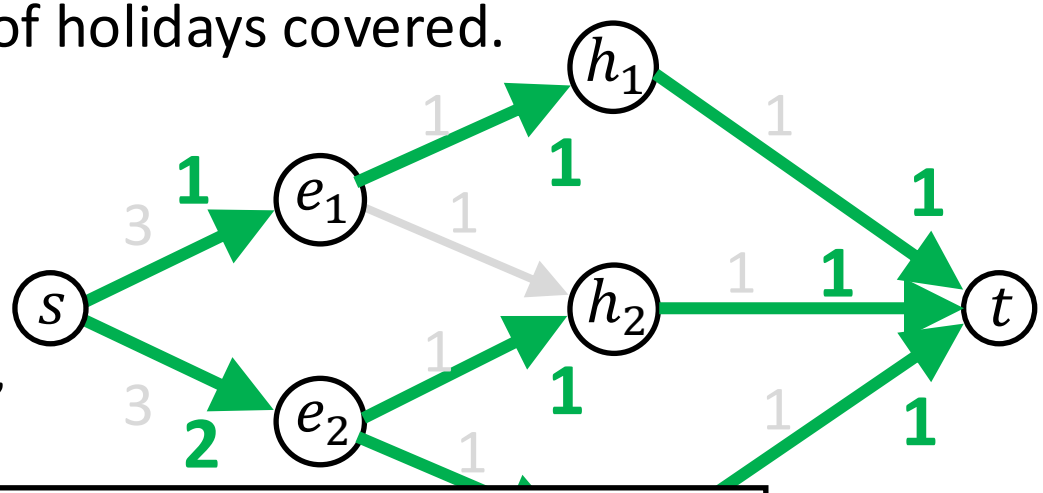
Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because if a holiday is covered, it's done with a single employee. Also, no capacities are violated.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality. Suppose:

$$\begin{array}{c}\text{max} \\ \text{flow } F\end{array} = \begin{array}{c}\text{\# holidays covered} \\ \text{by algorithm solution}\end{array} < \begin{array}{c}\text{\# holidays covered} \\ \text{by optimal solution}\end{array} = \begin{array}{c}\text{valid} \\ \text{flow } F'\end{array}$$

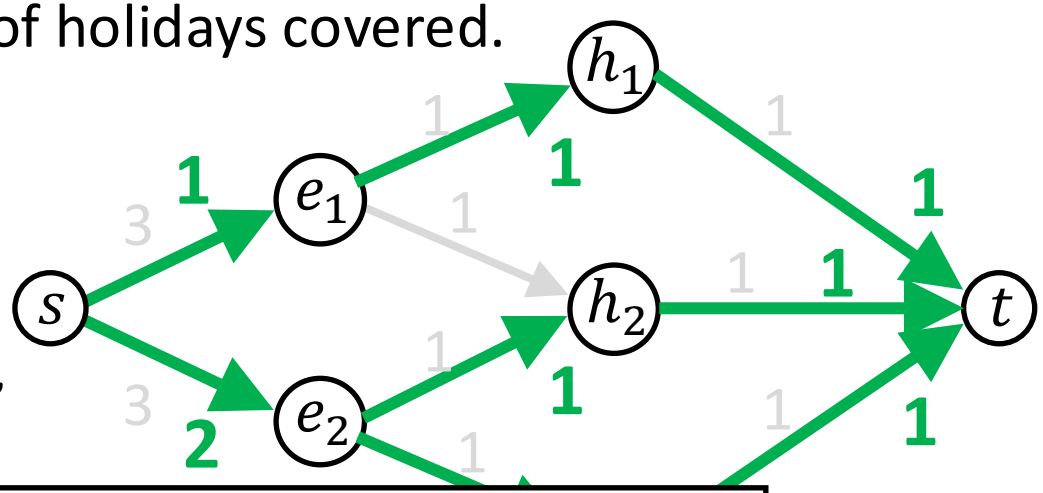
Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because if a holiday is covered, it's done with a single employee. Also, no capacities are violated. $\Rightarrow$ Valid Flow $> F$.

Work Scheduling

Problem: We need to make holiday schedules for our employees. Each employee has a set of holidays that they are able to work. Each employee should work at most 3 holidays. We want to maximize the number of holidays covered.

Algorithm:

1. Build flow network.
2. Find max flow.
3. If employee has outgoing edge carrying flow, assign them to work that holiday.



Optimality. Suppose:

$$\begin{array}{c}\text{max} \\ \text{flow } F\end{array} = \begin{array}{c}\text{\# holidays covered} \\ \text{by algorithm solution}\end{array} < \begin{array}{c}\text{\# holidays covered} \\ \text{by optimal solution}\end{array} = \begin{array}{c}\text{valid} \\ \text{flow } F'\end{array}$$

Make edge for each assignment carry 1. Make edge leaving each covered holiday carry 1. Make edge into each employee carry sum of the flow leaving. Conservation of flow holds at employees because that's how incoming flow is set. Conservation at holidays hold because if a holiday is covered, it's done with a single employee. Also, no capacities are violated. $\Rightarrow$ Valid Flow $> F$. Contradiction! Thus, algorithm is optimal.