ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΜΣ "ΠΡΟΗΓΜΕΝΑ ΣΥΣΤΗΜΑΤΑ ΠΛΗΡΟΦΟΡΙΚΗΣ -
ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΚΑΙ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ"

Σημειώσεις του μαθήματος

ΕΙΔΙΚΑ ΘΕΜΑΤΑ ΘΕΩΡΙΑΣ ΚΑΙ ΕΦΑΡΜΟΓΩΝ ΓΡΑΦΗΜΑΤΩΝ

Κ. Μανές - Ι. Τασούλας

Σημειώσεις διαλέξεων 10

ΠΕΙΡΑΙΑΣ 2023

Οι παρούσες σημειώσεις βασίζονται σε προηγούμενες σημειώσεις του μαθήματος που έχουν συγγράψει ο Καθηγητής κ. Αριστείδης Σαπουνάκης και ο Καθηγητής κ. Παναγιώτης-Γεώργιος Τσικούρας.

30. ΣΥΝΑΡΤΗΣΗ GRUNDY - SPRAGUE

Έστω $A \subseteq \mathbb{N}$. Ορίζουμε το mex **(minimum excluded value)** του A ως εξής:

$$\operatorname{mex} A = \min \mathbb{N} \setminus A,$$

δηλαδή $\operatorname{mex} A$ **είναι ο ελάχιστος φυσικός αριθμός που δεν ανήκει στο** A.
Παράδειγμα:

- $\operatorname{mex}\{1, 2, 4\} = 0$
- $\operatorname{mex}\{0, 1, 2, 6\} = 3$
- $\operatorname{mex}\{0, 1, 2, 3\} = 4$
- $\operatorname{mex} \emptyset = 0$.

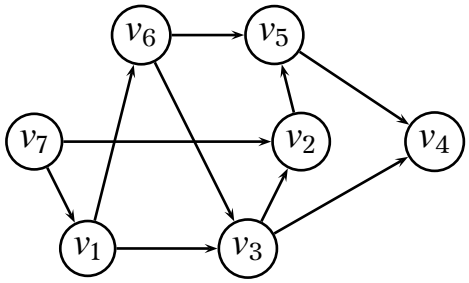
Για κάθε κορυφή v ενός γραφήματος τόξων $G = (V, U)$ με $\Gamma(v)$ συμβολίζουμε το σύνολο των κορυφών u που είναι άκρα τόξων με αρχή την κορυφή v (**γείτονες της** v), δηλαδή

$$\Gamma(v) = \{u \in X : (v, u) \in U\}$$

Παράδειγμα: Για το γράφημα τόξων $G = (V, U)$ όπου $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ και

$$U = \{(v_1, v_3), (v_1, v_6), (v_2, v_5), (v_3, v_2), (v_3, v_4), (v_5, v_4), (v_6, v_3), (v_6, v_5), (v_7, v_1), (v_7, v_2)\}.$$

Έχουμε ότι



$$\begin{aligned}
\Gamma(v_1) &= \{v_3, v_6\} \\
\Gamma(v_2) &= \{v_5\} \\
\Gamma(v_3) &= \{v_2, v_4\} \\
\Gamma(v_4) &= \emptyset \\
\Gamma(v_5) &= \{v_4\} \\
\Gamma(v_6) &= \{v_3, v_5\} \\
\Gamma(v_7) &= \{v_1, v_2\}.
\end{aligned}$$

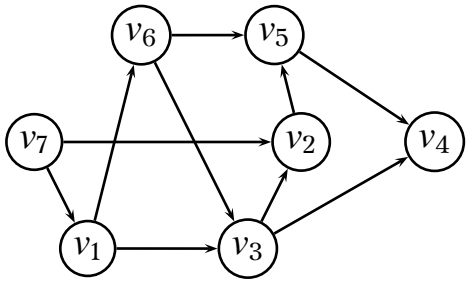
Σε κάθε γράφημα τόξων $G = (V, U)$ **χωρίς κυκλώματα** αντιστοιχεί μια (μοναδική) συνάρτηση $g : V \to \mathbb{N}$ η οποία ονομάζεται **συνάρτηση Grundy - Sprague** του G.

Συγκεκριμένα, για κάθε κορυφή v του γραφήματος ορίζουμε

$$g(v) = \operatorname{mex}\{g(u) : \text{ για κάθε } u \in \Gamma(v)\}.$$

Η συνάρτηση g τοποθετεί στις κορυφές του γραφήματος G φυσικούς αριθμούς, έτσι ώστε κάθε κορυφή v να έχει τιμή $g(v)$ τον ελάχιστο φυσικό αριθμό που δεν έχει τοποθετηθεί στους γείτονές της. Η τιμή $g(v)$ ονομάζεται g-**value** της κορυφής v.

Παράδειγμα: Να βρεθούν οι τιμές της συνάρτησης g του γραφήματος τόξων G:



$$\begin{aligned}
\Gamma(v_1) &= \{v_3, v_6\}\\
\Gamma(v_2) &= \{v_5\}\\
\Gamma(v_3) &= \{v_2, v_4\}\\
\Gamma(v_4) &= \emptyset\\
\Gamma(v_5) &= \{v_4\}\\
\Gamma(v_6) &= \{v_3, v_5\}\\
\Gamma(v_7) &= \{v_1, v_2\}.
\end{aligned}$$

Το G δεν έχει κυκλώματα, άρα ορίζεται η συνάρτηση Grundy - Sprague σ' αυτό. Στόχος μας είναι να συμπληρώσουμε τις κενές θέσεις στον επόμενο πίνακα

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$							

$$g(v) = \text{mex}\{g(u) : \text{ για κάθε } u \in \Gamma(v)\},$$

Έχουμε ότι

$$\begin{aligned}
\Gamma(v_1) &= \{v_3, v_6\}\\
\Gamma(v_2) &= \{v_5\}\\
\Gamma(v_3) &= \{v_2, v_4\}\\
\Gamma(v_4) &= \emptyset\\
\Gamma(v_5) &= \{v_4\}\\
\Gamma(v_6) &= \{v_3, v_5\}\\
\Gamma(v_7) &= \{v_1, v_2\}.
\end{aligned}$$

οπότε

$$\begin{aligned}
g(v_1) &= \text{mex}\{g(v_3), g(v_6)\}\\
g(v_2) &= \text{mex}\{g(v_5)\}\\
g(v_3) &= \text{mex}\{g(v_2), g(v_4)\}\\
g(v_4) &= \text{mex}\emptyset\\
g(v_5) &= \text{mex}\{g(v_4)\}\\
g(v_6) &= \text{mex}\{g(v_3), g(v_5)\}\\
g(v_7) &= \text{mex}\{g(v_1), g(v_2)\}.
\end{aligned}$$

Παρατηρούμε ότι:
Για να υπολογισθεί το $g(v_1)$ πρέπει να υπολογισθούν πρώτα τα $g(v_3)$, $g(v_6)$.
Για να υπολογισθεί το $g(v_2)$ πρέπει να υπολογισθεί πρώτα το $g(v_5)$,
Για να υπολογισθεί το $g(v_3)$ πρέπει να υπολογισθούν πρώτα τα $g(v_2)$, $g(v_4)$.
Για να υπολογισθεί το $g(v_5)$ πρέπει να υπολογισθεί πρώτα το $g(v_4)$,
Για να υπολογισθεί το $g(v_6)$ πρέπει να υπολογισθούν πρώτα τα $g(v_3)$, $g(v_5)$.
Για να υπολογισθεί το $g(v_7)$ πρέπει να υπολογισθούν πρώτα τα $g(v_1)$, $g(v_2)$.
Αντίθετα, για τον υπολογισμό του $g(v_4)$ δεν χρειάζονται άλλες τιμές της g και ισχύει ότι $g(v_4) = \text{mex}\emptyset = 0$.

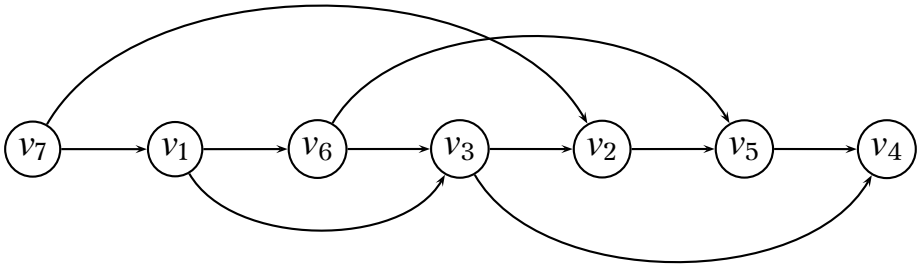
Επομένως, για να υπολογίσουμε τις τιμές της συνάρτησης g πρέπει να βρούμε (αν υπάρχει) μια σειρά στις κορυφές του, σύμφωνα με την οποία να υπολογίζουμε την τιμή $g(v)$ μια κορυφής αφού πρώτα έχουμε ήδη υπολογίσει τις τιμές $g(u)$ όλων των γειτόνων u της v.

Συνεπώς, ο υπολογισμός της συνάρτησης g είναι πρόβλημα προτεραιοτήτων, επομένως μπορεί να λυθεί υπολογίζοντας μια τοπολογική διάταξη στο σύνολο των κορυφών του γραφήματος G. Η διαφοροποίηση στο πρόβλημα αυτό είναι ότι

το τόξο (v, u) λειτουργεί ανάποδα, αφού για τον υπολογισμό της $g(v)$ χρειαζόμαστε την τιμή $g(u)$.

Άρα, για το πρόβλημα αυτό, θα βρούμε μια τοπολογική διάταξη στο γράφημα G και έπειτα θα υπολογίσουμε την συνάρτηση με την ανάστροφη σειρά.

Μια τέτοια διάταξη απεικονίζεται στο επόμενο σχήμα



(από την κορυφή v_i ξεκινάει βέλος προς την κορυφή v_j αν ο υπολογισμός της $g(v_i)$ απαιτεί τον υπολογισμό της $g(v_j)$).

Θα υπολογίσουμε τις τιμές της συνάρτησης g με την ανάστροφη σειρά

$$v_4, v_5, v_2, v_3, v_6, v_1, v_7$$

οπότε σε κάθε βήμα οι τιμές της g που απαιτούνται έχουν υπολογισθεί σε κάποιο προηγούμενο βήμα.

Πράγματι, έχουμε τα εξής:

1) $g(v_4) = \text{mex}\emptyset = 0$.

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$				0			

2) $g(v_5) = \text{mex}\{g(v_4)\} = \text{mex}\{0\} = 1$.

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$				0	1		

3) $g(v_2) = \text{mex}\{g(v_5)\} = \text{mex}\{1\} = 0$.

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$		0		0	1		

4) $g(v_3) = \text{mex}\{g(v_2), g(v_4)\} = \text{mex}\{0, 0\} = \text{mex}\{0\} = 1$.

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$		0	1	0	1		

5) $g(v_6) = \text{mex}\{g(v_3), g(v_5)\} = \text{mex}\{1, 1\} = \text{mex}\{1\} = 0$.

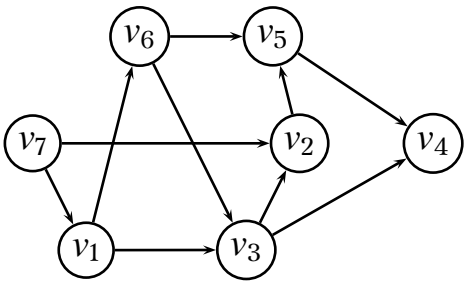
v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$		0	1	0	1	0	

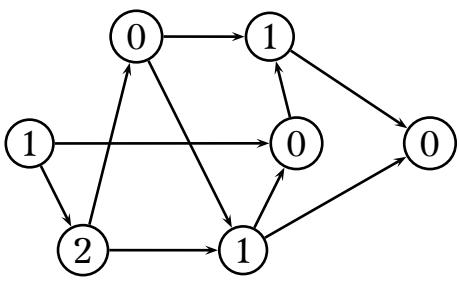
6) $g(v_1) = \text{mex}\{g(v_3), g(v_6)\} = \text{mex}\{1, 0\} = 2$.

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$	2	0	1	0	1	0	

7) Τέλος, $g(v_7) = \text{mex}\{g(v_1), g(v_2)\} = \text{mex}\{2, 0\} = 1$.

v	v_1	v_2	v_3	v_4	v_5	v_6	v_7
$g(v)$	2	0	1	0	1	0	1.





Παρατήρηση: Η συνάρτηση Grundy - Sprague ικανοποιεί τις παρακάτω ιδιότητες

- Αν για την κορυφή v ισχύει ότι $g(v) \neq 0$, τότε υπάρχει $u \in \Gamma(v)$ με $g(u) = 0$.
- Αν για την κορυφή v ισχύει ότι $g(v) = 0$, τότε για κάθε $u \in \Gamma(v)$ ισχύει ότι $g(u) \neq 0$.

Μια εφαρμογή της συνάρτησης Grundy - Sprague είναι ότι

- αποδεικνύει ότι υπάρχει στρατηγική νίκης σε συνδυαστικά παιχνίδια δύο παιχτών όπου έχουμε αποκλείσει την ισοπαλία (π.χ. σκάκι), δηλαδή αποδεικνύει ότι κάποιος από τους δύο παίχτες μπορεί πάντα να κερδίζει (αρκεί να γνωρίζει την στρατηγική).
- δίνει την στρατηγική νίκης στο παιχνίδι αυτό, αρκεί να είναι υπολογιστικά εφικτός ο υπολογισμός της συνάρτησης g. Η στρατηγική νίκης είναι (πάντα) να φτάνουμε σε καταστάσεις με τιμή g ίση με 0.

Άσκηση 13 (Μια παραλλαγή του Nim).
Δύο παίχτες Α και Β παίζουν το εξής παιχνίδι:

- *Υπάρχουν δύο σωροί με σπίρτα.*
  Ο πρώτος σωρός αποτελείται από x σπίρτα.
  Ο δεύτερος σωρός αποτελείται από y σπίρτα.
- *Οι παίχτες παίζουν εναλλάξ κάνοντας* **μια κίνηση ο καθένας***: Κάθε παίχτης αφαιρεί ένα ή περισσότερα σπίρτα*
  - *ή από τον πρώτο σωρό,*
  - *ή από τον δεύτερο σωρό,*
  - *ή τον ίδιο αριθμό σπίρτων και από τους δύο σωρούς.*
- **Χάνει** *ο παίχτης που δεν μπορεί να κάνει κίνηση.*
- *Για παράδειγμα, αν ο πρώτος σωρός περιέχει 12 σπίρτα και ο δεύτερος σωρός 8 σπίρτα ποιος από τους δύο παίχτες θα κερδίσει;*

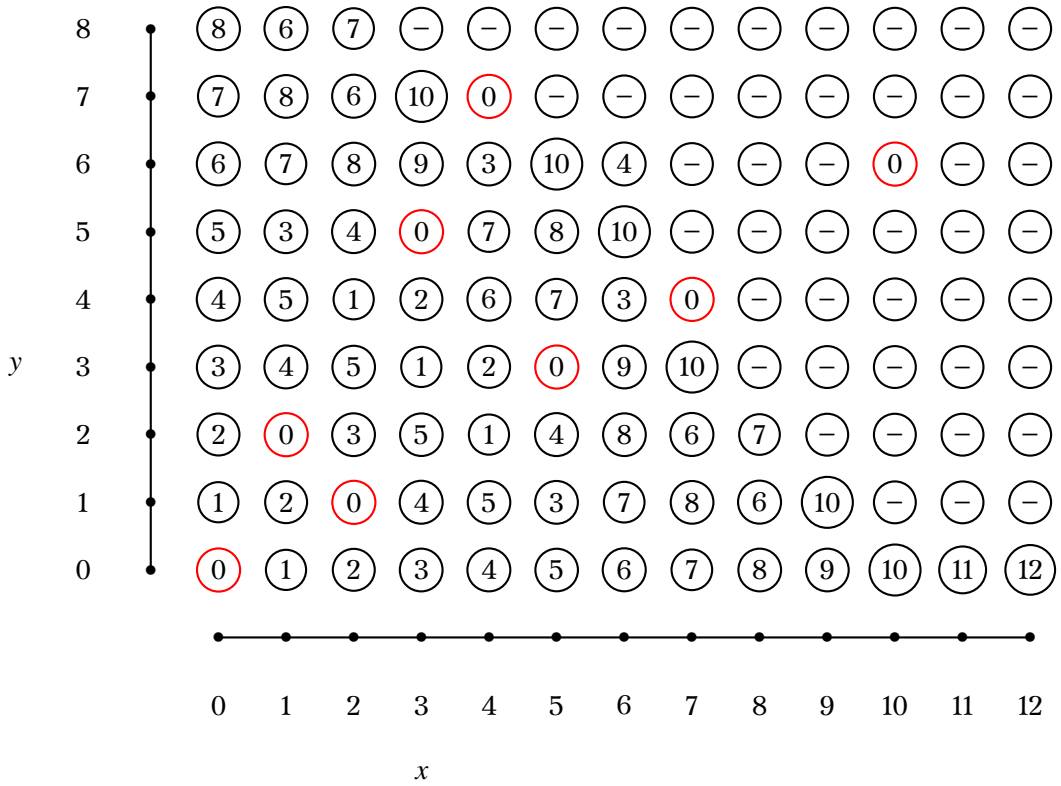
Λύση. Μπορούμε να μοντελοποιήσουμε το παιχνίδι ως μια ακολουθία καταστάσεων (ένα μονοπάτι κορυφών πάνω ένα γράφημα τόξων).

- Οι κορυφές του γραφήματος είναι οι καταστάσεις του παιχνιδιού: Κάθε κορυφή αντιστοιχεί σε ένα διατεταγμένο ζεύγος (x, y) όπου x ο αριθμός των σπίρτων στον πρώτο σωρό και y ο αριθμός των σπίρτων στον δεύτερο σωρό.

  Ισχύει ότι $0 \leq x \leq 12$ και $0 \leq y \leq 8$. Συνολικά υπάρχουν $13 \times 9 = 117$ καταστάσεις (κορυφές).
- Αρχικά το παιχνίδι βρίσκεται στην κατάσταση (κορυφή) $(12, 8)$. Με κάθε κίνηση των παιχτών το παιχνίδι αλλάζει καταστάσεις. Οι επιτρεπτές κινήσεις αντιστοιχούν στην μετακίνηση οριζόντια (προς τα αριστερά) ή κατακόρυφα (προς τα κάτω) ή διαγώνια (προς το κέντρο) οσαδήποτε βήματα. Κερδίζει ο παίχτης που θα φτάσει πρώτος στην κορυφή $(0, 0)$.

- Στο επόμενο σχήμα απεικονίζονται μόνο οι κορυφές του γραφήματος. Τα τόξα ορίζονται από τις επιτρεπτές κινήσεις.

  Επίσης, έχουν σημειωθεί οι τιμές της συνάρτησης Grundy - Sprague. Οι μηδενικές τιμές (τιμές με κόκκινο) ορίζουν την νικηφόρα στρατηγική του παιχνιδιού.



- Στο παράδειγμα, μπορεί να κερδίζει πάντα ο πρώτος παίχτης, διότι αφαιρώντας 2 σπίρτα και από τους δύο σωρούς καταλήγει στην κατάσταση $(10, 6)$ η οποία έχει τιμή g ίση με 0.

□

Παρατήρηση: Σε κάθε συνδυαστικό παιχνίδι δύο παιχτών (χωρίς ισοπαλία) μπορούμε να κατασκευάσουμε το γράφημα των καταστάσεων του παιχνιδιού. Το γράφημα αυτό δεν θα περιέχει κυκλώματα (αφού δεν επιτρέπεται ισοπαλία).

Πόρισμα 65. *Κάθε συνδυαστικό παιχνίδι δύο παιχτών (χωρίς ισοπαλία) έχει συνάρτηση g. Επομένως, μπορεί να κερδίζει πάντα ή ο πρώτος, ή ο δεύτερος παίχτης. (Αρκεί να γνωρίζει τις καταστάσεις που μηδενίζουν την συνάρτηση g.)*

31. ΔΙΑΤΑΞΗ ΠΑΡΑΓΩΓΗΣ - ΣΤΑΘΜΕΣ

Ένας εναλλακτικός τρόπος υπολογισμού της τοπολογικής διάταξης είναι χρησιμοποιώντας την μέθοδο Demoucron:

Σε κάθε γράφημα τόξων με p κορυφές σχηματίζουμε ένα πίνακα (με p γραμμές) ως εξής :

Στις πρώτες p στήλες $v_1, v_2, \ldots, v_p$ τοποθετούμε 0 και 1, όπως ακριβώς στη μήτρα γειτονικότητας του γραφήματος, (συνήθως παραλείπουμε τα 0).

Τις επόμενες στήλες $S_0, S_1, \ldots$, τις συμπληρώνουμε διαδοχικά, χρησιμοποιώντας την εξής αναδρομική διαδικασία:

Στη στήλη S_0 γράφουμε στις αντίστοιχες γραμμές το άθροισμα των 1 κάθε γραμμής (δηλαδή, τους βαθμούς εξόδου των κόμβων $v_1, v_2, \ldots, v_p$). Γράφουμε κάτω από τον πίνακα τους κόμβους με βαθμό εξόδου 0. Οι κόμβοι αυτοί θα τοποθετηθούν στην τελευταία στάθμη. (Δεν θα ασχοληθούμε άλλο με τις γραμμές που αντιστοιχούν στους κόμβους αυτούς. Γράφουμε $\times$ σε κάθε στήλη, δεξιά από κάθε 0).

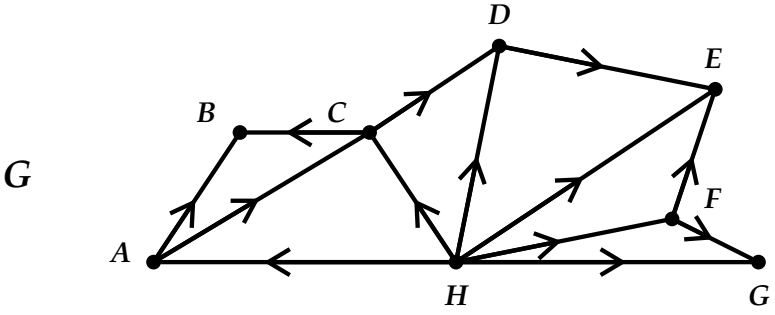
Έστω τώρα, ότι έχουμε συμπληρώσει μέχρι και τη στήλη S_n, ($n \geq 0$) και έχουμε γράψει κάτω από τον πίνακα και τους κόμβους $v_i, \ldots, v_j$ που αντιστοιχούν στα 0 της στήλης S_n, (οι οποίοι θα είναι οι κόμβοι στην n-στή πριν από το τέλος στάθμη).

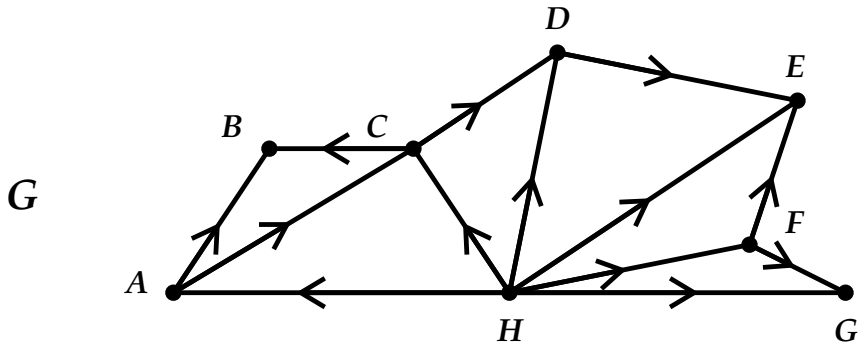
Στη στήλη S_{n+1} γράφουμε τα στοιχεία της στήλης S_n, καθένα μειωμένο κατά τόσες μονάδες, όσα είναι τα 1 που περιέχονται στις στήλες $v_i, \ldots, v_j$ της αντίστοιχης γραμμής. Οι κόμβοι που αντιστοιχούν στα 0 της στήλης αυτής, είναι τα στοιχεία της $(n+1)$-στής πριν από το τέλος στάθμης.

Η διαδικασία ολοκληρώνεται όταν σχηματισθεί μια στήλη S_l που αποτελείται μόνο από 0 και $\times$.

Παράδειγμα : Έστω A, B, C, D, E, F, G, H οι τεχνολογικές διαδικασίες, που υπόκεινται σε μια σχέση "τεχνολογικής προτεραιότητας" ως εξής : $A < B$, $A < C$, $C < B$, $C < D$, $D < E$, $F < E$, $F < G$, $H < A$, $H < C$, $H < D$, $H < E$, $H < F$, $H < G$, (γράφουμε $x < y$, όταν η δραστηριότητα x προηγείται της y).

Σχηματίζουμε το αντίστοιχο γράφημα τόξων G :



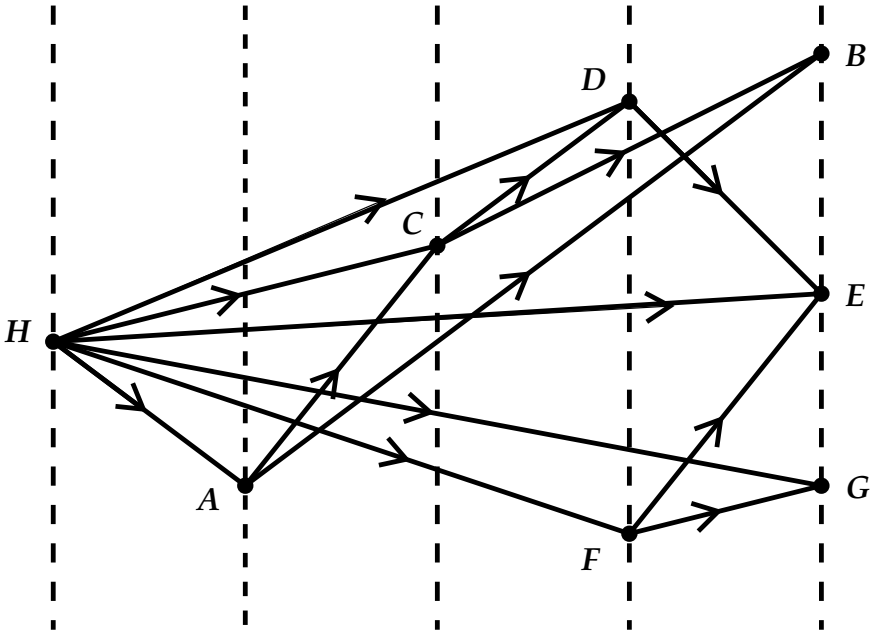


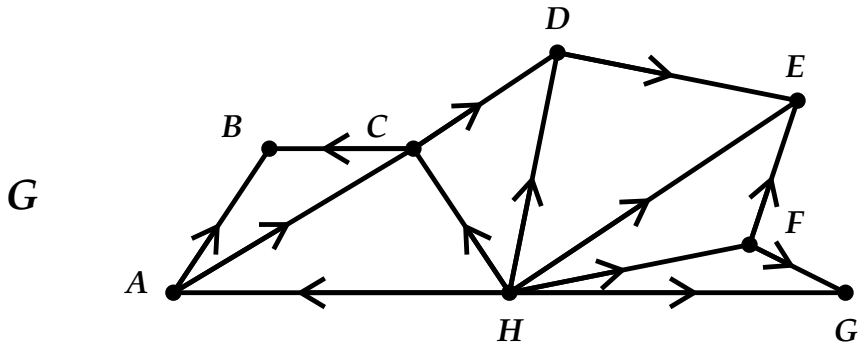
Η παραπάνω διαδικασία δίνει :

	A	B	C	D	E	F	G	H	S_0	S_1	S_2	S_3	S_4
A													
B													
C													
D													
E													
F													
G													
H													

Τα B, E, G βρίσκονται στην τελευταία στάθμη.
Τα D, F βρίσκονται στην προτελευταία στάθμη.
Το C βρίσκεται στην τρίτη από το τέλος στάθμη.
Το A βρίσκεται στην τετάρτη από το τέλος στάθμη.
Το H βρίσκεται στην πρώτη στάθμη,

οπότε



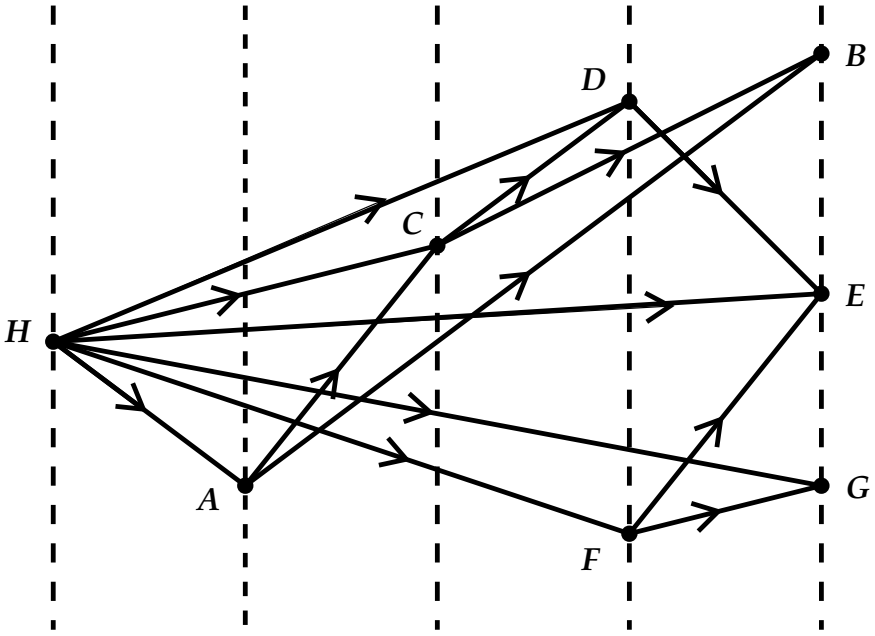


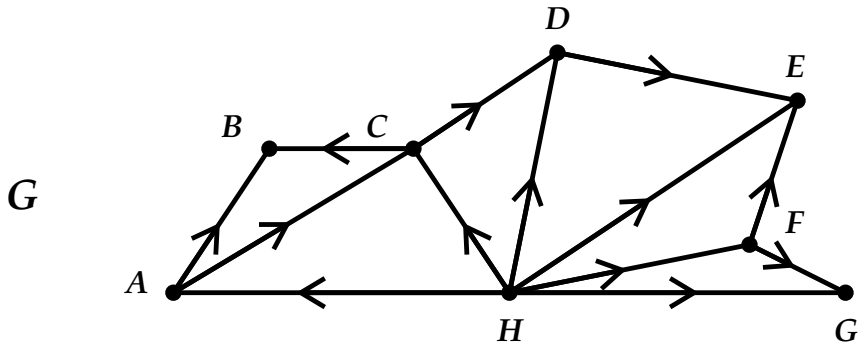
Η παραπάνω διαδικασία δίνει :

	A	B	C	D	E	F	G	H	S_0	S_1	S_2	S_3	S_4
A		1	1										
B													
C		1		1									
D					1								
E													
F					1		1						
G													
H	1		1	1	1	1	1						

Τα B, E, G βρίσκονται στην τελευταία στάθμη.
Τα D, F βρίσκονται στην προτελευταία στάθμη.
Το C βρίσκεται στην τρίτη από το τέλος στάθμη.
Το A βρίσκεται στην τετάρτη από το τέλος στάθμη.
Το H βρίσκεται στην πρώτη στάθμη,

οπότε



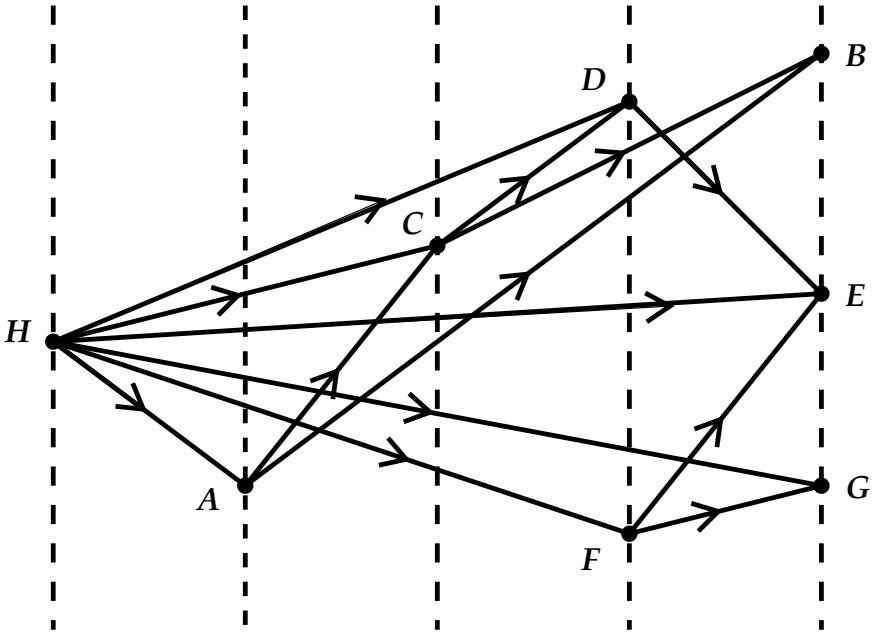


Η παραπάνω διαδικασία δίνει :

	A	B	C	D	E	F	G	H	S_0	S_1	S_2	S_3	S_4
A		1	1						2	1	1	0	×
B									0	×	×	×	×
C		1		1					2	1	0	×	×
D					1				1	0	×	×	×
E									0	×	×	×	×
F					1		1		2	0	×	×	×
G									0	×	×	×	×
H	1		1	1	1	1	1		6	4	2	1	0

Τα B, E, G βρίσκονται στην τελευταία στάθμη.
Τα D, F βρίσκονται στην προτελευταία στάθμη.
Το C βρίσκεται στην τρίτη από το τέλος στάθμη.
Το A βρίσκεται στην τετάρτη από το τέλος στάθμη.
Το H βρίσκεται στην πρώτη στάθμη,

οπότε



32. ΕΦΑΡΜΟΓΗ ΣΤΟ ΧΡΟΝΙΚΟ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (ΕΝΑ ΠΑΡΑΔΕΙΓΜΑ)

Υποθέτουμε ότι για τις εργασίες ενός έργου ισχύει ο παρακάτω πίνακας, όπου το i (αντίστοιχα το j) συμβολίζει την έναρξη (αντίστοιχα τη λήξη) μιας συγκεκριμένης δραστηριότητας σε ένα έργο, ενώ ο χρόνος t_{ij} είναι ο αναμενόμενος χρόνος για την πραγματοποίηση της δραστηριότητας (i, j) του έργου.

Δραστηριότητες		Χρόνος
i	j	t_{ij}
1	2	5
1	3	6
1	4	6
2	3	3
2	5	4
2	6	3
3	4	5
3	6	6
3	8	4
3	9	7
4	7	7
5	9	2
5	11	4
6	7	4
6	9	5
7	8	7
7	11	2
8	9	3
8	10	2
9	10	3
10	12	8
11	12	9

Θεωρούμε ότι η έναρξη της δραστηριότητας (j, k) λαμβάνει χώρα μόνο όταν έχουν πραγματοποιηθεί όλες οι δραστηριότητες (i, j), (για $j \neq 1$).

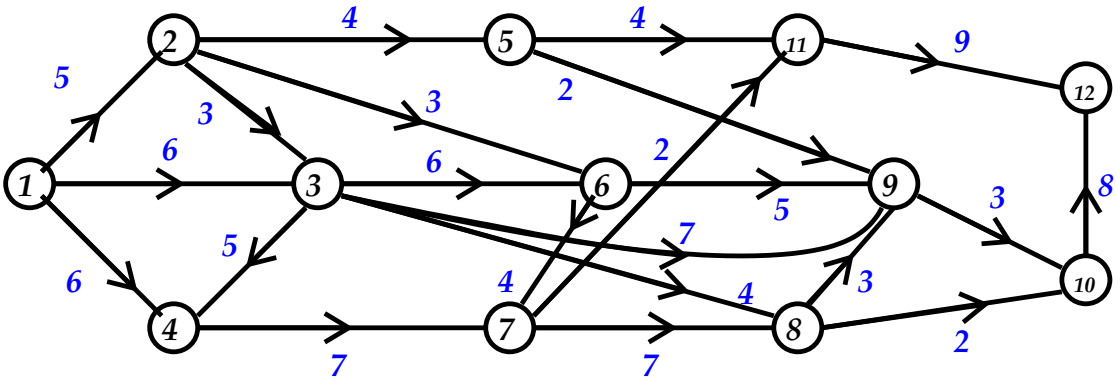
Από τα παραπάνω προκύπτει ένα γράφημα τόξων με κόμβους τις ενάρξεις και τις λήξεις των δραστηριοτήτων, τόξα τις αντίστοιχες δραστηριότητες και αριθμούς στα τόξα οι οποίοι δίνουν τους αναμενόμενους χρόνους για κάθε τέτοια δραστηριότητα. Ο κόμβος 1 δίνει την έναρξη και ο κόμβος 12 τη λήξη του έργου.

Το γράφημα δεν πρέπει να περιέχει κυκλώματα.

Ζητάμε τον ενωρίτερο χρόνο κατά τον οποίο μπορεί να ολοκληρωθεί το έργο. Αρκεί λοιπόν να βρούμε διαδοχικά τον ενωρίτερο χρόνο ολοκλήρωσης για να φτάσουμε σε κάθε κόμβο, υποθέτοντας ότι ο ενωρίτερος χρόνος για το 1 (έναρξη) είναι 0.

Ο ζητούμενος ενωρίτερος χρόνος ολοκλήρωσης του έργου, είναι λοιπόν προφανώς ο ενωρίτερος χρόνος για τον κόμβο 12 (λήξη του έργου).

Οι διαδοχικοί χρόνοι είναι σημειωμένοι δίπλα στην κάθε κορυφή του γραφήματος.



Βλέπουμε λοιπόν ότι ο ενωρίτερος χρόνος για την ολοκλήρωση του έργου με τα δεδομένα του προηγούμενου πίνακα είναι 41.

Ο δρόμος (1, 2, 3, 4, 7, 8, 9, 10, 12) που αντιστοιχεί στον ανωτέρω χρόνο ονομάζεται **κρίσιμος δρόμος** του έργου και οι αντίστοιχες δραστηριότητες **κρίσιμες δραστηριότητες**.

Άσκηση 14. *Πως θα αλλάξει ο χρόνος ολοκλήρωσης ενός έργου*

(1) αν όλοι οι χρόνοι των εργασιών πολλαπλασιαστούν επί μια σταθερά c;

(2) αν όλοι οι χρόνοι των εργασιών αυξηθούν κατά μια σταθερά c;

Ασκήσεις προς επίλυση

(1) Έστω $A, B, \Gamma, \Delta, E, Z$ οι τεχνολογικές διαδικασίες μιας παραγωγής, που υπόκεινται στη σχέση τεχνολογικής προτεραιότητας ως εξής:

$$A < \Delta,\ \ B < A,\ \ B < \Delta,\ \ B < E,\ \ B < Z,$$

$$\Gamma < B,\ \ \Gamma < E,\ \ E < \Delta,\ \ Z < A,\ \ Z < E$$

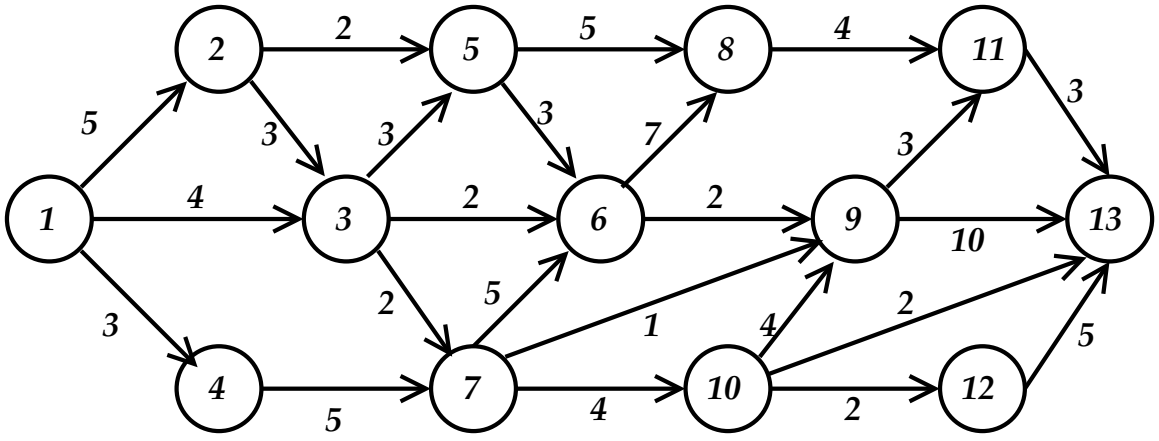
α) Να σχεδιαστεί το σχετικό γράφημα G.

β) Με εφαρμογή της μεθόδου Demoucron να ευρεθούν οι στάθμες των κορυφών του G.

γ) Να δοθεί γραφικά το σχετικό αποτέλεσμα, που περιγράφει την διάταξη παραγωγής.

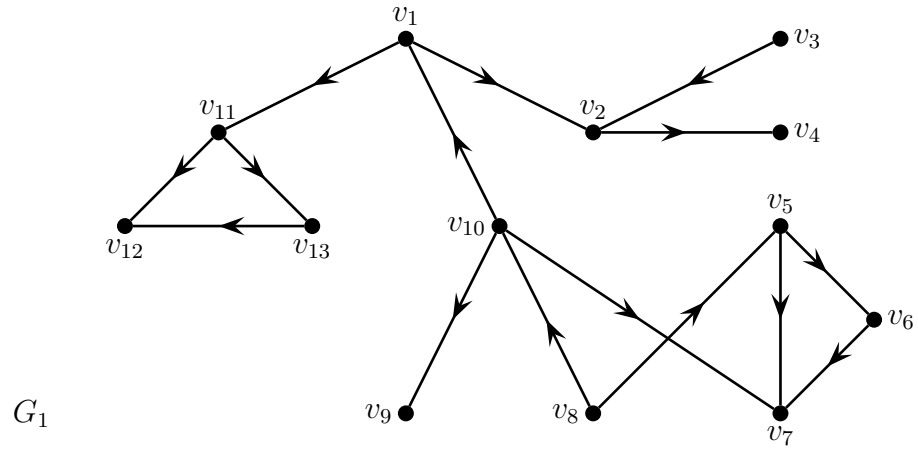
	Α	Β	Γ	Δ	Ε	Ζ						
Α												
Β												
Γ												
Δ												
Ε												
Ζ												

(2) Να βρεθεί ο ενωρίτερος χρόνος ολοκλήρωσης και ο κρίσιμος δρόμος του έργου που αρχίζει από τον κόμβο 1 και τελειώνει στον κόμβο 13, αν το έργο αυτό περιγράφεται από το παρακάτω γράφημα τόξων.

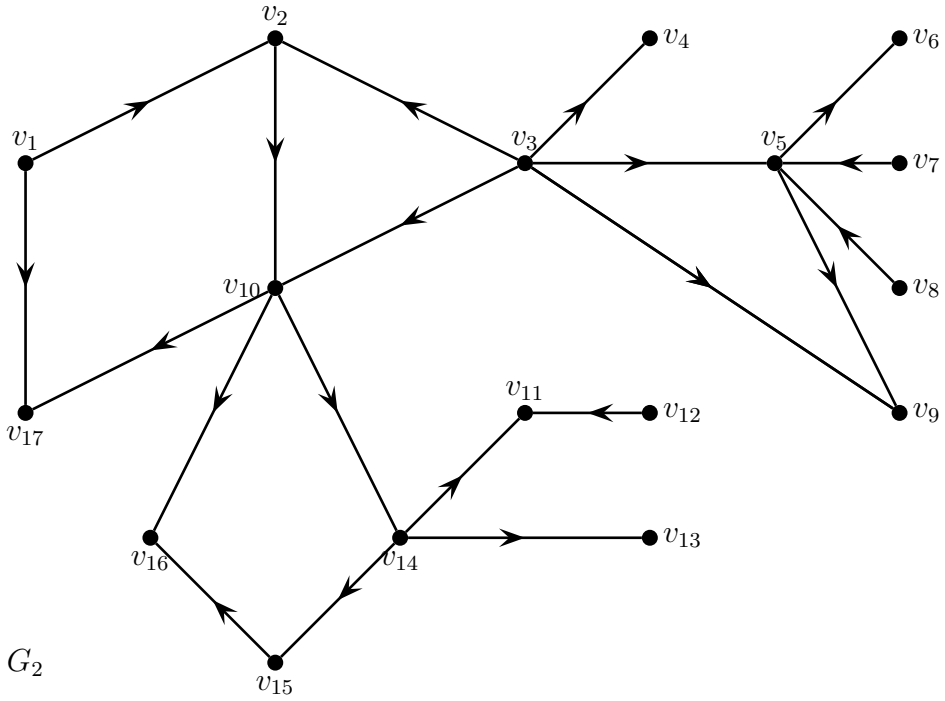


(Ο χρόνος για κάθε δραστηριότητα (i, j) σημειώνεται πάνω στο αντίστοιχο τόξο. Θεωρούμε ότι, για κάθε $j \neq 1$, η έναρξη της δραστηριότητας (j, k) λαμβάνει χώρα μόνο όταν έχουν πραγματοποιηθεί όλες οι δραστηριότητες (i, j)).

(3) Να βρεθεί η τοπολογική διάταξη για τα παρακάτω γραφήματα:

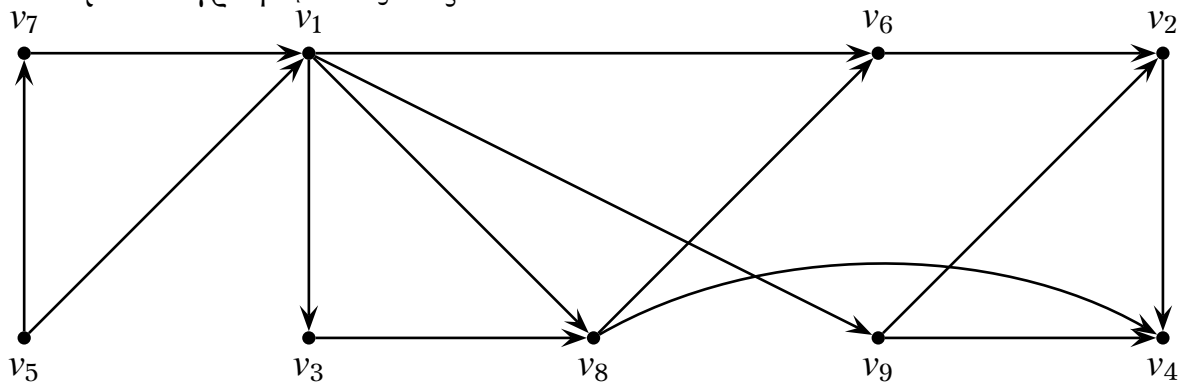


G_1



G_2

(4) Να υπολογισθούν οι τιμές της συνάρτησης Grundy - Sprague για τις κορυφές του επόμενου γραφήματος τόξων.



(5) Πάνω σε ένα τραπέζι βρίσκονται 25 σπίρτα. Δύο παίχτες κάνουν εναλλάξ την εξής κίνηση: Αφαιρούν 1, 2 ή 3 σπίρτα από το τραπέζι. Ο παίχτης που θα πάρει το τελευταίο σπίρτο χάνει. Να δειχθεί ότι ο δεύτερος παίχτης μπορεί πάντα να κερδίζει το παιχνίδι. (Δοκιμάστε να παίξετε το παιχνίδι με 5, 9 και 13 σπίρτα)

33. ΔΥΝΑΜΙΚΑ ΜΟΝΤΕΛΑ ΓΡΑΦΗΜΑΤΩΝ

Στην ενότητα αυτή παρουσιάζονται ορισμένα παραδείγματα που αφορούν γραφήματα τα οποία αλλάζουν δυναμικά (π.χ. προστίθενται ή/και διαγράφονται κορυφές ή/και δεσμοί, αλλάζουν οι ετικέτες των κορυφών ή/και των δεσμών).

33.1. ΔΙΚΤΥΑ ΕΠΙΡΡΟΗΣ. Τα δίκτυα επιρροής μοντελοποιούν το πως μια νέα ιδέα ή άποψη υιοθετείται ή απορρίπτεται από ένα δίκτυο ατόμων. Η μοντελοποίηση γίνεται με βάση την παραδοχή ότι η γνώμη που υιοθετεί ένα άτομο επηρεάζεται από την πίεση που του ασκούν οι επαφές του (peer pressure): Συγκεκριμένα, όσο μεγαλύτερο το ποσοστό των επαφών μας που μοιράζονται ή υιοθετούν μια ιδέα, τόσο πιο πιθανό είναι ότι θα την υιοθετήσουμε και εμείς.

Δίκτυο επιρροής είναι ένα γράφημα δεσμών (ή τόξων) $G = (V, E)$ στο οποίο κάθε κορυφή v έχει μια ετικέτα $l(v)$. Συγκεκριμένα, για κάθε κορυφή v είτε $l(v) = 1$ όταν η v αποδέχεται μια άποψη και χαρακτηρίζεται **ενεργή** είτε $l(v) = 0$, όταν η v δεν αποδέχεται την άποψη και χαρακτηρίζεται **μη ενεργή**.

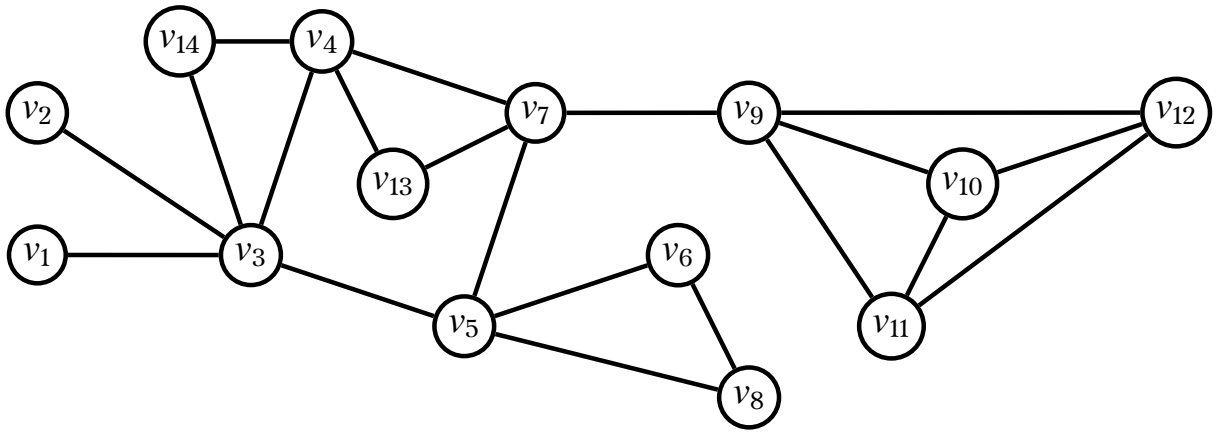
Το μοντέλο λειτουργεί δυναμικά (δηλαδή οι ετικέτες των κορυφών μπορούν να αλλάζουν). Ο βασικός μηχανισμός του μοντέλου περιγράφεται στα επόμενα βήματα:

- Αρχικά όλες οι κορυφές του γραφήματος $G = (V, E)$ είναι μη ενεργές, δηλαδή $l(v) = 0$ για κάθε $v \in V$.

  Επιλέγεται (είτε τυχαία, είτε μέσω κάποιας διαδικασίας) ένα υποσύνολο S των κορυφών του G οι οποίες ενεργοποιούνται, δηλαδή θέτουμε $l(v) = 1$ για κάθε $v \in S$.
- Επαναλαμβάνουμε τα επόμενα βήματα μέχρις ότου δεν υπάρχουν αλλαγές στις ετικέτες των κορυφών.
  - Αν μια κορυφή ειναι ενεργή, παραμένει ενεργή.
  - Αν μια κορυφή είναι μη ενεργή και στην προηγούμενη επανάληψη τουλάχιστον $\theta \cdot 100\%$ από τις γειτονικές[7] της κορυφές είναι ενεργές, τότε και αυτή γίνεται ενεργή (εναλλακτικά, γίνεται ενεργή με κάποια πιθανότητα).

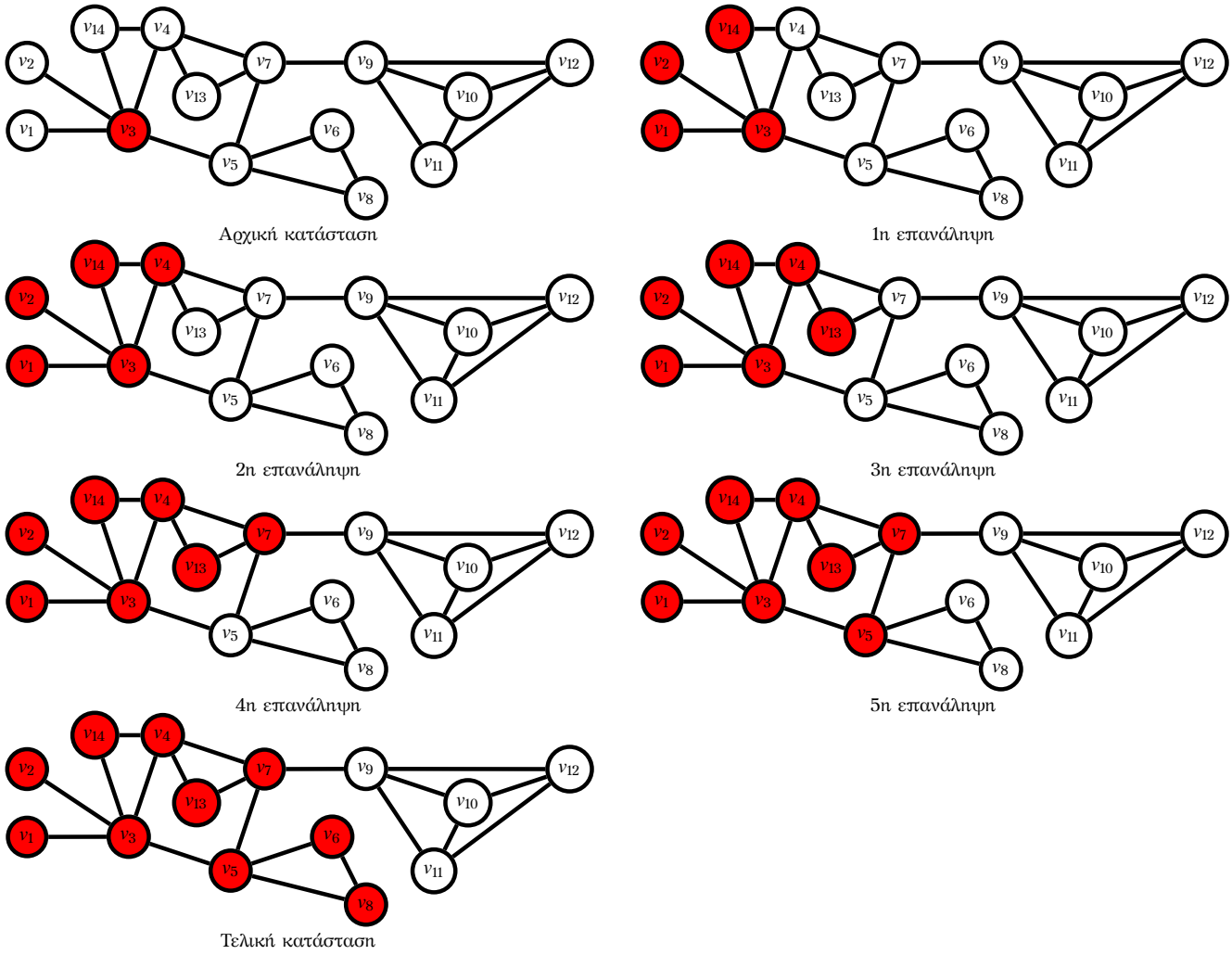
Ο αριθμός $\theta \in (0, 1]$ ονομάζεται **κατώφλι ενεργοποίησης** (activation threshold) και το συγκεκριμένο μοντέλο ονομάζεται **μοντέλο κατωφλίου** (threshold model).

Παράδειγμα:



[7] Σε γραφήματα τόξων, στο μοντέλο αυτό θεωρούμε ως γειτονικές τις κορυφές που είναι αρχή τόξων που εισέρχονται στην κορυφή.

Για παράδειγμα, αν επιλέξουμε $\theta = 50\%$ και ενεργοποιήσουμε την κορυφή v_3 του προηγούμενου γραφήματος, τότε στην πρώτη επανάληψη θα ενεργοποιηθούν οι κορυφές v_1, v_2, v_{14}. Στην επόμενη επανάληψη θα ενεργοποιηθεί η κορυφή v_4. Έπειτα, θα ενεργοποιηθεί η κορυφή v_{13}. Μετά, θα ενεργοποιηθεί η κορυφή v_7. Στην συνέχεια, θα ενεργοποιηθεί η κορυφή v_5. Ακολούθως, θα ενεργοποιηθούν οι κορυφές v_6, v_8. Η διαδικασία θα ολοκληρωθεί εδώ. Οι κορυφές v_9, v_{10}, v_{11}, v_{12} θα παραμείνουν μη ενεργές.

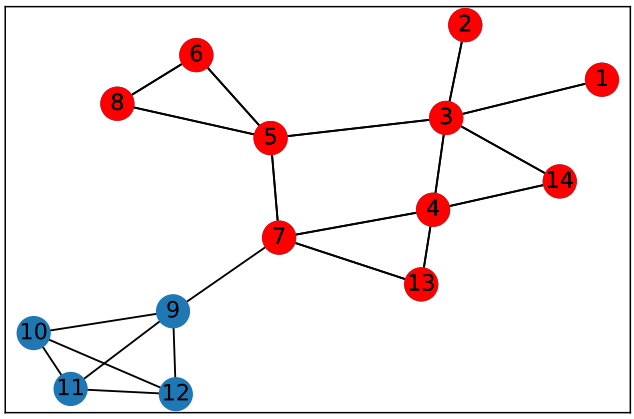


Μπορούμε εύκολα να δοκιμάσουμε το μοντέλο αυτό χρησιμοποιώντας την βιβλιοθήκη networkx.

```
import networkx as nx
import matplotlib.pyplot as plt

theta = 0.5 #threshold

G = nx.Graph()
n = 14 #A graph with 14 vertices
G.add_nodes_from(range(1,n+1))
E = [[1,3],[2,3],[3,4],[3,5],[3,14],[4,7],[4,13],[4,14],[5,7],[5,6],[5,8],
     [6,8],[7,9],[7,13],[9,10],[9,11],[9,12],[10,11],[10,12],[11,12]]
G.add_edges_from(E)

#Draw the initial graph before running the model
pos = nx.layout.kamada_kawai_layout(G)
```

```
nx.draw_networkx(G,pos)
#create the list of labels setting all labels equal to 0 (label[0] is not used)
#label[i] corresponds to the label of node i
labels = [0 for i in range(1,n+2)]
labels[3] = 1 #set node 3 active

changesMade = True
while(changesMade):
    changesMade = False
    for v in G:
        if(labels[v]==0): #if node v is inactive
            activeneighbors = 0
            for u in G.neighbors(v):
                if(labels[u]==1): activeneighbors+=1
            #if the percentage of active neigbhbors of v
            #is greater than or equal to theta then set v active
            if(activeneighbors/G.degree(v) >= theta):
                changesMade = True
                labels[v] = 1
                print("Message: Node",v,"is now active")

#list of active nodes after running the model
activeNodes = [v for v in G if labels[v]==1]
print("Number of nodes:",len(G),"number of active nodes:",len(activeNodes),
      "percentage of active nodes:",len(activeNodes)/len(G))
nx.draw_networkx(G.subgraph(activeNodes),pos,node_color="red")
plt.show()
```

```
Message: Node 1 is now active
Message: Node 2 is now active
Message: Node 14 is now active
Message: Node 4 is now active
Message: Node 13 is now active
Message: Node 7 is now active
Message: Node 5 is now active
Message: Node 6 is now active
Message: Node 8 is now active
Number of nodes: 14 number of active
    nodes: 10 percentage of active nodes
    0.7142857142857143
```



Υπάρχουν πολλά ερωτήματα που αφορούν αυτό το μοντέλο. Για παράδειγμα, πώς αλλάζει το ποσοστό των ενεργών κορυφών με βάση το κατώφλι ενεργοποίησης και τις αρχικές ενεργές κορυφές; Ποιες κορυφές αρκούν να ενεργοποιηθούν ώστε να πετύχουμε συγκεκριμένο ποσοστό ενεργοποίησης; Σε τι είδους γραφήματα καταλήγουμε σε ομοφωνία (όλοι να έχουν την ίδια γνώμη);

Τι άλλα μοντέλα επιρροής μπορούν να κατασκευασθούν; Για παράδειγμα, είναι γνωστό ότι δεν επηρεαζόμαστε από τις επαφές μας εξίσου, δηλαδή ορισμένες επαφές ασκούν μεγαλύτερη επιρροή. Στην περίπτωση αυτή, πρέπει να χρησιμοποιηθούν βάρη πάνω στις συνδέσεις που εκφράζουν τον βαθμό επιρροής.

33.2. ΑΝΙΧΝΕΥΣΗ ΚΟΙΝΟΤΗΤΩΝ. Έστω ένα γράφημα που αναπαριστά ένα κοινωνικό δίκτυο. Ένα υποσύνολο κορυφών του ονομάζεται **κοινότητα** (community) όταν τα μέλη του μοιράζονται κάτι κοινό (π.χ. την ίδια άποψη ή τα ίδια ενδιαφέροντα). Στην ενότητα αυτή παρουσιάζεται ένας απλός αλγόριθμος για την ανίχνευση κοινοτήτων σε ένα γράφημα (δεσμών ή τόξων).

Ο αλγόριθμος ονομάζεται **αλγόριθμος διάδοσης ετικετών** (label propagation algorithm) και βασίζεται στην παραδοχή ότι οι γειτονικές κορυφές συνήθως ανήκουν στην ίδια κοινότητα, διότι συνήθως τα άτομα προτιμούν να έχουν επαφές με κοινές ιδέες ή ενδιαφέροντα.

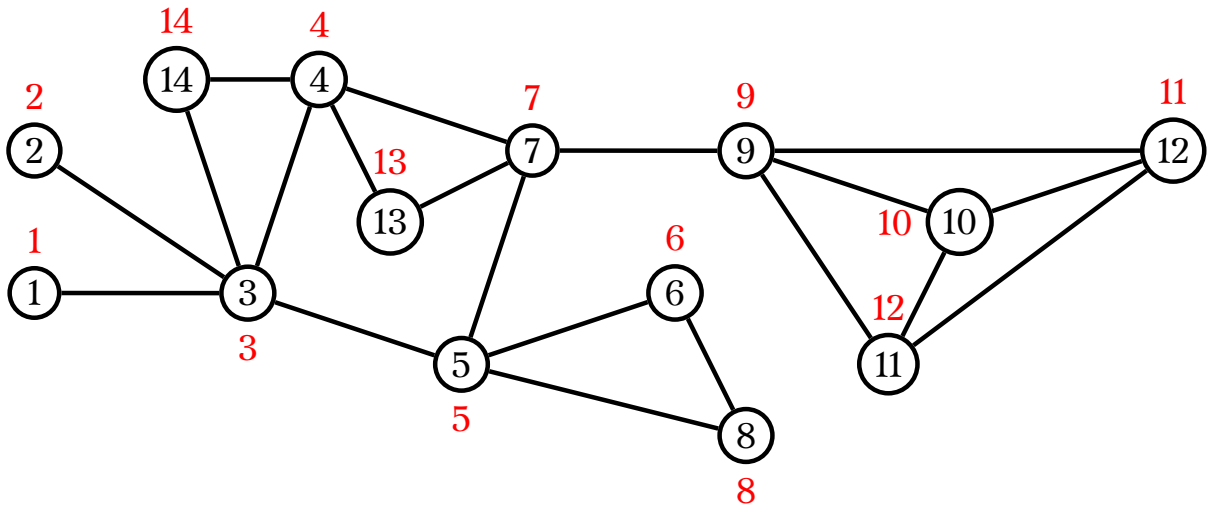
Σε κάθε βήμα του αλγορίθμου εξετάζεται κάθε κορυφή και εντάσσεται στην κοινότητα που ανήκει η πλειοψηφία των γειτόνων της. Η διαδικασία αυτή συγκλίνει σε μια σταθερή διαμέριση των κορυφών του γραφήματος, όπου κάθε κορυφή ανήκει στην ίδια κοινότητα που ανήκει η πλειοψηφία των γειτόνων της.

Συγκεκριμένα, έστω $G = (V, E)$ ένα γράφημα δεσμών.

- Αρχικά κάθε κορυφή v λαμβάνει μια διαφορετική ετικέτα $l(v)$.
- Επαναλαμβάνουμε τα επόμενα βήματα μέχρις ότου δεν υπάρχουν αλλαγές στις ετικέτες των κορυφών.
  - Για κάθε κορυφή v, σε τυχαία σειρά, η κορυφή v λαμβάνει την ετικέτα της πλειοψηφίας των γειτόνων της.
    Αν υπάρχουν πολλές ετικέτες που ισοψηφούν, τότε επιλέγεται τυχαία μια από αυτές.
    Αν η πλειοψηφία έχει την ίδια ετικέτα με την v δεν γίνεται καμιά αλλαγή.

Κατά την διάρκεια εκτέλεσης του αλγορίθμου οι ετικέτες των κοινοτήτων διαδίδονται μέσα στο δίκτυο: οι περισσότερες εξαφανίζονται ενώ άλλες κυριαρχούν. Επειδή η διαμέριση του δικτύου αλλάζει σε κάθε εκτέλεση της επανάληψης, απαιτούνται αρκετές επαναλήψεις μέχρι να συγκλίνει σε μια σταθερή κατάσταση. Παρόλα, αυτά συνήθως όμως ο αλγόριθμος συγκλίνει μετά από ένα μικρό αριθμό επαναλήψεων, που δεν εξαρτάται από το μέγεθος του γραφήματος.

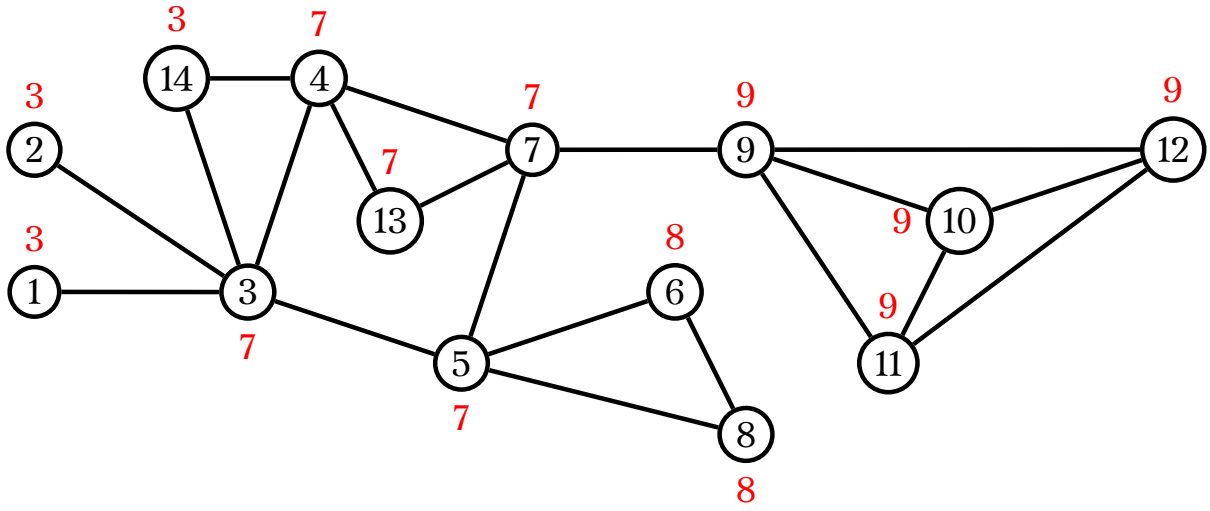
Παράδειγμα:



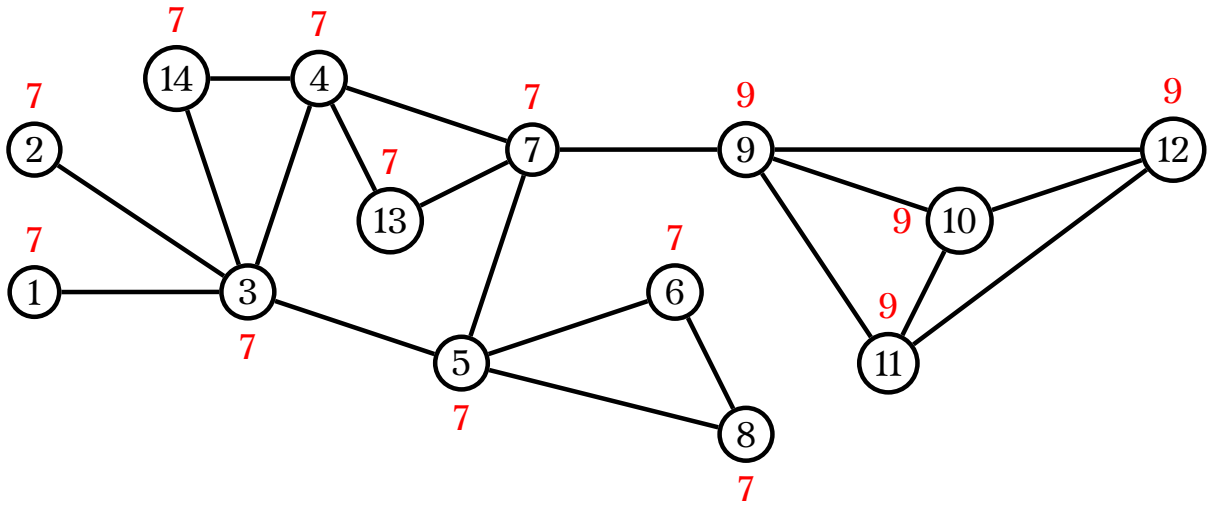
Αρχικά κάθε μια από τις 14 κορυφές του γραφήματος λαμβάνει ως ετικέτα ένα διαφορετικό τυχαίο αριθμό από το 1 εως το 14

Επιλέγουμε μια τυχαία σειρά στις κορυφές (διαλέγοντας μια τυχαία μετάθεση του [14]) πχ την μετάθεση 5 13 1 11 4 12 10 14 6 8 7 2 9 3.

Στην πρώτη επανάληψη, η κορυφή 5 λαμβάνει την ετικέτα 7, η κορυφή 13 λαμβάνει την ετικέτα 7, η κορυφή 1 λαμβάνει την ετικέτα 3, κορυφή 11 λαμβάνει την ετικέτα 9, η κορυφή 4 λαμβάνει την ετικέτα 7, η κορυφή 12 λαμβάνει την ετικέτα 9, η κορυφή 10 λαμβάνει την ετικέτα 9, η κορυφή 14 λαμβάνει την ετικέτα 3, η κορυφή 6 λαμβάνει την ετικέτα 8, η κορυφή 8 λαμβάνει την ετικέτα 8, η κορυφή 7 λαμβάνει την ετικέτα 7, η κορυφή 2 λαμβάνει την ετικέτα 3, κορυφή 9 λαμβάνει την ετικέτα 9, η κορυφή 3 λαμβάνει την ετικέτα 7.



Στην δεύτερη επανάληψη, οι κορυφές 1, 2, 14 λαμβάνουν την ετικέτα 7, οι κορυφές 6, 8 επίσης λαμβάνουν την ετικέτα 7, όλες οι υπόλοιπες κορυφές δεν αλλάζουν ετικέτα.



Στην τρίτη επανάληψη, δεν γίνεται καμία αλλαγή οπότε προκύπτει ότι στο γράφημα αυτό υπάρχουν 2 κοινότητες: Στην πρώτη ανήκουν οι κορυφές με ετικέτα 7 και στην δεύτερη ανήκουν οι κορυφές με ετικέτα 9.

Το πλεονέκτημα του αλγορίθμου διάδοσης ετικετών είναι ότι δεν απαιτεί εκ των προτέρων ούτε τον αριθμό των κοινοτήτων ούτε το μέγεθος κάθε μιας. Επίσης, δεν έχει κάποια παράμετρο ως είσοδο. Η υλοποίηση του είναι απλή και γρήγορη στην εκτέλεση ακόμα και σε γραφήματα με εκατομμύρια κορυφές. Τέλος, αν είναι

γνωστό εκ των προτέρων ότι κάποιοι κόμβοι ανήκουν σε συγκεκριμένες κοινότητες, τότε οι ετικέτες τους μπορούν να χρησιμοποιηθούν για να κατασκευασθεί μια αρχική διαμέριση.

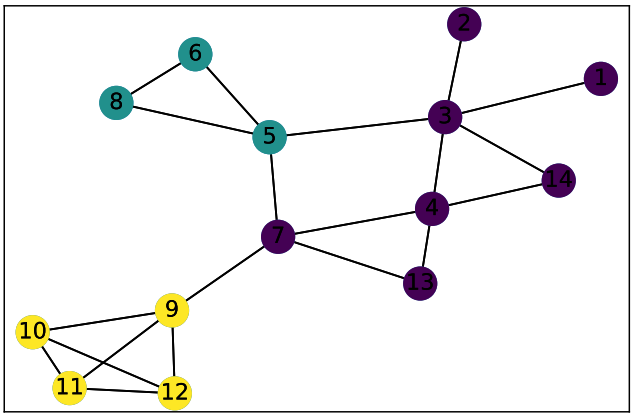
Η βιβλιοθήκη `networkx` διαθέτει μια υλοποίηση του αλγορίθμου διάδοσης ετικετών:

```
partition = nx.community.asyn_lpa_communities(G)
```

Ακολουθεί ένα παράδειγμα εκτέλεσης της μεθόδου, για το παραπάνω γράφημα, όπου ο αλγόριθμος ανιχνεύει 3 κοινότητες:

```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph()
n = 14 #A graph with 14 vertices
G.add_nodes_from(range(1,n+1))
E = [[1,3],[2,3],[3,4],[3,5],[3,14],[4,7],[4,13],[4,14],[5,7],[5,6],[5,8],
     [6,8],[7,9],[7,13],[9,10],[9,11],[9,12],[10,11],[10,12],[11,12]]
G.add_edges_from(E)

#get the communities using label propagation algorithm
partition = nx.community.asyn_lpa_communities(G)

#assign colors to nodes based on the communities
colors = [0 for i in range(1,n+2)]
color_id = 1
for community in partition:
    print("Comunity",color_id,"is:",community)
    for v in community: colors[v] = color_id
    color_id += 1 #assign different color to next community
colors.pop(0)

pos = nx.layout.kamada_kawai_layout(G)
nx.draw_networkx(G,pos,node_color=colors)
```

```
Comunity 1 is: {1, 2, 3, 4, 7, 13, 14}
Comunity 2 is: {8, 5, 6}
Comunity 3 is: {9, 10, 11, 12}
```



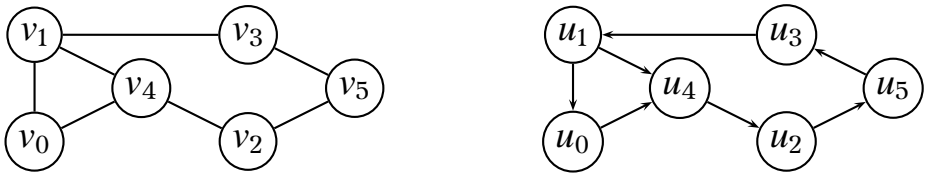
34. ΣΥΝΤΟΜΟΤΕΡΕΣ ΔΙΑΔΡΟΜΕΣ

Πολλά προβλήματα βελτιστοποίησης ανάγονται στην εύρεση της συντομότερης διαδρομής ανάμεσα σε κάποιες κορυφές ενός γραφήματος.

Προφανώς, η συντομότερη διαδρομή που συνδέει δύο κορυφές δεν περιέχει κύκλους επομένως της όλες οι κορυφές είναι διαφορετικές, δηλαδή είναι ένα μονοπάτι του γραφήματος.

Υπενθύμιση: Μονοπάτι με αρχή την κορυφή v και πέρας την κορυφή u σε ένα γράφημα ονομάζεται μια ακολουθία κορυφών $v_1, v_2, \ldots, v_k$, για την οποία ισχύουν ότι $v_1 = v$, $v_k = u$, για κάθε ζεύγος διαδοχικών κορυφών υπάρχει δεσμός μεταξύ τους (ή τόξο από την πρώτη προς την δεύτερη) και κάθε κορυφή εμφανίζεται ακριβώς μια φορά. Το μονοπάτι αυτό ονομάζεται (v, u)-μονοπάτι. Ο αριθμός των δεσμών (ή τόξων) που συνδέουν τις κορυφές του μονοπατιού ονομάζεται μήκος του μονοπατιού. Ένα μονοπάτι θεωρείται ελάχιστο μονοπάτι όταν έχει το ελάχιστο δυνατό μήκος.

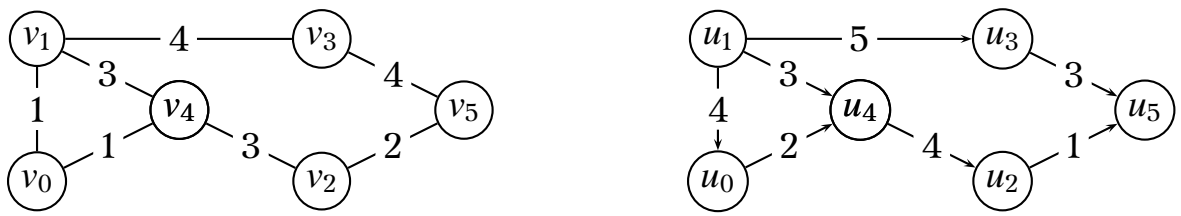
Παράδειγμα Στα επόμενα δύο γραφήματα



το ελάχιστο μονοπάτι μεταξύ των κορυφών v_1 και v_5 είναι το μονοπάτι v_1, v_3, v_5 με μήκος 2. Το ελάχιστο μονοπάτι με αρχή την κορυφή u_1 και πέρας την κορυφή u_5 είναι το μονοπάτι u_1, u_4, u_2, u_5 με μήκος 3, ενώ το ελάχιστο μονοπάτι με αρχή την κορυφή u_5 και πέρας την κορυφή u_1 είναι το μονοπάτι u_5, u_3, u_1 με μήκος απόσταση των κορυφών u_5 και u_1 είναι 2.

Στις εφαρμογές, η πιο συνηθισμένη περίπτωση είναι ότι σε κάθε δεσμός (ή τόξο) αντιστοιχεί ένα βάρος, μπορεί να σχετίζεται με κάποιο χρόνο, ή κόστος, ή απόσταση κ.ο.κ. Στην περίπτωση αυτή ελάχιστο θεωρείται το μονοπάτι στο οποίο άθροισμα των βαρών των τόξων του είναι ελάχιστο και όχι το μονοπάτι με το ελάχιστο πλήθος κορυφών.

Παράδειγμα Στα επόμενα δύο γραφήματα



το ελάχιστο μονοπάτι μεταξύ των κορυφών v_1 και v_5 είναι το μονοπάτι v_1, v_0, v_4, v_2 με συνολικό βάρος 7. Τα ελάχιστα μονοπάτια με αρχή την κορυφή u_1 και πέρας την κορυφή u_5 είναι το μονοπάτι u_1, u_4, u_2, u_5 και το μονοπάτι u_1, u_3, u_5 με συνολικό βάρος 8.

Θα συμβολίζουμε με $w(i, j)$ το βάρος του τόξου (ή δεσμού) από την κορυφή v_i στην κορυφή v_j.

Αν όλα τα βάρη είναι θετικά, τότε και σ' αυτές τις περιπτώσεις το κλειδί για την εύρεση του ελάχιστου (v, u)-μονοπατιού σε ένα γράφημα είναι η παρατήρηση ότι για κάθε κορυφή v_i που περιέχεται σ' αυτό το μονοπάτι πρέπει επίσης να ισχύει ότι το (v, v_i)-μονοπάτι είναι και αυτό ελάχιστο. Σε αντίθετη περίπτωση, θα υπήρχε ένα μονοπάτι με μικρότερο συνολικό βάρος και άρα δεν θα ήταν ελάχιστο.

Η παρατήρηση αυτή είναι η βάση για τον επόμενο αλγόριθμο εύρεσης των ελάχιστων μονοπατιών σε γραφήματα με βάρη, ο οποίος ονομάζεται αλγόριθμος του Dijkstra, ο οποίος είναι μια προσαρμογή της αναζήτησης κατά πλάτος ώστε να λαμβάνονται υπόψη τα βάρη των τόξων (ή δεσμών).

Για την υλοποίηση του αλγόριθμου του Dijkstra θα χρησιμοποιηθεί μια ουρά προτεραιότητας Q και δύο βοηθητικοί πίνακες A και B.

Οι πίνακες A, B έχουν μέγεθος ίσο με το πλήθος των κορυφών του γραφήματος.

Στον πίνακα A αποθηκεύουμε τα βάρη των ελάχιστων μονοπατιών με αρχή την κορυφή v_j, ενώ στον πίνακα B αποθηκεύουμε για κάθε κορυφή τον δείκτη της προηγούμενης κορυφής στο ελάχιστου μονοπάτι που ανήκει.

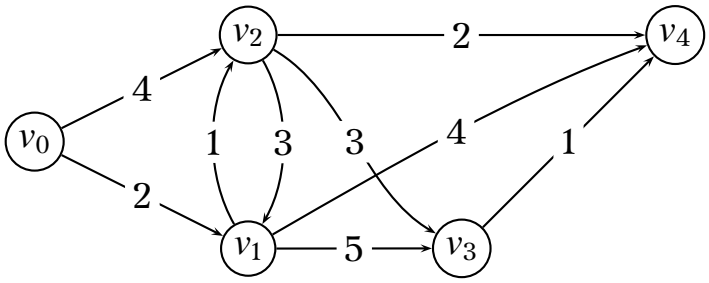
Υπενθύμιση: Η βασική ιδέα της ουράς προτεραιότητας είναι ότι τα στοιχεία της έχουν ορισμένα βάρη ως κλειδιά, και κάθε φορά προτεραιότητα έχει το στοιχείο με το ελάχιστο βάρος.

Τα βήματα του αλγορίθμου του Dijkstra, για την εύρεση των συντομότερων μονοπατιών που ξεκινουν από την κορυφή v_j σε γραφήματα με (θετικά) βάρη, είναι τα ακόλουθα:

(1) Αρχικά, θέτουμε σε όλες τις θέσεις του πίνακα A την τιμή ∞[8] και σε όλες τις θέσεις του πίνακα B την τιμή -1.
(2) Θέτουμε $A[j] = 0$.
(3) Τοποθετούμε όλες τις κορυφές του γραφήματος στην ουρά προτεραιότητας Q με κλειδιά τις τιμές του πίνακα A.
(4) Όσο η ουρά προτεραιότητας Q είναι μη κενή εκτελούμε τα ακόλουθα βήματα:
   (α΄) Αφαιρούμε από την Q την κορυφή με το μικρότερο κλειδί, έστω την v_k.
   (β΄) Για κάθε κορυφή v_i στην οποία καταλήγουν τόξα με αρχή την v_k εκτελούμε τα εξής βήματα:
      (i) Αν $A[k] + w(k, i) < A[i]$ τότε
         θέτουμε $A[i] = A[k] + w(k, i)$,
         θέτουμε $B[i] = k$
         ενημερώνουμε το κλειδί της v_i στην ουρά.

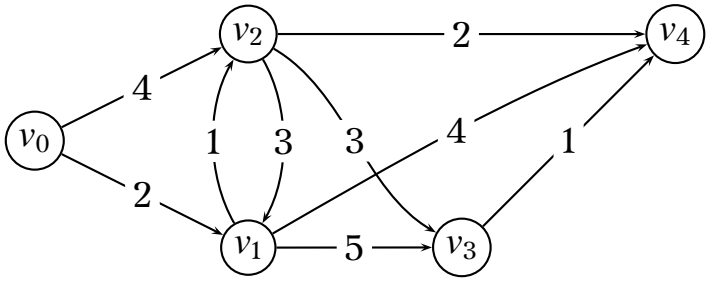
Παρατήρηση Η ουρά προτεραιότητας υλοποιείται συνήθως με τη βοήθεια ενός σωρού αλλά μπορεί να υλοποιηθεί και με τη βοήθεια ενός πίνακα.

Παράδειγμα Για το επόμενο κατευθυνόμενο γράφημα με τόξα



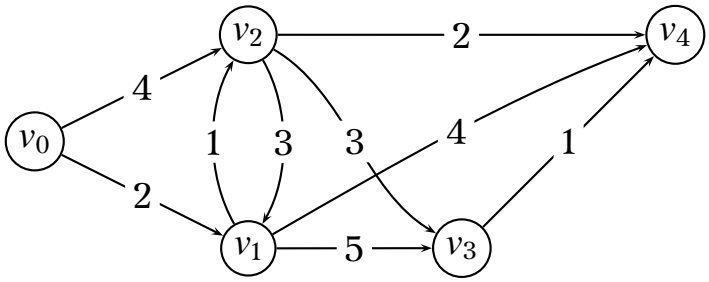
η εκτέλεση του αλγορίθμου του Dijkstra με αρχή την κορυφή v_0 είναι η εξής:

(1) Αρχικά θέτουμε σε όλες τις θέσεις του πίνακα A την τιμή ∞ και σε όλες τις θέσεις που πίνακα B την τιμή -1, ενώ η ουρά Q είναι άδεια. Στη συνέχεια, θέτουμε $A[0] = 0$.



	0	1	2	3	4	
A = [	0,	∞,	∞,	∞,	∞,	]
B = [	-1,	-1,	-1,	-1	-1	]
Q = [						]

(2) Στη συνέχεια τοποθετούμε όλες τις κορυφές του γραφήματος στην ουρά προτεραιότητας Q με κλειδί τις αντίστοιχες τιμές του πίνακα A.

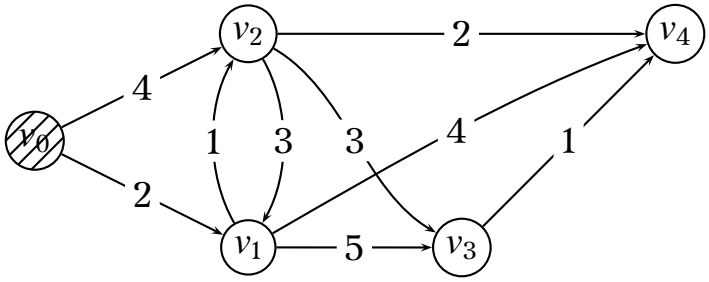


	0	1	2	3	4	
A = [	0,	∞,	∞,	∞,	∞,	]
B = [	-1,	-1,	-1,	-1	-1	]
Q = [	v_0,	v_1,	v_2,	v_3,	v_4	]

(3) Επειδή η ουρά προτεραιότητας Q είναι μη κενή, αφαιρούμε από την Q την κορυφή με το μικρότερο κλειδί, την v_0. Από την v_0 καταλήγουν τόξα στις κορυφές v_1 και v_2.

Επειδή $A[0] + w(0,1) = 0 + 2 = 2 < \infty = A[1]$ θέτουμε $A[1] = 2$, $B[1] = 0$ και ενημερώνουμε το κλειδί της κορυφής v_1 στην ουρά Q.

Επειδή $A[0] + w(0,2) = 0 + 4 = 4 < \infty = A[2]$ θέτουμε $A[2] = 4$, $B[2] = 0$ και ενημερώνουμε το κλειδί της v_2 στην ουρά Q.



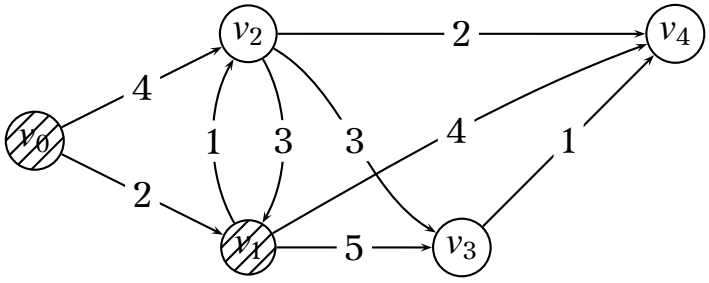
$$
\begin{array}{rccccc}
 & 0 & 1 & 2 & 3 & 4 \\
A = [& 0, & 2, & 4, & \infty, & \infty, \ ] \\
B = [& -1, & 0, & 0, & -1 & -1 \ ] \\
Q = [& & v_1, & v_2, & v_3, & v_4 \ ]
\end{array}
$$

(4) Επειδή η ουρά προτεραιότητας Q είναι μη κενή, αφαιρούμε από την Q την κορυφή με το μικρότερο κλειδί, την v_1. Από την v_1 καταλήγουν τόξα στις κορυφές v_2, v_3 και v_4.

Επειδή $A[1] + w(1,2) = 2 + 1 = 3 < 4 = A[2]$ θέτουμε $A[2] = 3$, $B[1] = 1$ και ενημερώνουμε το κλειδί της κορυφής v_2 στην ουρά Q.

Επειδή $A[1] + w(1,3) = 2 + 5 = 7 < \infty = A[3]$ θέτουμε $A[3] = 7$, $B[3] = 1$ και ενημερώνουμε το κλειδί της v_3 στην ουρά Q.

Επειδή $A[1] + w(1,4) = 2 + 4 = 6 < \infty = A[4]$ θέτουμε $A[4] = 6$, $B[4] = 1$ και ενημερώνουμε το κλειδί της v_4 στην ουρά Q.



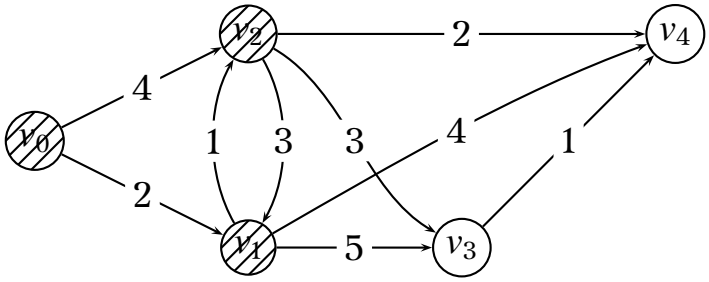
$$
\begin{array}{rccccc}
 & 0 & 1 & 2 & 3 & 4 \\
A = [& 0, & 2, & 3, & 7, & 6, \ ] \\
B = [& -1, & 0, & 1, & 1 & 1 \ ] \\
Q = [& & & v_2, & v_3, & v_4 \ ]
\end{array}
$$

(5) Επειδή η ουρά προτεραιότητας Q είναι μη κενή, αφαιρούμε από την Q την κορυφή με το μικρότερο κλειδί, την v_2. Από την v_2 καταλήγουν τόξα στις κορυφές v_1, v_3 και v_4.

Επειδή $A[2]+w(2,1) = 3+3 = 6 > 2 = A[1]$ δεν εκτελούμε καμιά ενέργεια.[9]

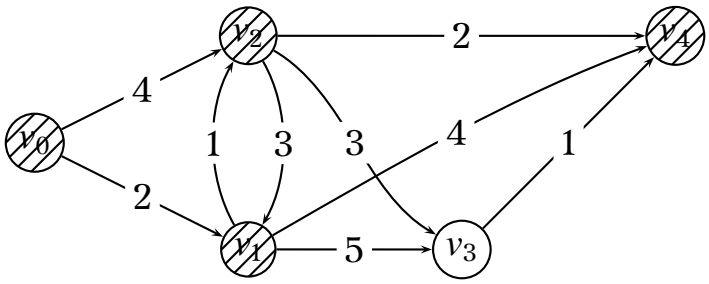
Επειδή $A[2] + w(2,3) = 3 + 3 = 6 < 7 = A[3]$ θέτουμε $A[3] = 6$, $B[3] = 2$ και ενημερώνουμε το κλειδί της v_3 στην ουρά Q.

Επειδή $A[2] + w(2,4) = 3 + 2 = 5 < 6 = A[4]$ θέτουμε $A[4] = 5$, $B[4] = 2$ και ενημερώνουμε το κλειδί της v_4 στην ουρά Q.



	0	1	2	3	4	
A = [	0,	2,	3,	6,	5,	]
B = [	−1,	0,	1,	2	2	]
Q = [				v_3,	v_4	]

(6) Επειδή η ουρά προτεραιότητας Q είναι μη κενή, αφαιρούμε από την Q την κορυφή με το μικρότερο κλειδί, την v_4. Δεν υπάρχουν τόξα με αρχή την v_4 επομένως δεν εκτελούμε καμιά ενέργεια.

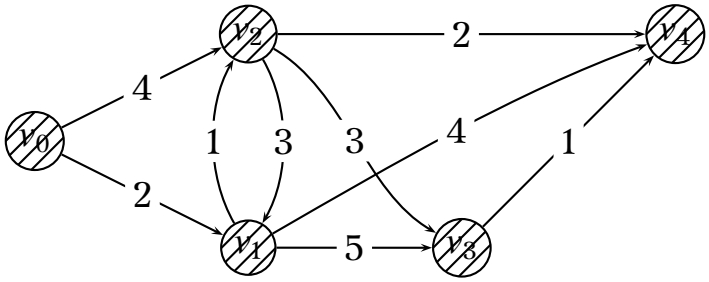


	0	1	2	3	4	
A = [	0,	2,	3,	6,	5,	]
B = [	−1,	0,	1,	2	2	]
Q = [				v_3		]

[9] Αυτό το βήμα μπορεί να παραλειφθεί διότι επειδή η v_1 δεν περιέχεται στην ουρά το κόστος της είναι ήδη μικροτερο από όλες τις κορυφές που περιέχονται στην ουρά.

(7) Επειδή η ουρά προτεραιότητας Q είναι μη κενή, αφαιρούμε από την Q την κορυφή με το μικρότερο κλειδί, την v_3. Από την v_3 καταλήγει τόξο στην κορυφή v_4.

Επειδή $A[3] + w(3,4) = 6 + 1 = 7 > 6 = A[5]$ δεν εκτελούμε καμιά ενέργεια.



	0	1	2	3	4	
A = [	0,	2,	3,	6,	5,	]
B = [	−1,	0,	1,	2	2	]
Q = [						]

(8) Επειδή η ουρά προτεραιότητας Q είναι κενή, η εύρεση των ελάχιστων μονοπατιών ολοκληρώθηκε.

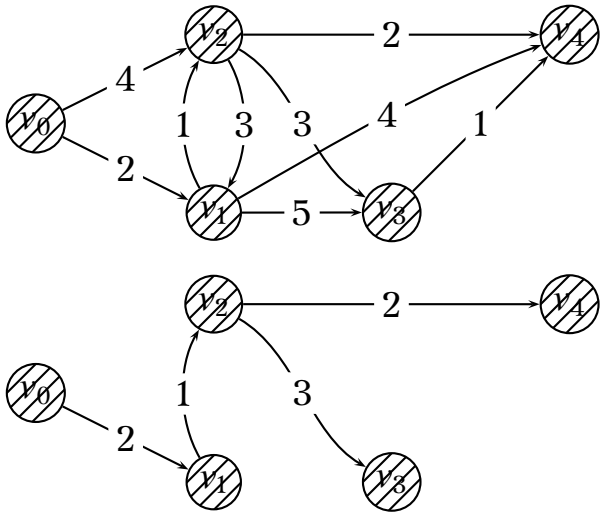
Τα μήκη των αντίστοιχων ελάχιστων μονοπατιών βρίσκονται στον πίνακα A.

Για την εύρεση των αντίστοιχων ελάχιστων μονοπατιών ξεκινάμε από την τελευταία κορυφή και χρησιμοποιώντας τον πίνακα B κινούμαστε ανάποδα μέχρι να φτάσουμε στην κορυφή v_0.

- Για την εύρεση του ελάχιστου μονοπατιού με τέλος την κορυφή v_1 έχουμε ότι:
  Η τελευταία κορυφή είναι η v_1.
  Επειδή $B[1] = 0$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_0.
  Άρα, το μονοπάτι είναι το v_0, v_1 με βάρος $A[1] = 2$.
- Για την εύρεση του ελάχιστου μονοπατιού με τέλος την κορυφή v_2 έχουμε ότι:
  Η τελευταία κορυφή είναι η v_2.
  Επειδή $B[2] = 1$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_1.
  Επειδή $B[1] = 0$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_0.
  Άρα, το μονοπάτι είναι το v_0, v_1, v_2 με βάρος $A[2] = 3$.
- Για την εύρεση του ελάχιστου μονοπατιού με τέλος την κορυφή v_3 έχουμε ότι:
  Η τελευταία κορυφή είναι η v_3.
  Επειδή $B[3] = 2$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_2.
  Επειδή $B[2] = 1$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_1.
  Επειδή $B[1] = 0$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_0.
  Άρα, το μονοπάτι είναι το v_0, v_1, v_2, v_3 με βάρος $A[3] = 6$.
- Για την εύρεση του ελάχιστου μονοπατιού με τέλος την κορυφή v_4 έχουμε ότι:
  Η τελευταία κορυφή είναι η v_4.
  Επειδή $B[4] = 2$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_2.
  Επειδή $B[2] = 1$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_1.
  Επειδή $B[1] = 0$, η προηγούμενη κορυφή στο ελάχιστο μονοπάτι είναι η v_0.
  Άρα, το μονοπάτι είναι το v_0, v_1, v_2, v_4 με βάρος $A[4] = 5$.

Σε κάθε εκτέλεση του αλγορίθμου του Dijkstra αντιστοιχεί ένα δένδρο με κορυφές τις κορυφές του γραφήματος και τόξα (ή δεσμούς) τα τόξα (ή τους δεσμούς) που συμμετέχουν στα ελάχιστα μονοπάτια, το οποίο ονομάζεται δένδρο ελάχιστων μονοπατιών.

Για παράδειγμα, στην εκτέλεση του αλγορίθμου του Dijkstra αντιστοιχεί το επόμενο δένδρο ελάχιστων μονοπατιών.



$$
\begin{array}{rc}
 & \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\
A = [& \begin{array}{ccccc} 0, & 2, & 3, & 6, & 5, \end{array}] \\
B = [& \begin{array}{ccccc} -1, & 0, & 1, & 2 & 2 \end{array}]
\end{array}
$$

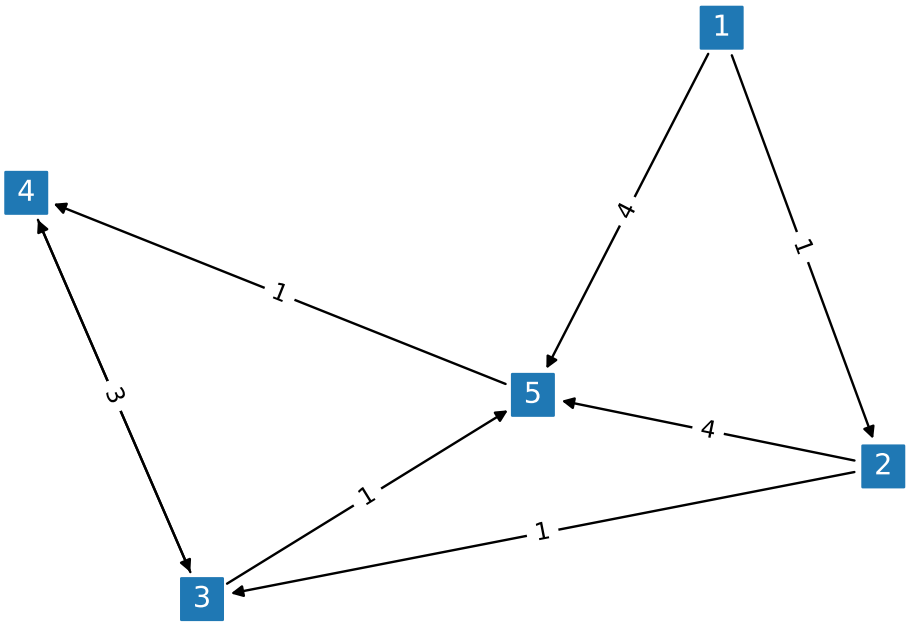
Η κατασκευή του αντίστοιχου δένδρου προκύπτει από τον πίνακα B θέτοντας κάθε κορυφή v_j, με πεπερασμένη θετική τιμή $B[j] = i$ ως παιδί της κορυφής v_j.

Η βιβλιοθήκη `networkx` διαθέτει μια υλοποίηση του αλγορίθμου του Dijkstra:

```
import networkx as nx
import matplotlib.pyplot as plt
#https://networkx.org/documentation/stable/reference/algorithms/shortest_paths.html

G = nx.DiGraph()
V = [1,2,3,4,5]
E = [(1,2, 1), (2,3, 1), (3,5, 1), (5,4, 1), (3,4, 2), (4,3, 3), (1,5, 4), (2,5, 4)]
G.add_weighted_edges_from(E)
weight_labels = nx.get_edge_attributes(G,'weight')
#print(weight_labels)
s = 1 #source
pd, dd = nx.single_source_dijkstra(G, s, weight = 'weight') #resturns dict of
    distances, dict of paths

#print(pd)
#print(dd)

print("A shortest path between", s)
for u in dd.keys():
    print("and",u,"is:",dd[u], ", weight =", pd[u])

pos = nx.spring_layout(G)
nx.draw(G,pos,font_color = 'white', node_shape = 's', with_labels = True)
nx.draw_networkx_edge_labels(G,pos,edge_labels=weight_labels)
plt.show()
```

Output:

```
  A shortest path between 1
and 1 is: [1] , weight = 0
and 2 is: [1, 2] , weight = 1
and 5 is: [1, 2, 3, 5] , weight = 3
and 3 is: [1, 2, 3] , weight = 2
and 4 is: [1, 2, 3, 4] , weight = 4
```



34.1. Ο ΑΛΓΟΡΙΘΜΟΣ A^*. Ο αλγόριθμος A^* (alpha star) αναζητά το συντομότερο μονοπάτι σε ένα γράφημα $G = (V, U)$ με βάρη από μια αρχική κορυφή s (source) σε μια τελική κορυφή t (target), με τη βοήθεια μιας ευρετικής (heuristic) συνάρτησης $h : V \to [0, +\infty]$, η οποία δίνει για κάθε $v \in V$ μια εκτίμηση για το συντομότερο μονοπάτι από τη v στην t. Με τον τρόπο αυτό αποκλείονται κάποιες κορυφές του γραφήματος από την διαδικασία αναζήτησης (αφού εκτιμούμε ότι το βάρος του s–t μονοπατιού που διέρχεται από αυτές δεν μπορεί να είναι ελάχιστο) και μειώνεται το κόστος υπολογισμού της συντομότερης $s - t$ διαδρομής.

Ο ορισμός της συνάρτησης αυτής εξαρτάται από το ίδιο το πρόβλημα. Για παράδειγμα, αν οι κορυφές αντιπροσωπεύουν πόλεις, τότε η $h(v)$ μπορεί να ορισθεί ως η Ευκλείδεια απόσταση από την v στην t.

Με τον τρόπο αυτό, κατά τον υπολογισμό της συντομότερης διαδρομής μεταξύ Αθήνας και Πάτρας θα αρχίσουν να αποκλείονται επιλογές που οδηγούν πολύ ανατολικότερα σε σχέση με την τρέχουσα τοποθεσία διότι σίγουρα δεν θα μπορούν να επεκταθούν σε συντομότερες διαδρομές (από την στιγμή που υπάρχουν άλλες καλύτερες υποψήφιες επιλογές).

Η h ονομάζεται **αποδεκτή** (admissible) αν δεν παίρνει ποτέ μεγαλύτερη τιμή από (δεν υπερεκτιμά) την πραγματική απόσταση (άρα $h(t) = 0$). Στην περίπτωση αυτή, ο αλγόριθμος εγγυάται ότι θα βρει τη βέλτιστη λύση, εφόσον αυτή υπάρχει. Για παράδειγμα, η Ευκλείδεια απόσταση είναι μια αποδεκτή ευρετική συνάρτηση.

Ο αλγόριθμος A^* χρησιμοποιεί μια ουρά προτεραιότητας Q (που αρχικά περιέχει μόνο την s) και τις συναρτήσεις f, g, h, όπου

$g(v)$ είναι η απόσταση του συντομότερου μονοπατιού από την s στη v που είναι γνωστό μέχρι στιγμής,

$h(v)$ είναι η εκτίμηση της απόστασης του συντομότερου μονοπατιού από την v στην t, και

$f(v) = g(v) + h(v)$ είναι η (εκτίμηση) της (συνολικής) απόστασης του συντομότερου s–t μονοπατιού που διέρχεται από την v. (Παρατηρείστε ότι $g(s) = 0$ και $f(s) = h(s)$).

Σε κάθε επανάληψη αφαιρεί από την Q την κορυφή v με την ελάχιστη τιμή $f(v)$, ενημερώνει την τιμή g των γειτόνων της και τους προσθέτει στην Q (αν δεν είναι ήδη μέσα).

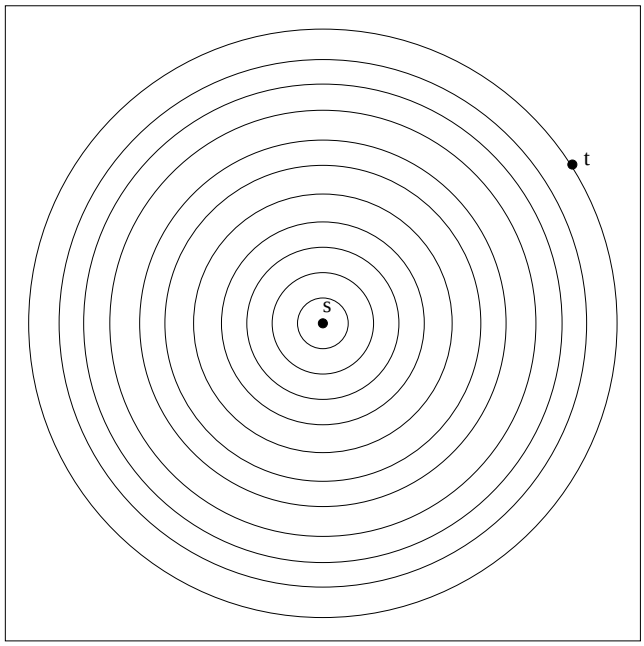
Αν η h είναι **συνεπής** (consistent), δηλαδή ικανοποιεί την ανισότητα $h(v) \leq w(v, u) + h(u)$, για κάθε $(v, u) \in U$, (π.χ. η Ευκλείδεια απόσταση) τότε αποδεικνύεται ότι καμία κορυφή δεν θα μπει δεύτερη φορά στην ουρά. Αυτό σημαίνει ότι όταν μια κορυφή v αφαιρείται από την ουρά, η τιμή $g(v)$ ισούται με την πραγματική ελάχιστη απόσταση από την s στην v.

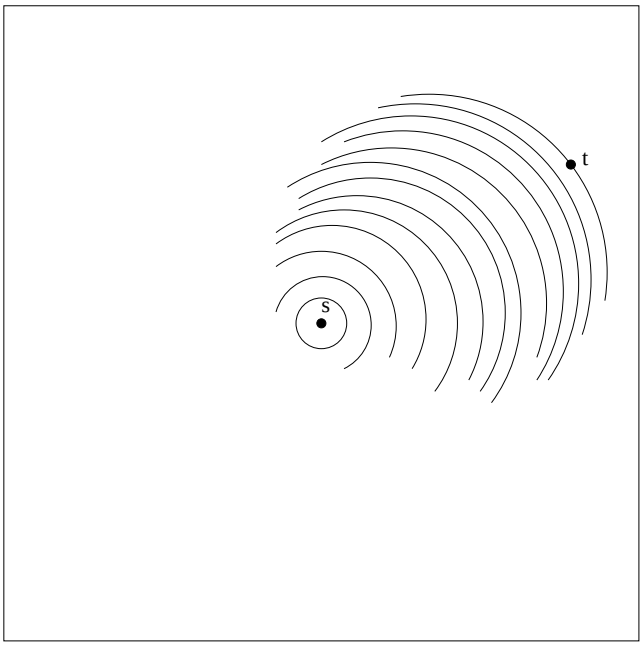
Ο αλγόριθμος τερματίζει είτε όταν αφαιρείται η t από την Q, οπότε η ελάχιστη απόσταση από την s στην t ισούται με $f(t) = g(t)$, είτε όταν αδειάζει η ουρά, πράγμα που σημαίνει ότι η t δεν μπορεί να προσεγγιστεί από την s.

Είσοδος: Ένα γράφημα $G = (V, U)$, μια κορυφή s μια κορυφή t, η ευρετική συνάρτηση h και ο πίνακας βαρών w του G.
Αποτέλεσμα: Η τιμή $g(t)$ ισούται τελικά με το βάρος της συντομότερης διαδρομής από την s στην t.

```
για κάθε κορυφή v ∈ V  g(v) ← f(v) ← ∞;
f(s) ← h(s);
g(s) ← 0;
Q ← {s};
ενόσω Q ≠ ∅
    αφαίρεσε από την Q την κορυφή v, με την ελάχιστη τιμή f(v);
    αν v = t τότε return επιτυχία;
    για κάθε γείτονα z της v
        g' = g(v) + w(v, z);
        αν g' < g(z) τότε
            prev[z] ← v;
            g(z) = g';
            f(z) = g(z) + h(z);
        τέλος
        αν z ∉ Q τότε πρόσθεσε την z στην Q;
    τέλος
    return αποτυχία;
τέλος ενόσω
```

Η διαφορά του A^* από τον αλγόριθμο του Dijkstra μπορεί να απεικονισθεί σχηματικά με τα παρακάτω διαγράμματα:



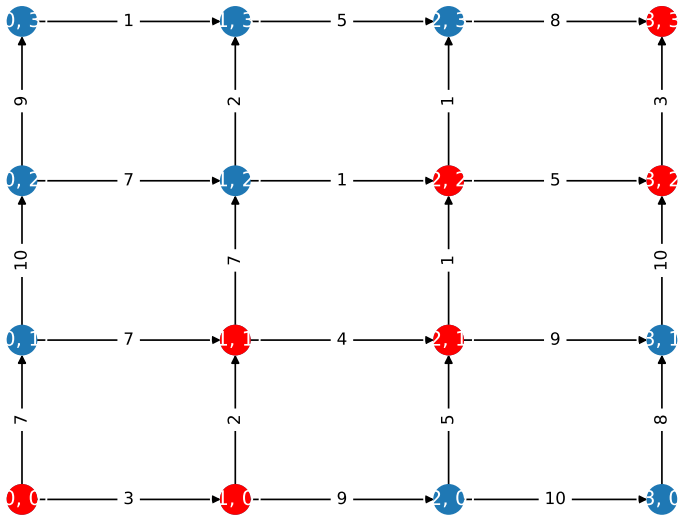


Με τον αλγόριθμο A^* (με καλή ευρετική συνάρτηση) σαρώνεται ένα πολύ μικρότερο μέρος του γραφήματος κατά την διαδικασία εύρεσης της συντομότερης $s - t$ διαδρομής, σε σχέση με τον αλγόριθμο του Dijkstra.

Ο επόμενος κώδικας κατασκευάζει ένα πλέγμα $N \times N$ και υπολογίζει ένα συντομότερο μονοπάτι από την $s = (0,0)$ στην $t = (N-1, N-1)$ με τον αλγόριθμο A^*. Ως ευρετική συνάρτηση h χρησιμοποιείται η απόσταση Manhattan.

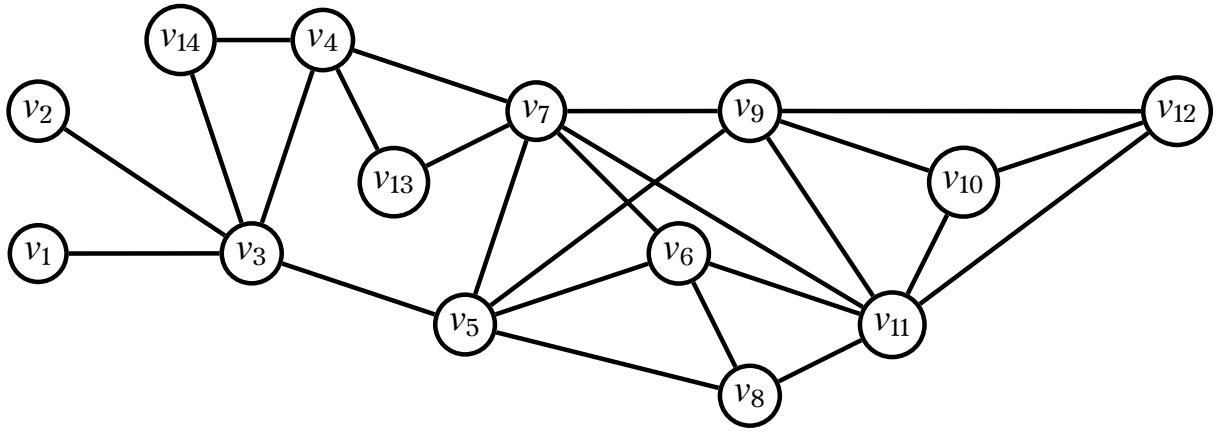
```
import networkx as nx
import matplotlib.pyplot as plt
import random as rd

G = nx.DiGraph()
N = 4 #create a NXN grid
V = []
for i in range(N):
    for j in range(N):
        V.append((i,j))
G.add_nodes_from(V)
for i in range(N-1):
    for j in range(N-1):
        G.add_edge((i,j),(i+1,j), weight = rd.randint(1,10))
        G.add_edge((i,j),(i,j+1), weight = rd.randint(1,10))
for i in range(N-1):
    G.add_edge((i,N-1),(i+1,N-1), weight = rd.randint(1,10))
    G.add_edge((N-1,i),(N-1,i+1), weight = rd.randint(1,10))

s, t = (0,0), (N-1, N-1) #source, target nodes

def h(v,u): return abs(u[0]-v[0]) + abs(u[1]-v[1])

P = nx.astar_path(G, s, t, heuristic = h, weight='weight')
GP = nx.Graph()
GP.add_nodes_from(P)


pos = {v:v for v in V}
els = nx.get_edge_attributes(G,'weight')
nx.draw(G,pos,font_color = 'white', with_labels = True)
nx.draw_networkx_edge_labels(G,pos,edge_labels=els)
nx.draw(GP,pos, node_color = "red")
plt.show()
```

Output:



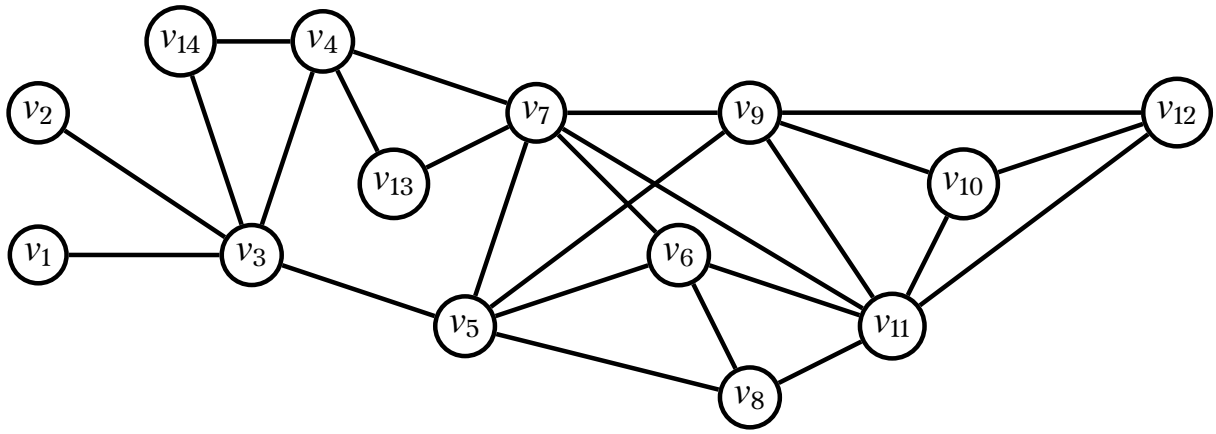
Ασκήσεις προς επίλυση

(1) Να εξετασθεί τι θα συμβεί στο παρακάτω δίκτυο επιρροής

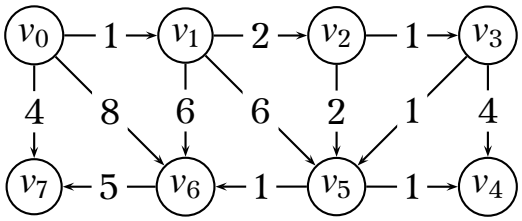


αν χρησιμοποιήσουμε τον αλγόριθμο κατωφλίου θέτοντας $\theta = 50\%$ και ενεργοποιώντας αρχικά τις κορυφές v_5 και v_7.

(2) Χρησιμοποιώντας τον αλγόριθμο διάδοσης ετικετών να ανιχνευτούν οι κοινότητες του επόμενου γραφήματος

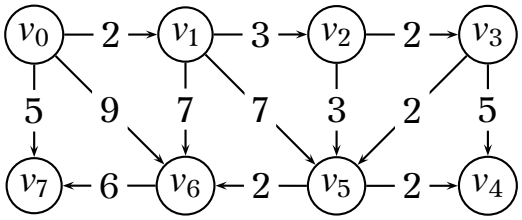


(3) Να βρεθούν τα ελάχιστα μονοπάτια του επόμενου κατευθυνόμενου γραφήματος με αρχή την κορυφή v_0



και στη συνέχεια να κατασκευασθεί το αντίστοιχο δένδρο ελάχιστων μονοπατιών.

(4) Να βρεθούν τα ελάχιστα μονοπάτια του επόμενου κατευθυνόμενου γραφήματος με αρχή την κορυφή v_0, το οποίο προκύπτει από το προηγούμενο με αύξηση όλων των βαρών του κατά 1

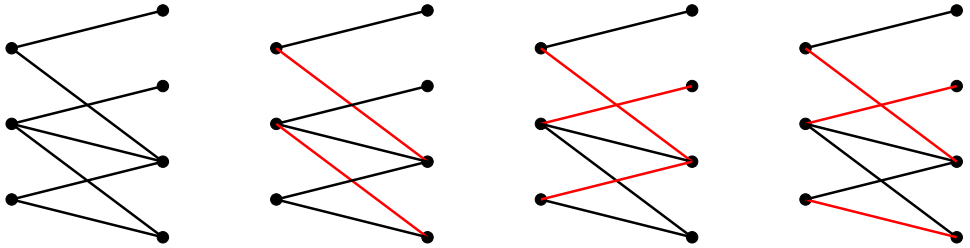


και στη συνέχεια να κατασκευασθεί το αντίστοιχο δένδρο ελάχιστων μονοπατιών. Τί παρατηρείτε; Είναι τα ελάχιστα μονοπάτια ίδια με πριν; Γιατί;

35. ΠΑΡΑΡΤΗΜΑ: ΤΑΙΡΙΑΣΜΑΤΑ

Ένα σύνολο δεσμών $M \subseteq E$ ενός γραφήματος $G = (V, E)$ ονομάζεται **ταίριασμα** (matching) του G ανν δεν υπάρχουν δεσμοί στο M με κοινή κορυφή, ή ισοδύναμα κάθε κορυφή του G περιέχεται το πολύ σε ένα δεσμό του M.

Οι κόκκινοι δεσμοί των σχημάτων (β) και (δ) αποτελούν ταιριάσματα του διμερούς γραφήματος (α), ενώ οι κόκκινοι δεσμοί του σχήματος (γ) δεν είναι ταίριασμα διότι υπάρχουν δεσμοί που έχουν κοινό άκρο.



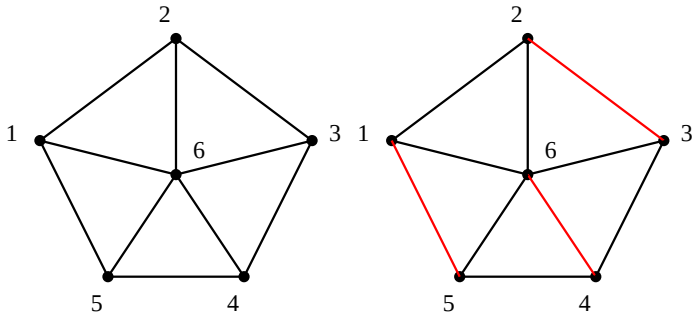
(α) (β) (γ) (δ)

Οι κορυφές του G που περιέχονται σε δεσμούς του ταιριάσματος M ονομάζονται **ταιριασμένες** (matched) κορυφές. Αν ο δεσμός $\{u, v\}$ ανήκει στο ταίριασμα M τότε λέμε ότι η κορυφή u **ταιριάζει** με την κορυφή v στο M. Μια κορυφή που δεν ανήκει σε δεσμό του M ονομάζεται **ακόρεστη** (unsaturated) ή **ελεύθερη** (free) (ως προς το M). Το πλήθος των δεσμών ενός ταιριάσματος M ονομάζεται **μέγεθος** του M και σημειώνεται με $|M|$

Ένα ταίριασμα M ονομάζεται **μέγιστο** (maximum) ανν έχει τον μεγαλύτερο δυνατό αριθμό δεσμών. Στο παράδειγμα, το ταίριασμα (δ) είναι μέγιστο, ενώ το ταίριασμα (β) δεν είναι μέγιστο.

Ένα ταίριασμα M ονομάζεται **τέλειο** (perfect) ανν κάθε κορυφή του γραφήματος G εμφανίζεται σε ακριβώς ένα δεσμό του M, ή ισοδύναμα ανν οι δεσμοί του M αποτελούν μια διαμέριση των κορυφών του G σε ζεύγη.[10]

Κανένα από τα ταιριάσματα (β), (δ) δεν είναι τέλειο. Στο σχήμα (στ) οι κόκκινοι δεσμοί αποτελούν ένα τέλειο ταίριασμα για το γράφημα (ε)

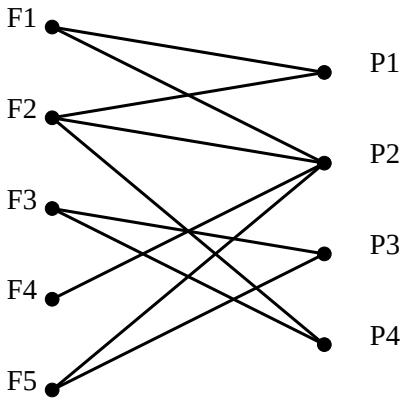


(ε) (στ)

Οι πιο συνηθισμένες εφαρμογές των ταιριασμάτων αναφέρονται σε διμερή γραφήματα. Ένα κλασικό παράδειγμα όπου μας ενδιαφέρει να βρούμε το μέγιστο ταίριασμα είναι το εξής: Έστω ένα διμερές γράφημα που αφορά το πρόγραμμα άσκησης ενός Τμήματος. Οι κορυφές του διμερούς γραφήματος αντιπροσωπεύουν

[10] Ένα τέλειο ταίριασμα ονομάζεται και 1-**παράγοντας** (1-factor) του G (οι δεσμοί του M αποτελούν ένα γεννητικό υπογράφημα του G του οποίου οι κορυφές έχουν βαθμό 1.)

αφενός τους φοιτητές και αφετέρου διαθέσιμες θέσεις πρακτικής άσκησης. Οι δεσμοί του ορίζονται με βάση τις προτιμήσεις των φοιτητών. Κάθε δεσμός συνδέει ένα φοιτητή με μια θέση που τον ενδιαφέρει. Ένας φοιτητής μπορεί να ενδιαφέρεται για πολλές θέσεις.

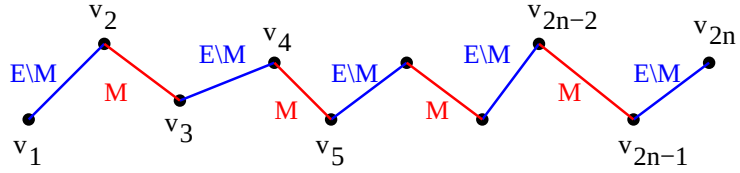


Ένα μέγιστο ταίριασμα στο γράφημα αυτό βρίσκει την πολυπληθέστερη ανάθεση, με τον μεγαλύτερο δυνατό αριθμό ενδιαφερόμενων να είναι ευχαριστημένοι.

Προφανώς, για κάθε ταίριασμα M ισχύει ότι $|M| \leq |E|$ και $|M| \leq |V|/2$. Επίσης, προφανώς, αν ένα γράφημα έχει περιττό αριθμό κορυφών, τότε δεν περιέχει τέλειο ταίριασμα.

Πως γνωρίζουμε αν ένα ταίριασμα M είναι μέγιστο; Ο Claude Berge έδωσε την απάντηση χρησιμοποιώντας την έννοια του M-αυξητικού μονοπατιού:

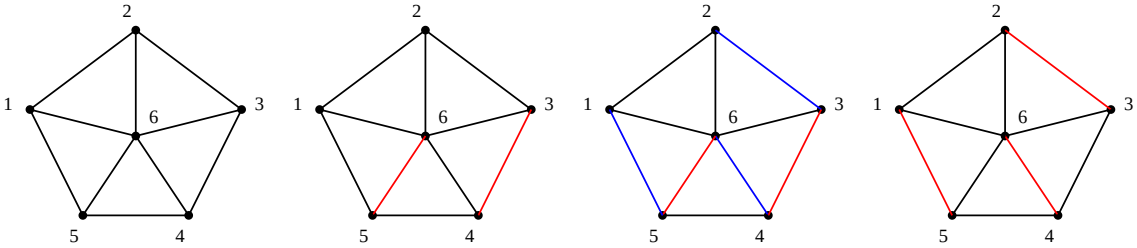
Ένα M-**αυξητικό** (M-augmenting) μονοπάτι στο G είναι ένα μονοπάτι P του οποίου οι δεσμοί ανήκουν εναλλάξ στα σύνολα $E \setminus M$ και M και τα άκρα του είναι ελεύθερες κορυφές.



Επομένως, το P θα έχει περιττό μήκος $2n-1$, $n \geq 1$ και άρα θα είναι της μορφής $P = (v_1, v_2, \ldots, v_{2n-1}, v_{2n})$ όπου οι n δεσμοί $\{v_1, v_2\}$, $\{v_3, v_4\}$, …, $\{v_{2i-1}, v_{2i}\}$, …, $\{v_{2n-1}, v_{2n}\}$ ανήκουν στο $E \setminus M$ και οι $n-1$ δεσμοί $\{v_2, v_3\}$, $\{v_4, v_5\}$, …, $\{v_{2i}, v_{2i+1}\}$, …, $\{v_{2n-2}, v_{2n-1}\}$ ανήκουν στο M.

Γενικότερα, ένα M-**εναλλασσόμενο** (M-alternating) μονοπάτι στο G είναι ένα μονοπάτι του οποίου οι δεσμοί ανήκουν εναλλάξ στα $E \setminus M$ και M.

Παράδειγμα : Το μονοπάτι $P = (2, 3, 4, 6, 5)$ του σχήματος (γ) είναι ένα M-αυξητικό μονοπάτι για το ταίριασμα $M = \{\{3, 4\}, \{5, 6\}\}$. του σχήματος (β).



(α) (β) (γ) (δ)

Πρόταση 66 (Claude Berge, 1973). *Ένα ταίριασμα M του G είναι μέγιστο ανν το G δεν περιέχει M-αυξητικό μονοπάτι.*

Απόδειξη. Αν υπάρχει M-αυξητικό μονοπάτι για το M, τότε το μέγεθος του ταιριάσματος μπορεί να αυξηθεί, οπότε το ταίριασμα αυτό δεν είναι μέγιστο.

Αν δεν υπάρχει τέτοιο M-αυξητικό μονοπάτι τότε αυτό είναι μέγιστο. Πράγματι, έστω ότι αυτό δεν ισχύει για κάποιο συγκεκριμένο ταίριασμα M σε ένα γράφημα G. Έστω M^* ένα μέγιστο ταίριασμα στο G. Σύμφωνα με την υπόθεση ο αριθμός των δεσμών του M^* είναι μεγαλύτερος τουλάχιστον κατά 1 από τον αριθμό των δεσμών του M, δηλαδή $|M^*| > |M|$. Θεωρούμε τους δεσμούς στο ταίριασμα που προκύπτει από την συμμετρική διαφορά $M \triangle M^* = (M \setminus M^*) \cup (M^* \setminus M)$, δηλαδή στο σύνολο όλων των δεσμών που ανήκουν ή στο M, ή στο M^*, αλλά όχι και στα δύο. Παρατηρήστε ότι $|M^* \setminus M| > |M \setminus M^*|$, επειδή εξ υποθέσεως $|M^*| > |M|$. Έστω G' το υπογράφημα του G που αποτελείται από τους δεσμούς του $M \triangle M^*$. Από τον ορισμό του ταιριάσματος, μια οποιασδήποτε κορυφή που ανήκει στο $G' \subseteq G$ μπορεί να περιέχεται σε ένα το πολύ δεσμό του M και σε ένα το πολύ δεσμό του M^*. Επομένως, καθεμιά από τις κορυφές του G' έχει βαθμό 2 ή μικρότερο και επομένως κάθε συνεκτική συνιστώσα του G' είναι είτε ένα μονοπάτι, είτε ένας κύκλος αρτίου μήκους, αποτελούμενος από εναλλασσόμενους δεσμούς, από τα $M \setminus M'$ και $M^* \setminus M$. Εφόσον $|M^*| > |M|$ και ο αριθμός των δεσμών που προέρχονται από τα $M \setminus M^*$ και $M^* \setminus M$ είναι ο ίδιος για οποιονδήποτε κύκλο αρτίου μήκους με εναλλάξ ακμές στο G', θα πρέπει να υπάρχει τουλάχιστον ένα μονοπάτι αποτελούμενο από εναλλάξ ακμές, το οποίο να ξεκινά και να τερματίζεται ένα δεσμό από το $M^* \setminus M$. Επομένως, αυτό είναι ένα M-αυξητικό μονοπάτι για το ταίριασμα M, κάτι που έρχεται σε αντίθεση με την υπόθεση ότι δεν υπάρχει τέτοιο M-αυξητικό μονοπάτι. □

Η αλγοριθμική αξία της πρότασης του Berge είναι πολύ σημαντική. Αν ένα ταίριασμα δεν είναι μέγιστο τότε πάντα υπάρχει ένα M-αυξητικό μονοπάτι, και με βάση αυτό το μονοπάτι μπορούμε να κατασκευάσουμε ένα ταίριασμα M' με μεγαλύτερο μέγεθος. Συγκεκριμένα, αν M είναι ένα ταίριασμα και $P = (v_1, v_2, \ldots, v_{2n-1}, v_{2n})$ ένα M-αυξητικό μονοπάτι μήκους $2n-1$, τότε σβήνοντας τους $n-1$ δεσμούς $\{v_2, v_3\}$, $\{v_4, v_5\}$, …, $\{v_{2i}, v_{2i+1}\}$, …, $\{v_{2n-2}, v_{2n-1}\}$ ανήκουν στο M και προσθέτοντας τους υπόλοιπους n δεσμούς $\{v_1, v_2\}$, $\{v_3, v_4\}$, …, $\{v_{2i-1}, v_{2i}\}$, …, $\{v_{2n-1}, v_{2n}\}$ του $E \setminus M$ προκύπτει ένα ταίριασμα M' του G με ένα παραπάνω δεσμό από ότι το M και δύο επιπλέον ταιριασμένες κορυφές.

Παρατηρήστε ότι αν μια κορυφή v είναι ταιριασμένη στο M τότε με αυτήν την κατασκευή, η v παραμένει ταιριασμένη και στο M' (αλλά αν εμφανίζεται στο M-αυξητικό μονοπάτι αλλάζει η κορυφή με την οποία ταιριάζει).

Υπάρχουν διάφοροι αλγόριθμοι για την εύρεση του μέγιστου ταιριάσματος σε ένα γράφημα που χρησιμοποιούν M-αυξητικά μονοπάτια. Ειδικά για διμερή γραφήματα είναι πολύ γνωστός ο **αλγόριθμος των Hopcroft και Karp** ο οποίος έχει πολυπλοκότητα $O(\sqrt{|V|} \times |E|)$.

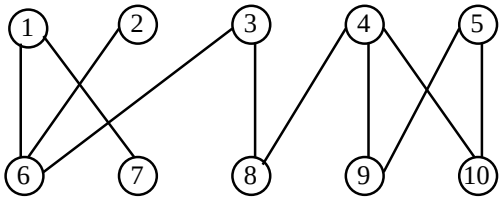
Στην συνέχεια δίδεται ο σκελετός ενός απλούστερου αλγόριθμου ο οποίος βρίσκει το μέγιστο ταίριασμα σε οποιοδήποτε γράφημα δεσμών και μπορεί να υλοποιηθεί με αναζήτηση σε βάθος ή αναζήτηση κατά πλάτος.

Αλγόριθμος εύρεσης των μέγιστων ταιριασμάτων.
Είσοδος: Ένα γράφημα δεσμών $G = (V, E)$
Έξοδος: Ένα μέγιστο ταίριασμα M του G
Η γενική περιγραφή του αλγορίθμου είναι η εξής:

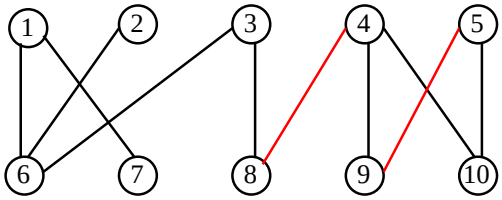
- Ξεκινάμε με ένα αρχικό ταίριασμα M (π.χ. κενό σύνολο δεσμών)
- (Επαναληπτικό βήμα) Βρίσκουμε ένα M-αυξητικό μονοπάτι και αυξάνουμε το τρέχον ταίριασμα M.
- Αν δεν μπορούμε να βρούμε τέτοιο μονοπάτι ο αλγόριθμος τερματίζεται και το ταίριασμα M που βρήκαμε είναι μέγιστο.

Σύμφωνα με την πρόταση του Berge αν το ταίριασμα δεν είναι μέγιστο υπάρχει πάντα ένα M-αυξητικό μονοπάτι. Πως βρίσκουμε ένα τέτοιο αυξητικό μονοπάτι P; Η ιδέα είναι απλή, ξεκινάμε από οποιαδήποτε ελεύθερη κορυφή v του G και χρησιμοποιώντας δεσμούς εναλλάξ μόνο από το $E \setminus M$ και το M προσπαθούμε να καταλήξουμε πάλι σε μια ελεύθερη κορυφή.

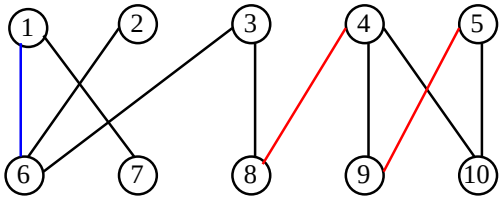
Παράδειγμα : Να βρεθεί ένα μέγιστο ταίριασμα για το διμερές γράφημα[11]



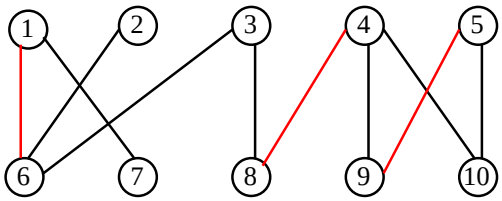
Δίδεται ως αρχικό ταίριασμα M το σύνολο των δεσμών $\{\{4, 8\}, \{5, 9\}\}$.



Όσο υπάρχουν ελεύθερες κορυφές, επιλέγουμε μια οποιαδήποτε ελεύθερη κορυφή, εδώ στο παράδειγμα την 1, και προσπαθούμε να κατασκευάσουμε ένα M-αυξητικό μονοπάτι P το οποίο καταλήγει σε ελεύθερη κορυφή. Οι γείτονες της 1 είναι οι κορυφές 6 και 7. Επιλέγουμε την κορυφή 6. Η κορυφή 6 είναι ελεύθερη, άρα βρήκαμε ένα M-αυξητικό μονοπάτι, το $P = (1, 6)$.

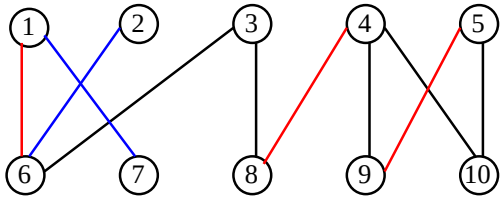


Οπότε προσθέτουμε τον δεσμό $\{1, 6\}$ στο M και προκύπτει το ταίριασμα $M = \{\{4, 8\}, \{5, 9\}, \{1, 6\}\}$.

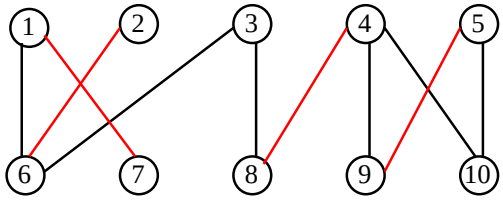


[11] Η μέθοδος λειτουργεί για οποιοδήποτε γράφημα

Επειδή υπάρχουν ελεύθερες κορυφές, επιλέγουμε μια από αυτές, εδώ στο παράδειγμα την 2, και προσπαθούμε πάλι να κατασκευάσουμε ενα M-αυξητικό μονοπάτι P (το οποίο καταλήγει σε ελεύθερη κορυφή). Ο μοναδικός γείτονας της 2 είναι η 6, η οποία είναι ταιριασμένη με την 1, άρα κατασκευάζουμε το M-εναλλασσόμενο μονοπάτι $P = (2, 6, 1)$. Συνεχίζουμε μέχρι να βρεθούμε σε αδιέξοδο ή να καταλήξουμε σε μια ελεύθερη κορυφή. Η κορυφή 1 έχει και άλλο γείτονα εκτός από την 6, την κορυφή 7. Η κορυφή 7 είναι ελεύθερη, άρα βρήκαμε ένα M-αυξητικό μονοπάτι, το $P = (2, 6, 1, 7)$.



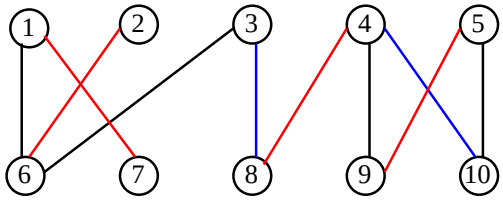
Οπότε αφαιρούμε τον δεσμό $\{16\}$ από το M και προσθέτουμε τους δεσμούς $\{2, 6\}$ και $\{1, 7\}$ και προκύπτει το ταίριασμα $M = \{\{4, 8\}, \{5, 9\}, \{2, 6\}, \{1, 7\}\}$.



Επειδή υπάρχουν ελεύθερες κορυφές, επιλέγουμε μια άλλη ελεύθερη κορυφή, εδώ στο παράδειγμα την 3, και προσπαθούμε να κατασκευάσουμε ένα M-αυξητικό μονοπάτι P. Η κορυφή 3 έχει δύο γείτονες, την κορυφή 6 και την κορυφή 8.

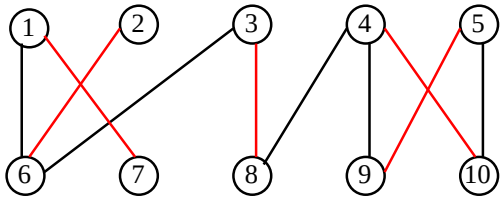
Επιλέγουμε την κορυφή 6 η οποία είναι ταιριασμένη με την κορυφή 2 οπότε το M-εναλλασσόμενο μονοπάτι P γίνεται $P = (3, 6, 2)$. Όμως η 2 δεν είναι δεν έχει άλλους γείτονες και δεν είναι ελεύθερη, οπότε αυτή η επιλογή οδηγεί σε αδιέξοδο.

Επιλέγουμε την κορυφή 8 η οποία είναι ταιριασμένη με την κορυφή 4 οπότε το M-εναλλασσόμενο μονοπάτι P γίνεται $P = (3, 8, 4)$. Η κορυφή 4 έχει δύο γείτονες, την κορυφή 9 και την 10. Επιλέγουμε την κορυφή 10 η οποία είναι ελεύθερη, άρα βρήκαμε ένα M-αυξητικό μονοπάτι, το $P = (3, 8, 4, 10)$.



Οπότε σβήνουμε τον δεσμό $\{8, 4\}$ από το M και προσθέτουμε τους δεσμούς $\{3, 8\}$ και $\{4, 10\}$ και προκύπτει το ταίριασμα

$$M = \{5, 9\}, \{2, 6\}, \{1, 7\}, \{3, 8\}, \{4, 10\}.$$



Επειδή δεν υπάρχουν ελεύθερες κορυφές (άρα δεν υπάρχουν ούτε M-αυξητικά μονοπάτια) ο αλγόριθμος ολοκληρώθηκε. Άρα το ταίριασμα

$$M = \{5, 9\}, \{2, 6\}, \{1, 7\}, \{3, 8\}, \{4, 10\}.$$

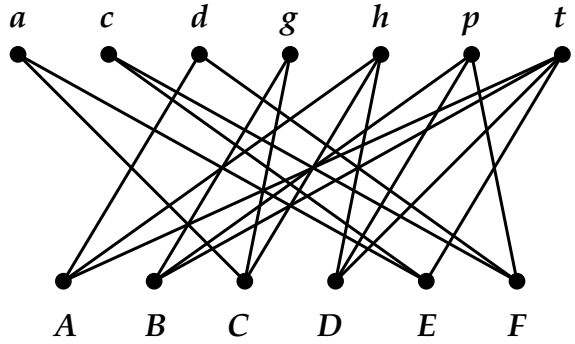
είναι ένα μέγιστο ταίριασμα του G, το οποίο εδώ στο παράδειγμα είναι και τέλειο.

Άσκηση 15. *Σε κάποιο project είναι προγραμματισμένες να γίνουν έξι εργασίες A, B, C, D, E, F. Για την εκτέλεση καθεμιάς από αυτές τις εργασίες είναι διαθέσιμοι οι παρακάτω συνεργάτες a, c, d, g, h, p, t. Κάθε εργασία μπορεί να διεκπεραιωθεί μόνο από ορισμένους συνεργάτες, σύμφωνα με τον παρακάτω πίνακα:*

A: d, h, t	B: g, p, t	C: a, g, k
D: h, p, t	E: a, c, t	F: c, d, p

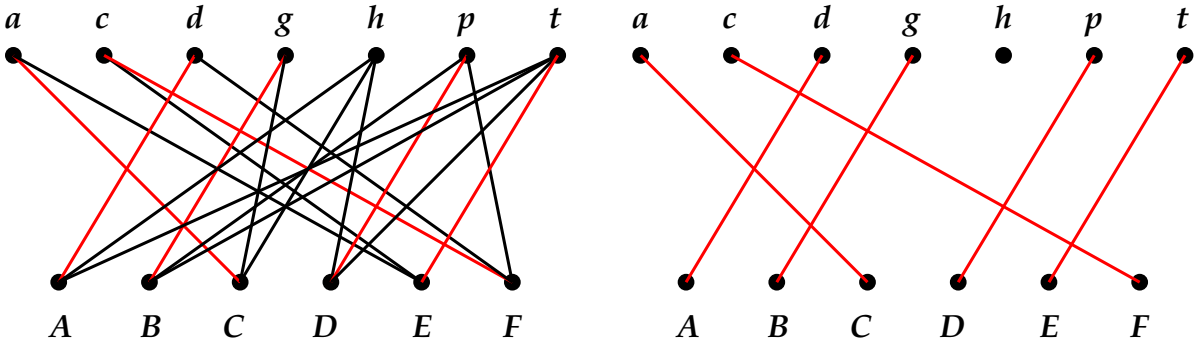
Πως πρέπει να γίνει η ανάθεση των εργασιών ώστε να καλυφθούν όλες οι εργασίες από ένα συνεργάτη η κάθε μία;

Λύση. Μπορούμε να μοντελοποιήσουμε το πρόβλημα χρησιμοποιώντας το επόμενο διμερές γράφημα:



Οι κορυφές του γραφήματος αποτελούνται αφενός από το σύνολο $U = \{A, B, C, D, E, F\}$ των εργασιών, και αφετέρου από το σύνολο $W = \{a, c, d, g, h, p, t\}$ των συνεργατών. Μια κορυφή $u \in U$ ενώνεται με μια κορυφή $w \in W$ αν ο u μπορεί να εκτελέση την εργασία w.

Μια λύση του προβλήματος δίδεται από το επόμενο μέγιστο ταίριασμα:



□

Τέλεια ταιριάσματα σε διμερή γραφήματα. Για κάθε υποσύνολο κορυφών $S \subseteq V$ του G, με $N(S)$ συμβολίζεται το σύνολο των κορυφών του G που είναι γειτονικές με τουλάχιστον μια κορυφή του S και ονομάζεται **γειτονιά** (neighbor) του S.

Πρόταση 67 (Philip Hall). *Έστω $G = (V, E)$ ένα διμερές γράφημα με διαμέριση κορυφών (U, W), και $|U| \leq |W|$. Τότε το G έχει ταίριασμα που περιέχει όλες τις κορυφές του U ανν*

$$|S| \leq |N(S)|$$

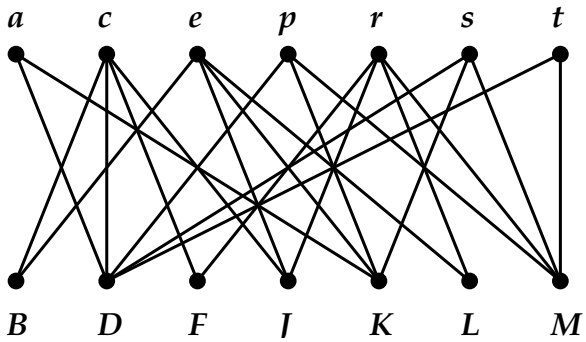
για κάθε υποσύνολο S του U.

Παράδειγμα : Επτά φοιτητές B, D, F, J, K, L, M ενδιαφέρονται να κάνουν πρακτική άσκηση σε επτά προσφερόμενες θέσεις a, c, e, p, r, s, t. Κάθε ένας από τους επτά έχει κάνει αίτηση σε κάποιες από αυτές:

B: c, e	D: a, c, p, s, t	F: c, r
J: c, e, r	K: a, e, p, s	L: e, r
M: p, r, s, t		

Είναι δυνατόν κάθε φοιτητής να κάνει πρακτική σε κάποια από τις θέσεις που τον ενδιαφέρουν;

Λύση. Το πρόβλημα αυτό μπορεί να μοντελοποιηθεί ως πρόβλημα εύρεσης τέλειου ταιριάσματος στο επόμενο διμερές γράφημα.



Οι κορυφές του γραφήματος αποτελούνται αφενός από το σύνολο $U = \{B, D, F, J, K, L, M\}$ των φοιτητών, και αφετέρου από το σύνολο $W = \{a, c, e, p, r, s, t\}$ των προσφερόμενων θέσεων. Μια κορυφή $u \in U$ ενώνεται με μια κορυφή $w \in W$ αν ο u έχει κάνει αίτηση για την θέση w.

Αν θεωρήσουμε το υποσύνολο $X = \{B, F, J, L\}$ έχουμε ότι $N(X) = \{c, e, r\}$. Επειδή $|N(X)| = 3 < 4 = |X|$ από το θεώρημα του Hall προκύπτει ότι δεν υπάρχει ταίριασμα για το διμερές γράφημα με μέγεθος 7. Επομένως, δεν είναι δυνατόν να γίνει αντιστοίχιση των επτά φοιτητών στις επτά θέσεις έτσι ώστε όλοι να είναι ευχαριστημένοι. □

Πόρισμα 68. *Ένα k-κανονικό διμερές γράφημα έχει τέλειο ταίριασμα.*

Απόδειξη. Έστω (U, W) μια διαμέριση των κορυφών του σύμφωνα με την ιδιότητα του διμερούς γραφήματος. Για κάθε υποσύνολο S του U υπάρχουν $k|S|$ δεσμοί τον οποίων ακριβώς το ένα άκρο ανήκει στο S. Το άλλο άκρο κάθε δεσμού ανήκει στην γειτονιά του S, η οποία είναι υποσύνολο του W, δηλαδή $N(S) \subseteq W$.

Επειδή από την γειτονιά $N(S)$ ξεκινούν τουλάχιστον $k|S|$ δεσμοί, έπεται ότι $k|N(S)| \geq k|S|$, δηλαδή $|N(S)| \geq |S|$. □

Τέλεια ταιριάσματα σε γραφήματα δεσμών. Μια συνεκτική συνιστώσα ενός γραφήματος ονομάζεται άρτια ή περιττή ανάλογα αν περιέχει άρτιο ή περιττό πλήθος κορυφών. Έστω $O(G)$ το πλήθος των περιττών συνεκτικών συνιστωσών του G.

Πρόταση 69 (William Tutte, 1947). *Ένα γράφημα $G = (V, E)$ έχει τέλειο ταίριασμα ανν*

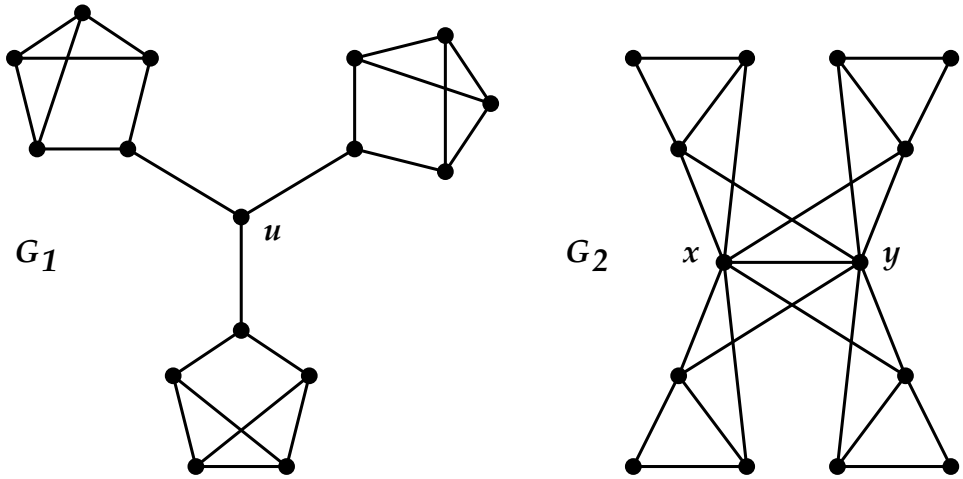
$$O(G - S) \leq |S|$$

για κάθε γνήσιο υποσύνολο S του V.

Με αντιθετοαναστροφή, το θεώρημα του Tutte μας λέει ότι αν υπάρχει κάποιο γνήσιο υποσύνολο S των κορυφών ενός γραφήματος G έτσι ώστε $O(G - S) > |S|$, τότε το G δεν έχει τέλειο ταίριασμα.

Έτσι, για παράδειγμα, αν το G έχει περιττό πλήθος κορυφών και S είναι το κενό σύνολο τότε $O(G - S) \geq 1 > 0 = |S|$, άρα είναι αδύνατο το G να έχει τέλειο ταίριασμα.

Πιο ενδιαφέροντα είναι τα επόμενα παραδείγματα: Τα γραφήματα G_1 και G_2 δεν έχουν τέλειο ταίριασμα.



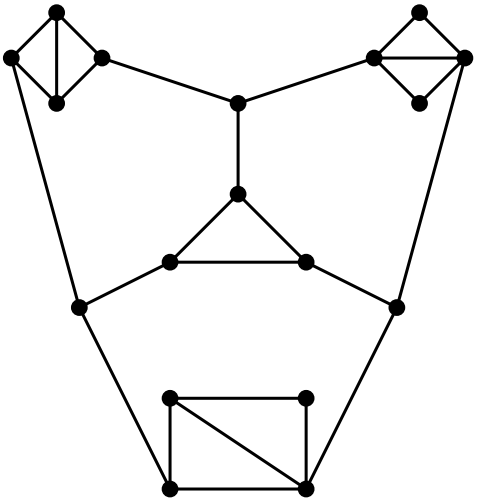
Πράγματι, αν $S_1 = \{u\}$, τότε $O(G_1 - S_1) = 3 > 1 = |S_1|$.

Επίσης, αν $S_2 = \{x, y\}$, τότε $O(G_2 - S_2) = 4 > 2 = |S_2|$.

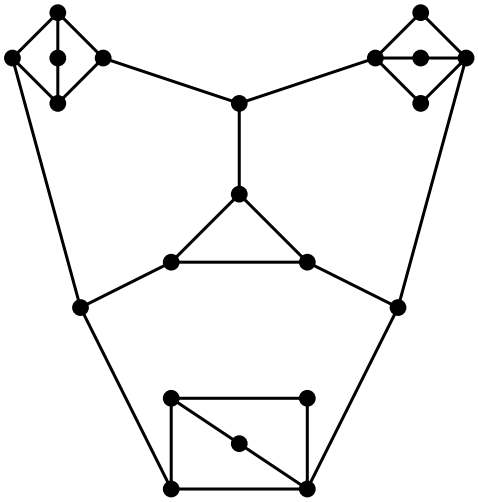
Άρα, από το θεώρημα του Tutte τα G_1, G_2 δεν έχουν τέλεια ταιριάσματα.

Άσκηση 16. *Να εξετασθεί αν τα επόμενα γραφήματα έχουν τέλειο ταίριασμα ή όχι.*

H_1



H_2



Πόρισμα 70 (Petersen, 1891). *Κάθε κυβικό γράφημα χωρίς γέφυρες έχει τέλειο ταίριασμα.*