



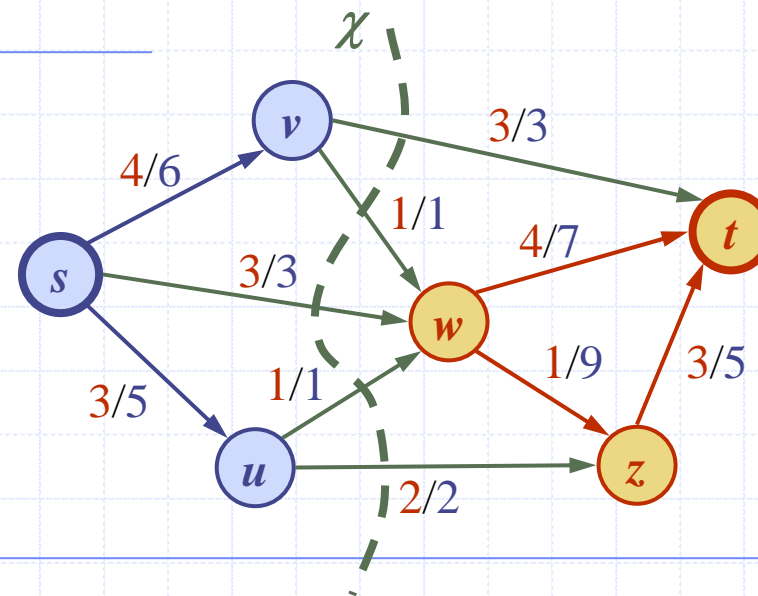
ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Αλγόριθμοι και πολυπλοκότητα

Maximum Flow

Ιωάννης Τόλλης
Τμήμα Επιστήμης Υπολογιστών

Maximum Flow



Outline and Reading

◆ Flow networks

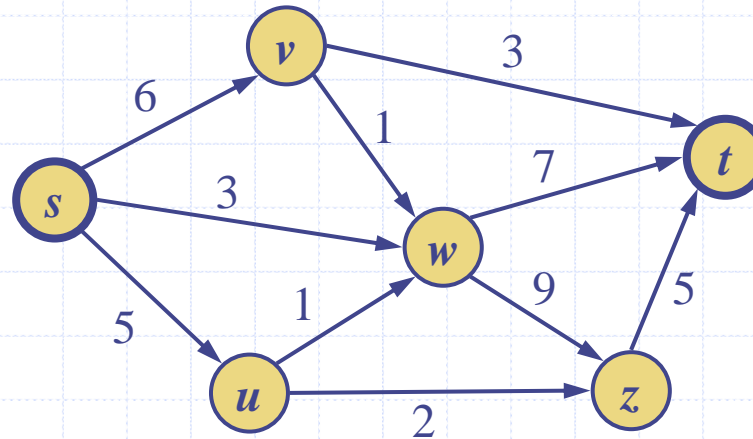
- Flow (§8.1.1)
- Cut (§8.1.2)

◆ Maximum flow

- Augmenting path (§8.2.1)
- Maximum flow and minimum cut (§8.2.1)
- Ford-Fulkerson's algorithm (§8.2.2-8.2.3)

Flow Network

- ◆ A flow network (or just network) N consists of
 - A weighted digraph G with nonnegative integer edge weights, where the weight of an edge e is called the capacity $c(e)$ of e
 - Two distinguished vertices, s and t of G , called the source and sink, respectively, such that s has no incoming edges and t has no outgoing edges.
- ◆ Example:



Flow

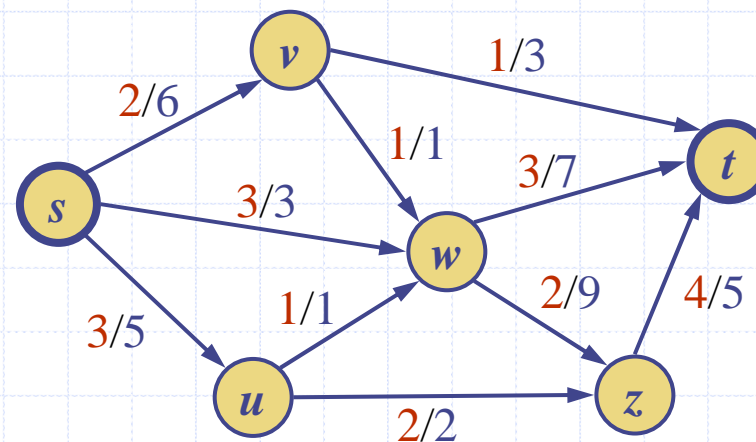
- ◆ A flow f for a network N is an assignment of an integer value $f(e)$ to each edge e that satisfies the following properties:

Capacity Rule: For each edge e , $0 \leq f(e) \leq c(e)$

Conservation Rule: For each vertex $v \neq s, t$
$$\sum_{e \in E^-(v)} f(e) = \sum_{e \in E^+(v)} f(e)$$

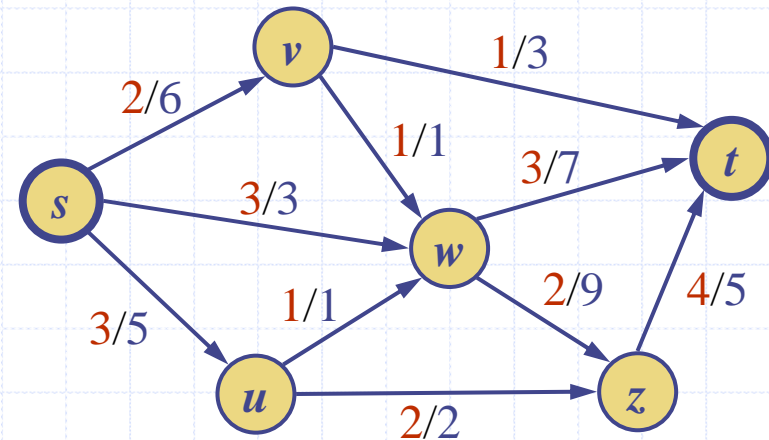
where $E^-(v)$ and $E^+(v)$ are the incoming and outgoing edges of v , resp.

- ◆ The value of a flow f , denoted $|f|$, is the total flow from the source, which is the same as the total flow into the sink
- ◆ Example:

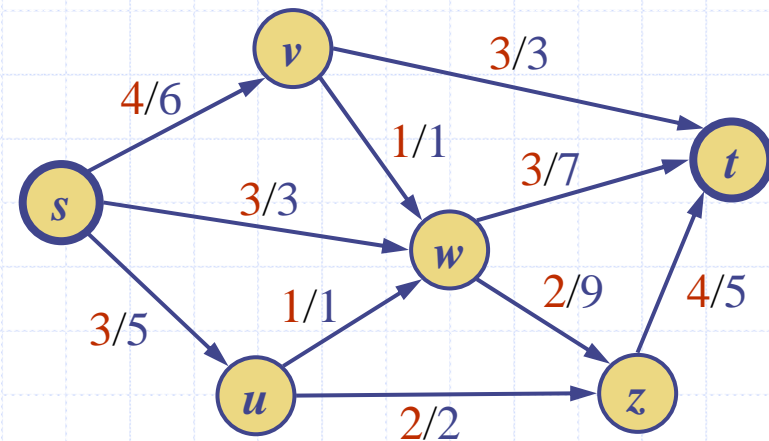


Maximum Flow

- ◆ A flow for a network N is said to be maximum if its value is the largest of all flows for N
- ◆ The maximum flow problem consists of finding a maximum flow for a given network N
- ◆ Applications
 - Hydraulic systems
 - Electrical circuits
 - Traffic movements
 - Freight transportation



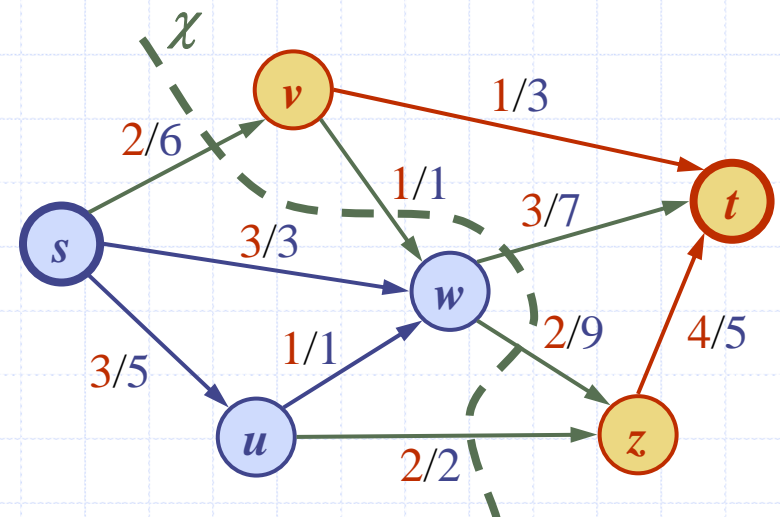
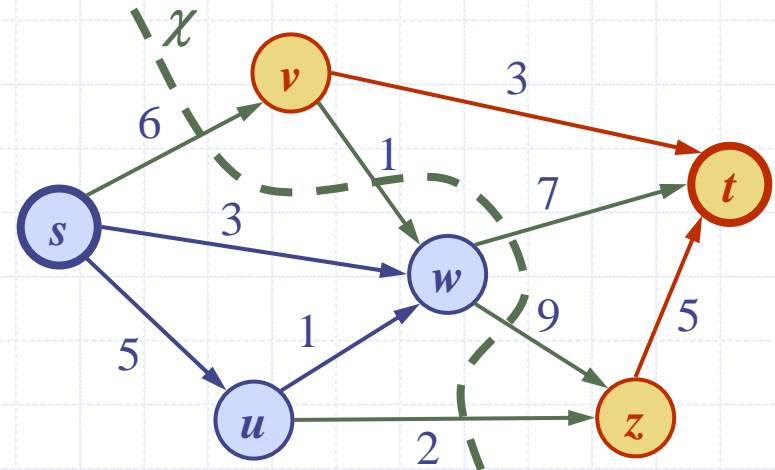
Flow of value $8 = 2 + 3 + 3 = 1 + 3 + 4$



Maximum flow of value $10 = 4 + 3 + 3 = 3 + 3 + 4$

Cut

- ◆ A cut of a network N with source s and sink t is a partition $\chi = (V_s, V_t)$ of the vertices of N such that $s \in V_s$ and $t \in V_t$
 - Forward edge of cut χ : origin in V_s and destination in V_t
 - Backward edge of cut χ : origin in V_t and destination in V_s
- ◆ Flow $f(\chi)$ across a cut χ : total flow of forward edges minus total flow of backward edges
- ◆ Capacity $c(\chi)$ of a cut χ : total capacity of forward edges
- ◆ Example:
 - $c(\chi) = 24$
 - $f(\chi) = 8$



Flow and Cut

Lemma:

The flow $f(\chi)$ across any cut χ is equal to the flow value $|f|$

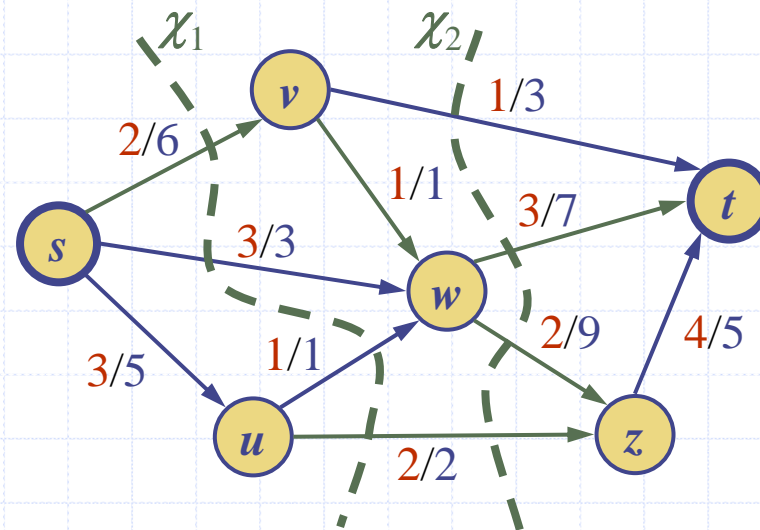
Lemma:

The flow $f(\chi)$ across a cut χ is less than or equal to the capacity $c(\chi)$ of the cut

Theorem:

The value of any flow is less than or equal to the capacity of any cut, i.e., for any flow f and any cut χ , we have

$$|f| \leq c(\chi)$$



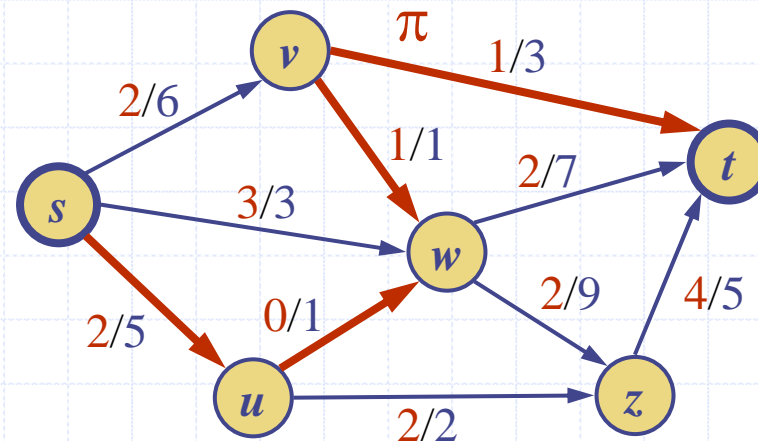
$$c(\chi_1) = 12 = 6 + 3 + 1 + 2$$

$$c(\chi_2) = 21 = 3 + 7 + 9 + 2$$

$$|f| = 8$$

Augmenting Path

- ◆ Consider a flow f for a network N
- ◆ Let e be an edge from u to v :
 - Residual capacity of e from u to v : $\Delta_f(u, v) = c(e) - f(e)$
 - Residual capacity of e from v to u : $\Delta_f(v, u) = f(e)$
- ◆ Let π be a path from s to t
 - The residual capacity $\Delta_f(\pi)$ of π is the smallest of the residual capacities of the edges of π in the direction from s to t
- ◆ A path π from s to t is an augmenting path if $\Delta_f(\pi) > 0$



$$\Delta_f(s, u) = 3$$

$$\Delta_f(u, w) = 1$$

$$\Delta_f(w, v) = 1$$

$$\Delta_f(v, t) = 2$$

$$\Delta_f(\pi) = 1$$

$$|f| = 7$$

Flow Augmentation

Lemma:

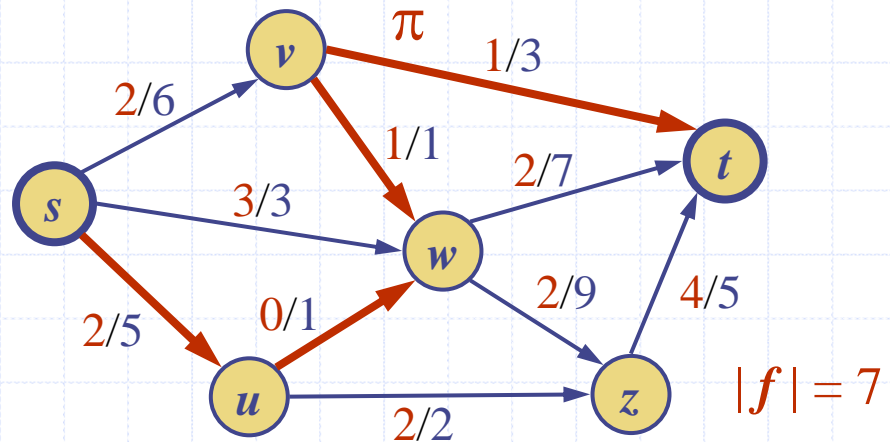
Let π be an augmenting path for flow f in network N . There exists a flow f' for N of value

$$|f'| = |f| + \Delta_f(\pi)$$

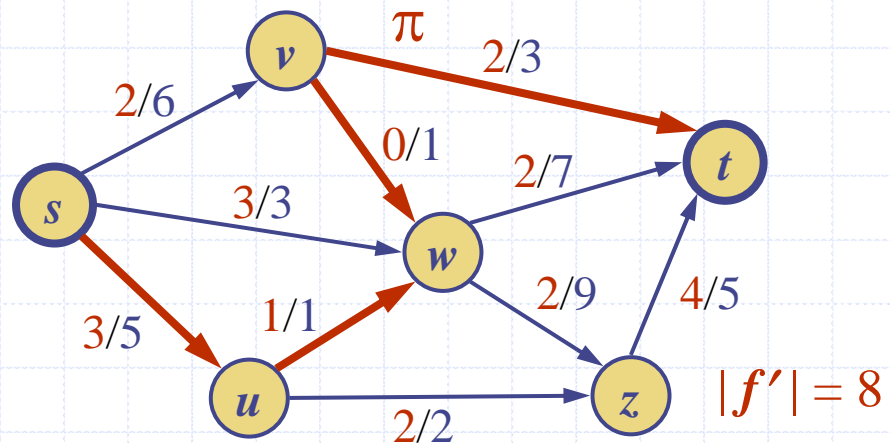
Proof:

We compute flow f' by modifying the flow on the edges of π

- Forward edge:
 $f'(e) = f(e) + \Delta_f(\pi)$
- Backward edge:
 $f'(e) = f(e) - \Delta_f(\pi)$



$$\Downarrow \Delta_f(\pi) = 1$$



Ford-Fulkerson's Algorithm

- ◆ Initially, $f(e) = 0$ for each edge e
- ◆ Repeatedly
 - Search for an augmenting path π
 - Augment by $\Delta_f(\pi)$ the flow along the edges of π
- ◆ A specialization of DFS (or BFS) searches for an augmenting path
 - An edge e is traversed from u to v provided $\Delta_f(u, v) > 0$

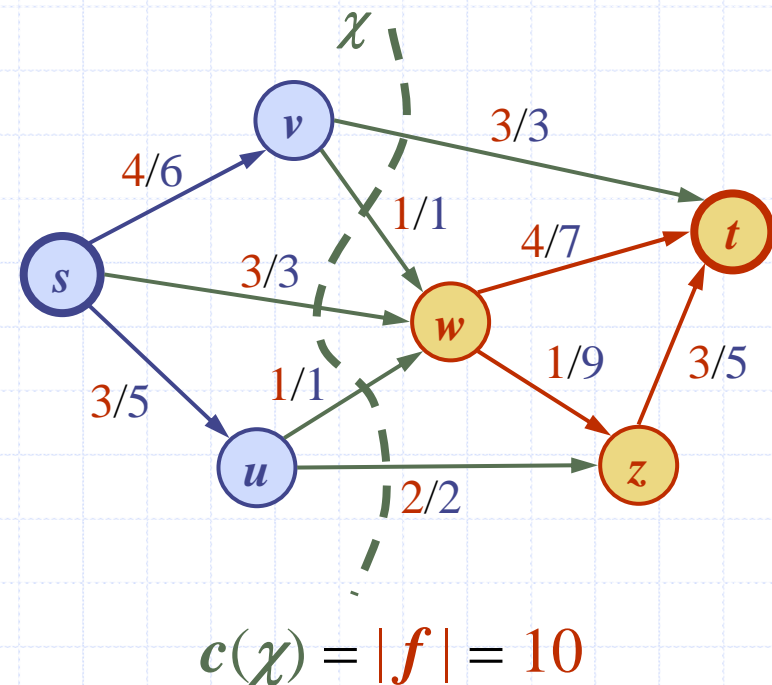
```
Algorithm FordFulkersonMaxFlow( $N$ )  
  for all  $e \in G.edges()$   
    setFlow( $e, 0$ )  
  while  $G$  has an augmenting path  $\pi$   
    { compute residual capacity  $\Delta$  of  $\pi$  }  
     $\Delta \leftarrow \infty$   
    for all edges  $e \in \pi$   
      { compute residual capacity  $\delta$  of  $e$  }  
      if  $e$  is a forward edge of  $\pi$   
         $\delta \leftarrow \text{getCapacity}(e) - \text{getFlow}(e)$   
      else {  $e$  is a backward edge }  
         $\delta \leftarrow \text{getFlow}(e)$   
      if  $\delta < \Delta$   
         $\Delta \leftarrow \delta$   
    { augment flow along  $\pi$  }  
    for all edges  $e \in \pi$   
      if  $e$  is a forward edge of  $\pi$   
        setFlow( $e, \text{getFlow}(e) + \Delta$ )  
      else {  $e$  is a backward edge }  
        setFlow( $e, \text{getFlow}(e) - \Delta$ )
```

Max-Flow and Min-Cut

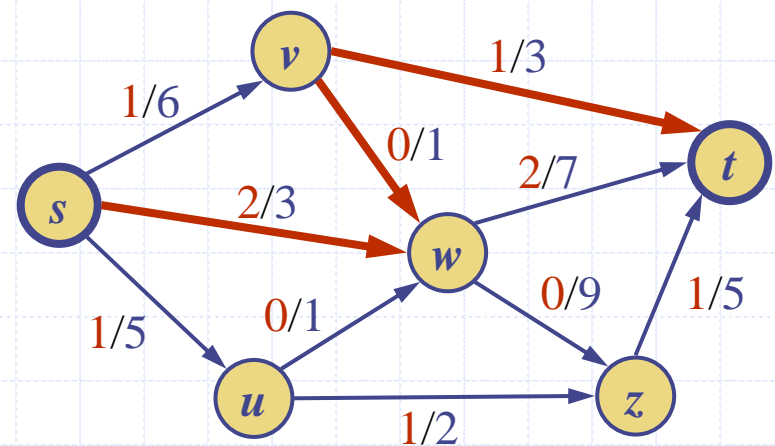
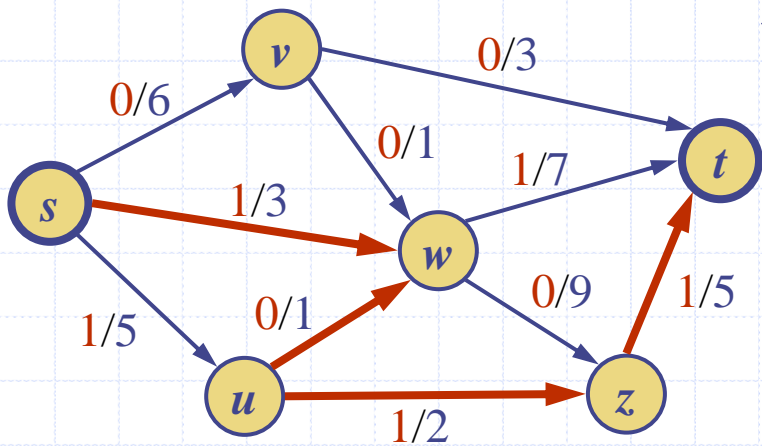
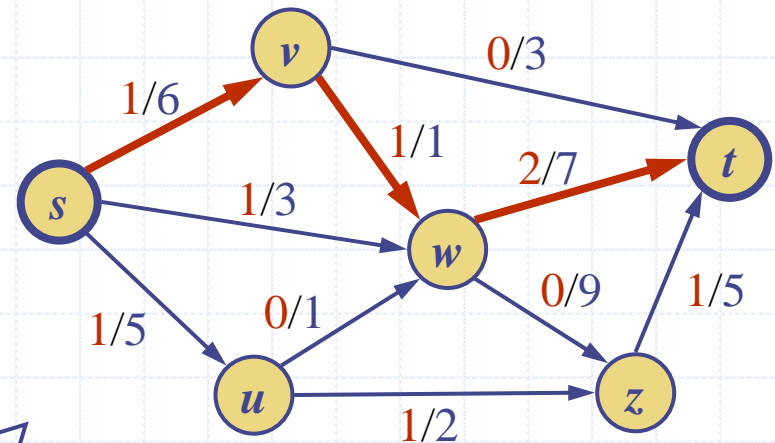
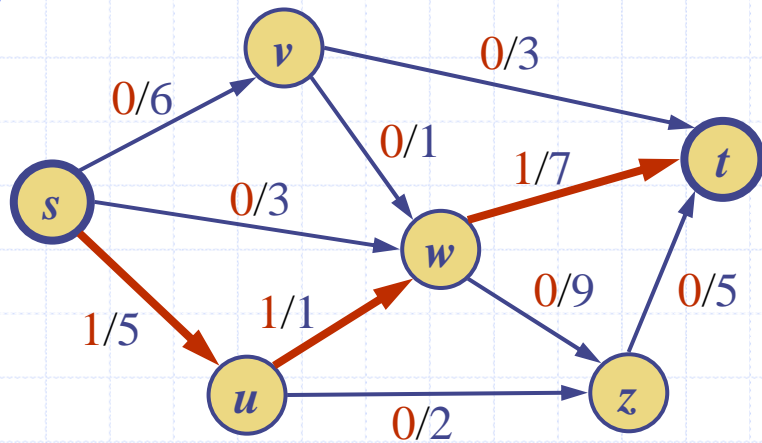
- ◆ Termination of Ford-Fulkerson's algorithm
 - There is no augmenting path from s to t with respect to the current flow f
- ◆ Define
 - V_s set of vertices reachable from s by augmenting paths
 - V_t set of remaining vertices
- ◆ Cut $\chi = (V_s, V_t)$ has capacity
 - $c(\chi) = |f|$
 - Forward edge: $f(e) = c(e)$
 - Backward edge: $f(e) = 0$
- ◆ Thus, flow f has maximum value and cut χ has minimum capacity

Theorem:

The value of a maximum flow is equal to the capacity of a minimum cut



Example (1)

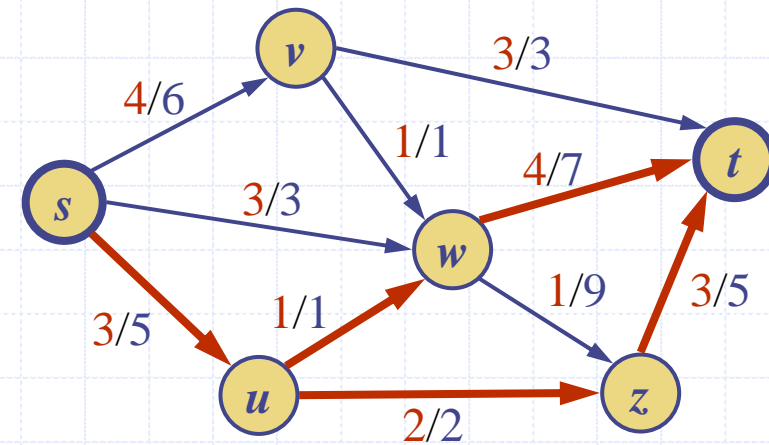
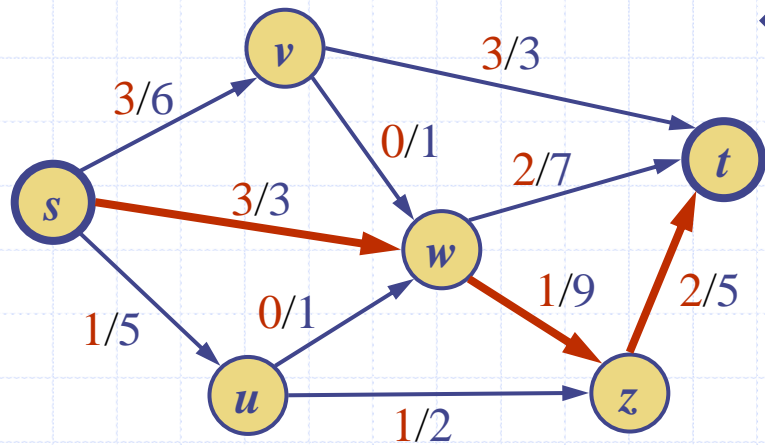
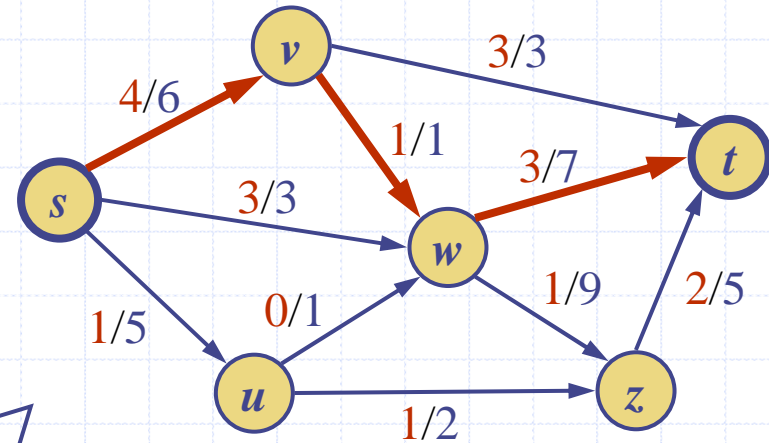
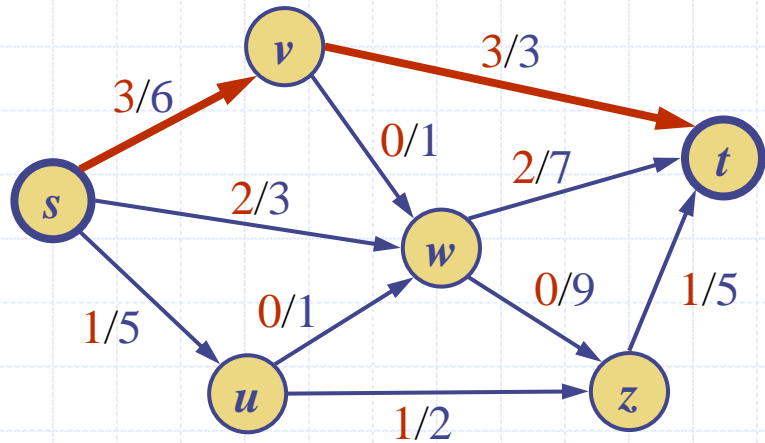


5/11/2008 11:05 PM

Maximum Flow

12

Example (2)



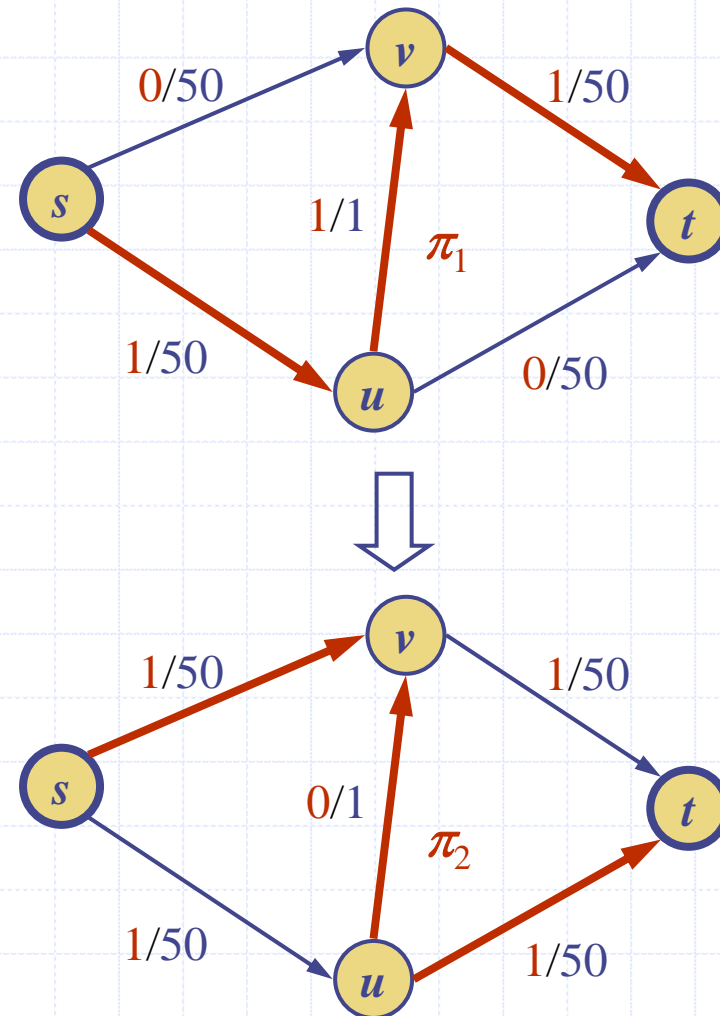
5/11/2008 11:05 PM

Maximum Flow

13

Analysis

- ◆ In the worst case, Ford-Fulkerson's algorithm performs $|f^*|$ flow augmentations, where f^* is a maximum flow
- ◆ Example
 - The augmenting paths found alternate between π_1 and π_2
 - The algorithm performs 100 augmentations
- ◆ Finding an augmenting path and augmenting the flow takes $O(n + m)$ time
- ◆ The running time of Ford-Fulkerson's algorithm is $O(|f^*|(n + m))$



Τέλος Ενότητας



Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.



Σημειώματα

Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση, Όχι Παράγωγο Έργο 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».



[1] <http://creativecommons.org/licenses/by-nc-nd/4.0/>

- Ως Μη Εμπορική ορίζεται η χρήση:
 - που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
 - που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
 - που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο
- Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

Σημείωμα Αναφοράς

Copyright Πανεπιστήμιο Κρήτης, Ιωάννης Τόλλης 2015. «Αλγόριθμοι και πολυπλοκότητα. Maximum Flow». Έκδοση: 1.0. Ηράκλειο 2015.

Διαθέσιμο από τη δικτυακή διεύθυνση:

<https://opencourses.uoc.gr/courses/course/view.php?id=368>

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.