Δίκτυα Υπολογιστών ΙΙ

(Ασκήσεις Πράξης)

Least Cost Algorithms

Τομέας Τηλεπικοινωνιών και Δικτύων

Δρ. Αναστάσιος Πολίτης
Καθηγητής Εφαρμογών
anpol@teiser.gr

Αλγόριθμοι Ελαχίστου Κόστους

• Οι αποφάσεις δρομολόγησης βασίζονται σε κριτήρια **ελαχίστου κόστους**

- αριθμός αλμάτων (hops)
- υψηλότερη χωρητικότητα
- τηλεπικοινωνιακός φόρτος
- οικονομικό κόστος
- συνδυασμός αυτών

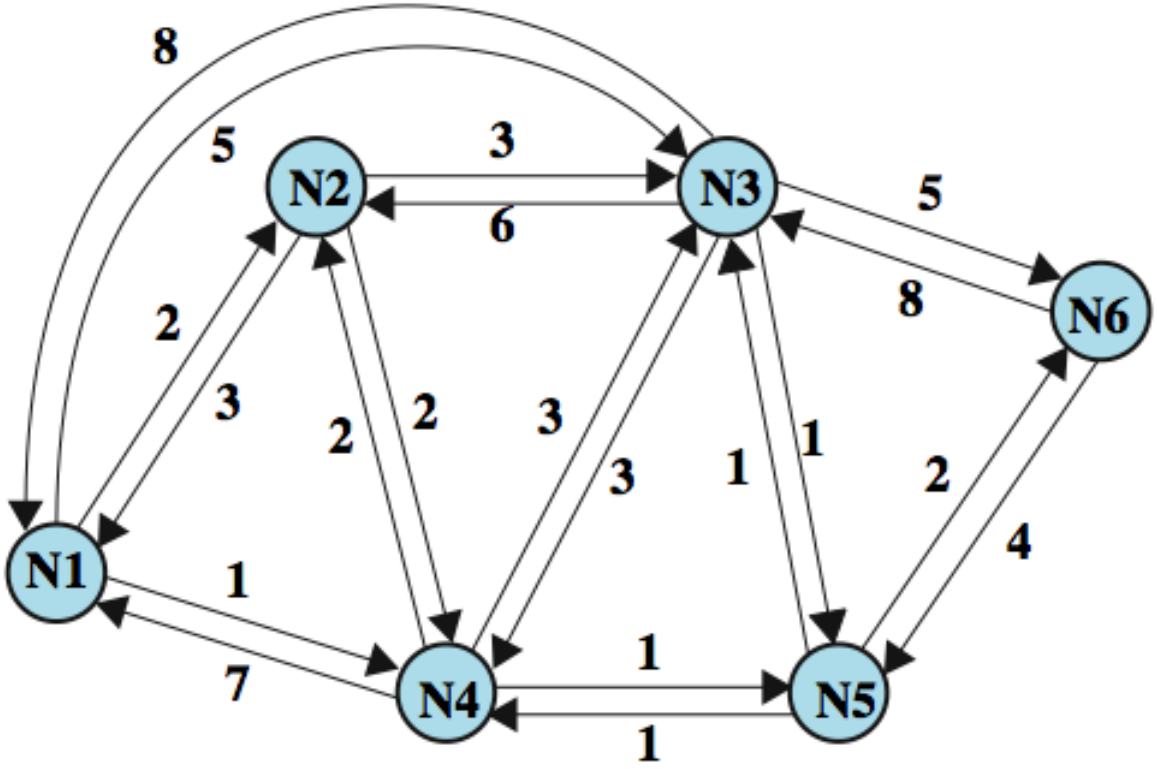
• Τα κριτήρια αυτά χρησιμοποιούνται ως είσοδος σε έναν αλγόριθμο δρομολόγησης ο οποίος έχει στόχο:

*"Σε ένα δίκτυο με κόμβους συνδεδεμένους με **αμφίδρομες** συνδέσεις, όπου κάθε σύνδεση έχει ένα **κόστος** για κάθε κατεύθυνση, προσδιόρισε το κόστος μιας **διαδρομής** ανάμεσα σε δύο κόμβους ως το **άθροισμα** των κοστών των συνδέσεων που συνθέτουν την διαδρομή. Για κάθε ζεύγος κόμβων, βρες το μονοπάτι με το **ελάχιστο** κόστος."*

• Οι περισσότεροι αλγόριθμοι δρομολόγησης που χρησιμοποιούνται στα δίκτυα είναι παραλλαγές είτε του αλγορίθμου ***Dijkstra*** είτε του αλγορίθμου ***Bellman-Ford***.

Αλγόριθμος Dijkstra

- Έστω ότι έχουμε το δίκτυο:



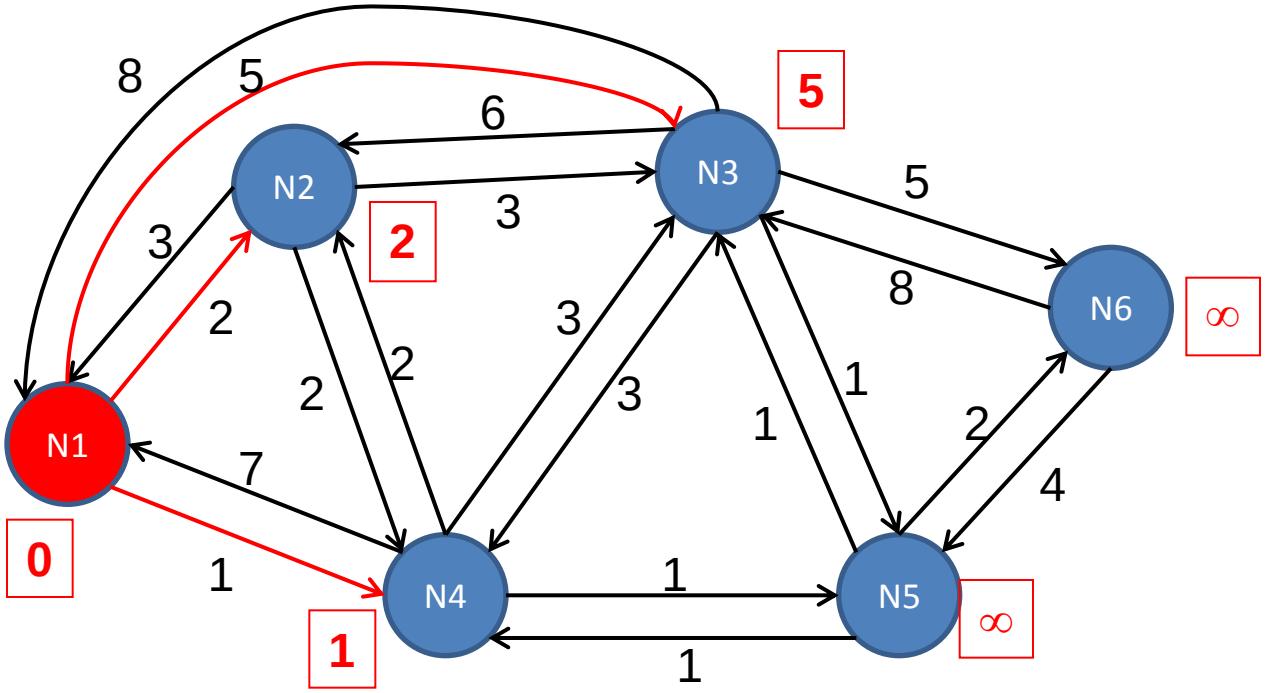
Ποιά είναι η συντομότερη διαδρομή από κάθε κόμβο προς κάθε άλλον στο δίκτυο;

Αλγόριθμος Dijkstra

- Ο αλγόριθμος Dijkstra με λίγα λόγια:
  - Επιλέγουμε έναν κόμβο εκκίνησης *s*
  - Κατόπιν επιλέγουμε τον **κοντινότερο** κόμβο (αυτόν με το μικρότερο κόστος) στον ριζικό κόμβο *s*.
    - *ο νέος κόμβος έχει ανακαλυφθεί (discovered)*
  - Κατόπιν εντοπίζουμε τον **κοντινότερο** κόμβο στο *s* τέτοιον ώστε:
    - *αυτός να μην έχει ανακαλυφθεί ακόμα*
    - *αυτός να είναι απευθείας συνδεδεμένος στον s ή να είναι συνδεδεμένος με κάποιον κόμβο που έχει ήδη ανακαλυφθεί.*
  - Η διαδικασία συνεχίζεται μέχρι να ανακαλυφθούν όλοι οι κόμβοι στο δικτύωμα.
- Κατά την αρχικοποίηση του αλγορίθμου όλοι οι κόμβοι που δεν είναι άμεσα συνδεδεμένοι στον *s* κατέχουν άπειρη απόσταση από αυτόν.

Αλγόριθμος Dijkstra

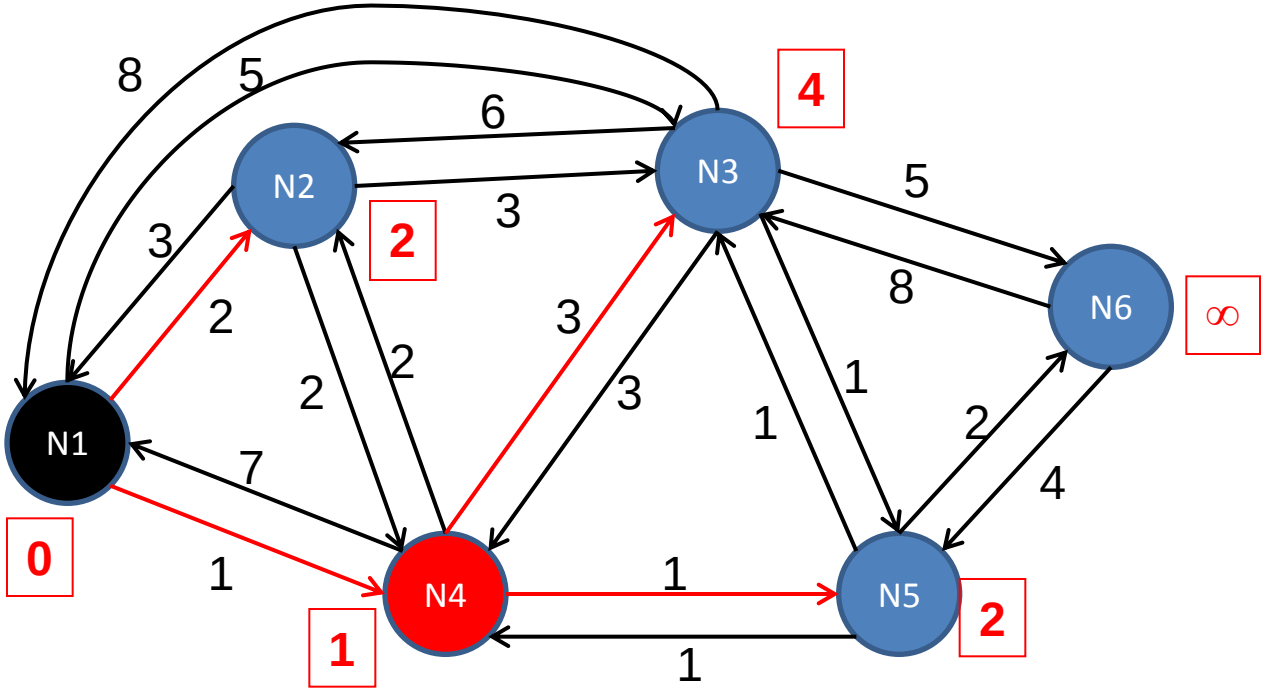
• Εκκινούμε έστω από τον κόμβο N1. Του αναθέτουμε απόσταση L(0)=0. Στους γειτονικούς του όσο είναι το βάρος της ακμής που τους συνδέει και σε όλους τους άλλους άπειρο.



Επανάληψη	Κόμβοι	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	2 (1-2)	5 (1-3)	1 (1-4)	∞ (-)	∞ (-)

Αλγόριθμος Dijkstra

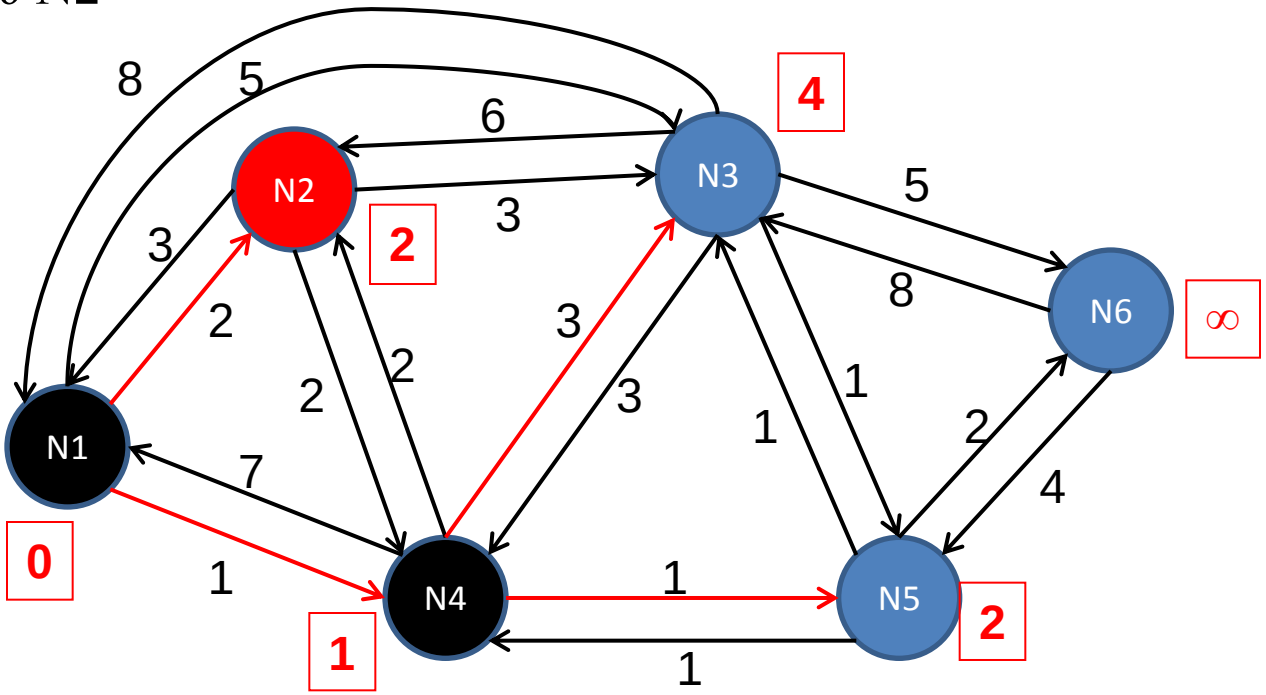
- Εξετάζουμε όλους τους γειτονικούς του N1 ξεκινώντας από αυτόν με το μικρότερο βάρος. Ανανεώνουμε τις αποστάσεις των γειτονικών του νέου κόμβου εάν η νέα απόσταση είναι μικρότερη από αυτή που ήδη έχει.



Επανάληψη	T	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	2 (1-2)	5 (1-3)	1 (1-4)	∞ (-)	∞ (-)
2	{1,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)

Αλγόριθμος Dijkstra

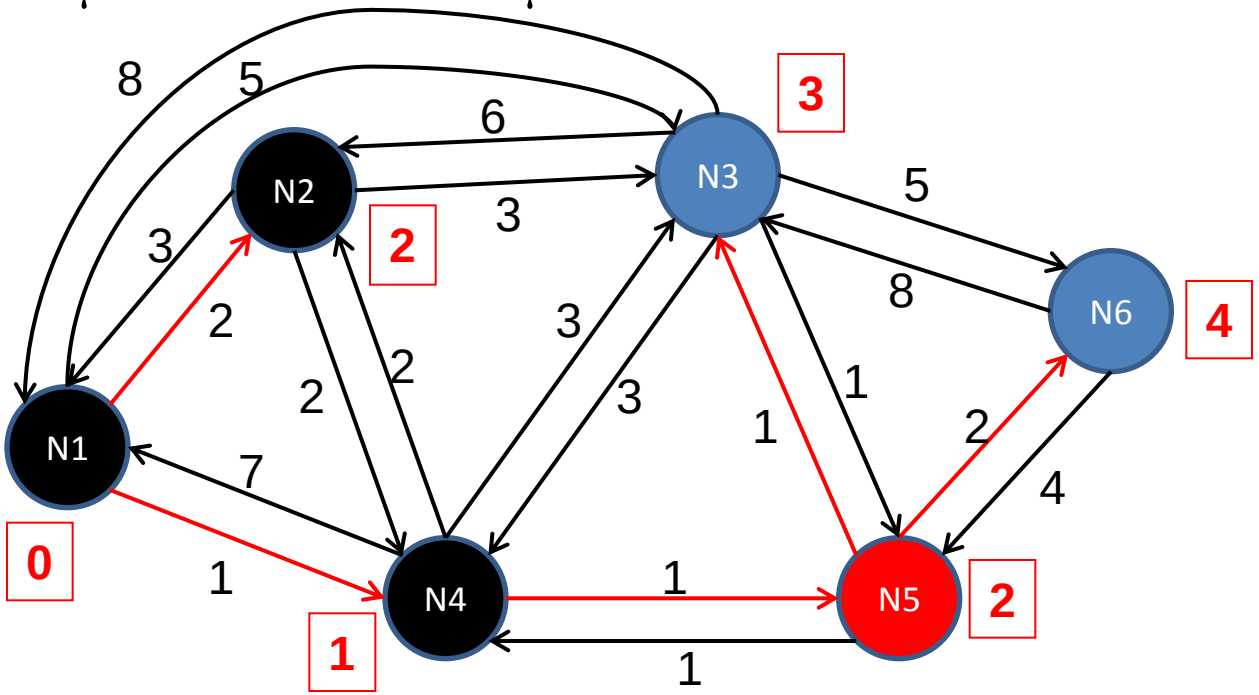
- Η σειρά του N2



Επανάληψη	T	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	2 (1-2)	5 (1-3)	1 (1-4)	∞ (-)	∞ (-)
2	{1,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
3	{1,2,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)

Αλγόριθμος Dijkstra

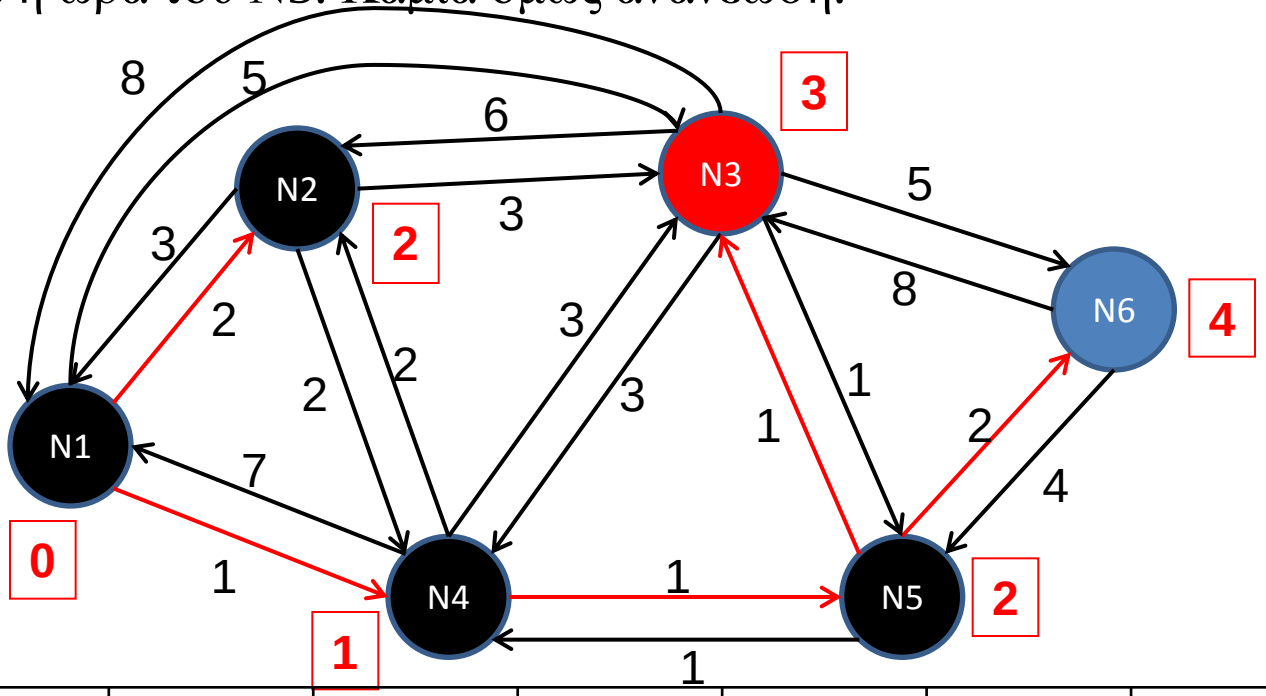
• Ο N3 δεν εξετάζεται τώρα γιατί η νέα απόσταση του είναι μικρότερη της ακμής που τον συνδέει με τον N1. Οπότε πάμε στον N5.



Επανάληψη	T	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	2 (1-2)	5 (1-3)	1 (1-4)	∞ (-)	∞ (-)
2	{1,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
3	{1,2,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
4	{1,2,4,5}	2 (1-2)	3 (1-4-5-3)	1 (1-4)	2 (1-4-5)	4 (1-4-5-6)

Αλγόριθμος Dijkstra

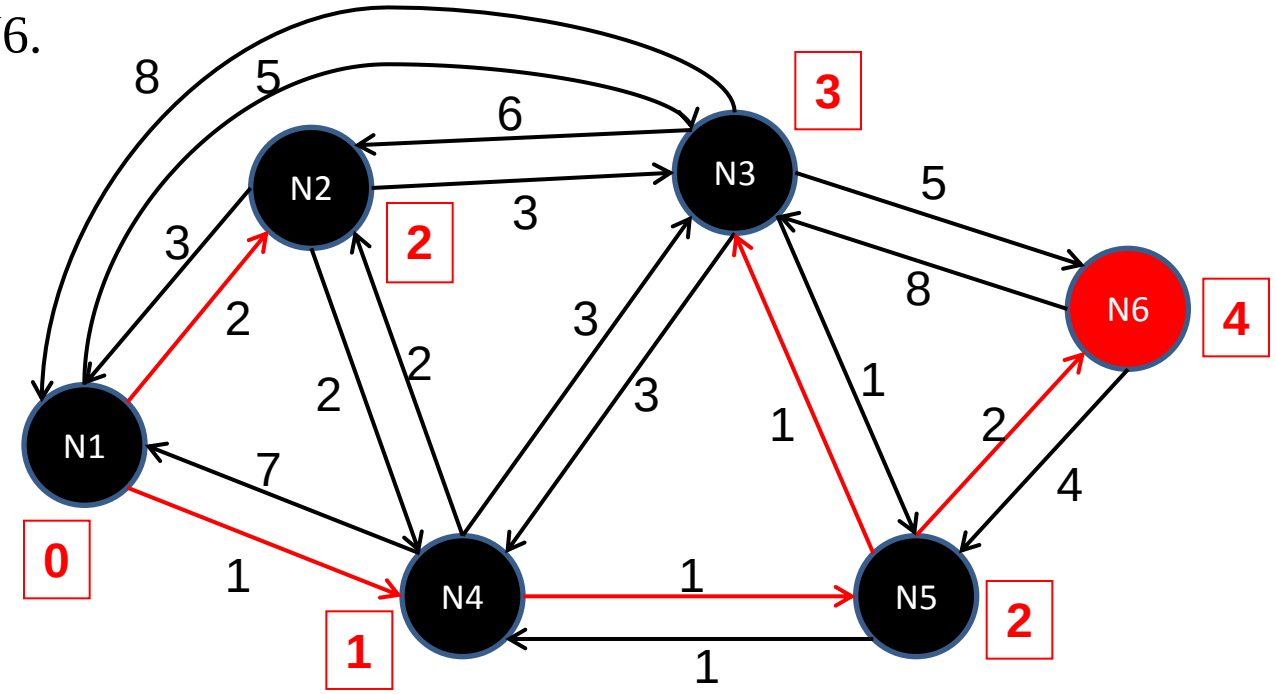
- Τώρα ήρθε η ώρα του N3. Καμία όμως ανανέωση.



Επανάληψη	T	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	2 (1-2)	5 (1-3)	1 (1-4)	∞ (-)	∞ (-)
2	{1,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
3	{1,2,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
4	{1,2,4,5}	2 (1-2)	3 (1-4-5-3)	1 (1-4)	2 (1-4-5)	4 (1-4-5-6)
5	{1,2,3,4,5}	2 (1-2)	3 (1-4-5-3)	1 (1-4)	2 (1-4-5)	4 (1-4-5-6)

Αλγόριθμος Dijkstra

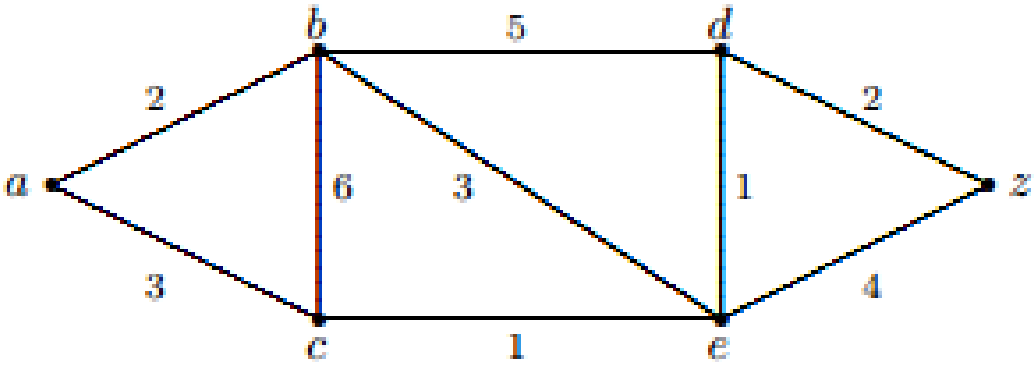
- Τέλος, ο N6.



Επανάληψη	T	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	2 (1-2)	5 (1-3)	1 (1-4)	∞ (-)	∞ (-)
2	{1,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
3	{1,2,4}	2 (1-2)	4 (1-4-3)	1 (1-4)	2 (1-4-5)	∞ (-)
4	{1,2,4,5}	2 (1-2)	3 (1-4-5-3)	1 (1-4)	2 (1-4-5)	4 (1-4-5-6)
5	{1,2,3,4,5}	2 (1-2)	3 (1-4-5-3)	1 (1-4)	2 (1-4-5)	4 (1-4-5-6)
6	{1,2,3,4,5,6}	2 (1-2)	3 (1-4-5-3)	1 (1-4)	2 (1-4-5)	4 (1-4-5-6)

Αλγόριθμος Dijkstra

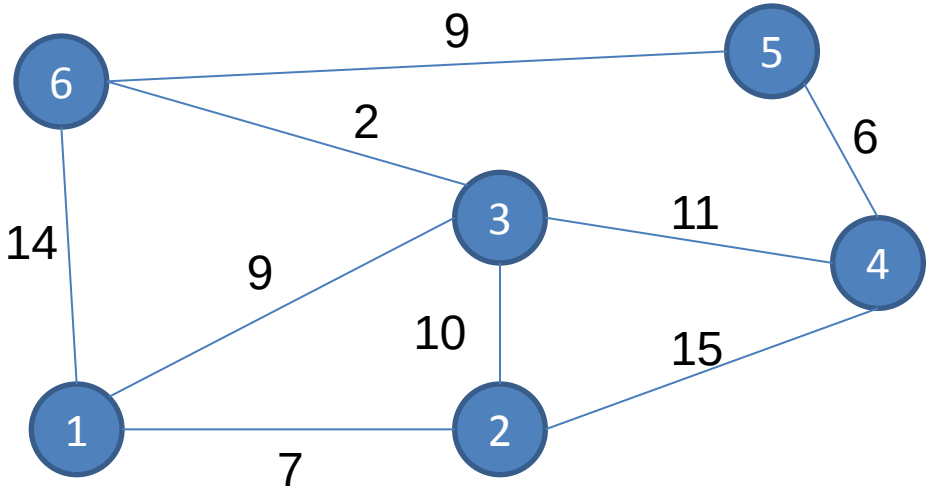
• Να φτιάξετε τον πίνακα του αλγορίθμου Dijkstra για το παρακάτω δίκτυο ξεκινώντας από τον κόμβο ***a***. (Οι ακμές είναι αμφικατευθυντικές με το ίδιο βάρος).



S	$L(a)$	$L(b)$	$L(c)$	$L(d)$	$L(e)$	$L(z)$
$\emptyset$	$\boxed{0}$	∞	∞	∞	∞	∞
$\{a\}$		$\boxed{2}$	3	∞	∞	∞
$\{a,b\}$			$\boxed{3}$	7	5	∞
$\{a,b,c\}$				7	$\boxed{4}$	∞
$\{a,b,c,e\}$				$\boxed{5}$		8
$\{a,b,c,e,d\}$						$\boxed{7}$
$\{a,b,c,e,d,z\}$						

Αλγόριθμος Dijkstra

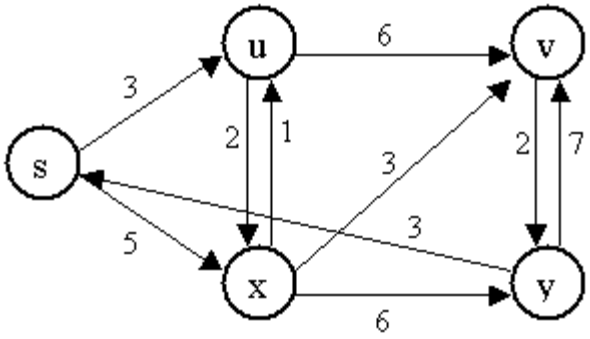
• Να φτιάξετε τον πίνακα του αλγορίθμου Dijkstra για το παρακάτω δίκτυο ξεκινώντας από τον κόμβο 1. (Οι ακμές είναι αμφικατευθυντικές με το ίδιο βάρος).



Επανάληψη	T	L(2)	L(3)	L(4)	L(5)	L(6)
1	{1}	7 (1-2)	9 (1-3)	∞ (-)	∞ (-)	14 (1-6)
2	{1,2}	7 (1-2)	9 (1-3)	22 (1-2-4)	∞ (-)	14 (1-6)
3	{1,2,4}	7 (1-2)	9 (1-3)	20 (1-3-4)	∞ (-)	11 (1-3-6)
4	{1,2,4,5}	7 (1-2)	9 (1-3)	20 (1-3-4)	20 (1-3-6-5)	11 (1-3-6)
5	{1,2,3,4,5}	7 (1-2)	9 (1-3)	20 (1-3-4)	20 (1-3-6-5)	11 (1-3-6)
6	{1,2,3,4,5,6}	7 (1-2)	9 (1-3)	20 (1-3-4)	20 (1-3-6-5)	11 (1-3-6)

Αλγόριθμος Dijkstra

• Να φτιάξετε τον πίνακα του αλγορίθμου Dijkstra για το παρακάτω δίκτυο ξεκινώντας από τον κόμβο s. (Οι ακμές είναι αμφικατευθυντικές με το ίδιο βάρος).

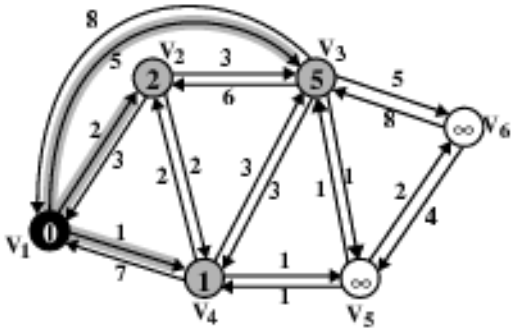


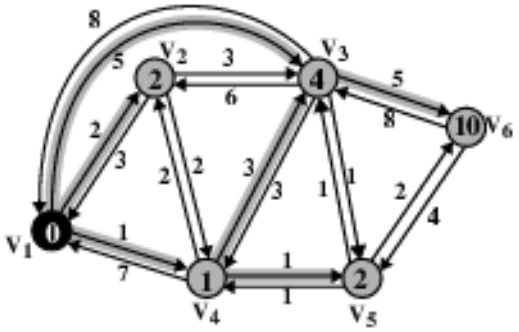
Επανάληψη	T	L(u)	L(x)	L(v)	L(y)
1	{s}	3 (s-u)	5 (s-x)	∞ (-)	∞ (-)
2	{s,u}	3 (s-u)	5 (s-x)	9 (s-u-v)	∞ (-)
3	{s,u,x}	3 (s-u)	5 (s-x)	8 (s-x-v)	11 (s-x-y)
4	{s,u,x,v}	3 (s-u)	5 (s-x)	8 (s-x-v)	10 (s-x-v-y)
5	{s,u,x,v,y}	3 (s-u)	5 (s-x)	8 (s-x-v)	10 (s-x-v-y)

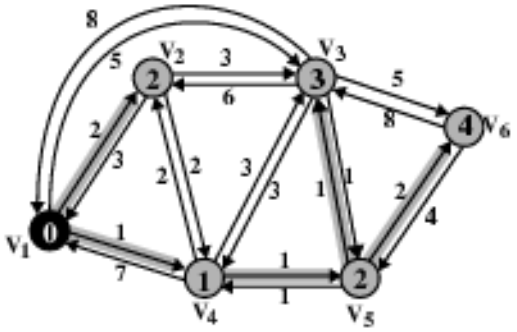
Αλγόριθμος Bellman-Ford

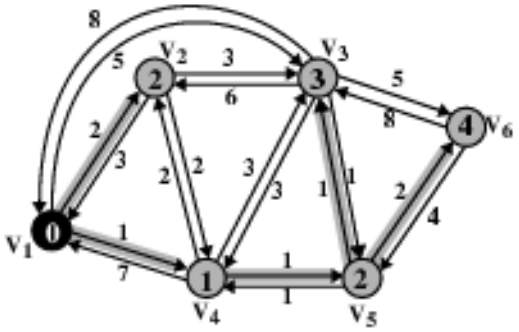
- Ο αλγόριθμος Bellman-Ford με λίγα λόγια:
  - Ο αλγόριθμος εκκινεί από έναν ριζικό κόμβο s.
  - Βρίσκει την συντομότερη διαδρομή προς τους κόμβους με την προϋπόθεση ότι περιέχουν το πολύ **μια** ζεύξη.
  - Κατόπιν βρίσκει την συντομότερη διαδρομή προς τους κόμβους με την προϋπόθεση ότι περιέχουν το πολύ **δύο** ζεύξεις.
  - Κατόπιν βρίσκει την συντομότερη διαδρομή προς τους κόμβους με την προϋπόθεση ότι περιέχουν το πολύ **τρείς** ζεύξεις.
  - Συνεχίζει κατά τον τρόπο αυτό για n-1 φορές όπου n το πλήθος των κόμβων.
  - Αρχικά όλοι οι κόμβοι n-1 θεωρούνται ότι έχουν “άπειρη” απόσταση από τον s.

Αλγόριθμος Bellman-Ford









h	$L_h(2)$	Path	$L_h(3)$	Path	$L_h(4)$	Path	$L_h(5)$	Path	$L_h(6)$	Path
0	∞	—	∞	—	∞	—	∞	—	∞	—
1	2	1-2	5	1-3	1	1-4	∞	—	∞	—
2	2	1-2	4	1-4-3	1	1-4	2	1-4-5	10	1-3-6
3	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
4	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6