Αλγόριθμοι και Πολυπλοκότητα

Αρχοντία Γιαννοπούλου
Όλγα Φουρτουνέλλη

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Άπληστοι Αλγόριθμοι

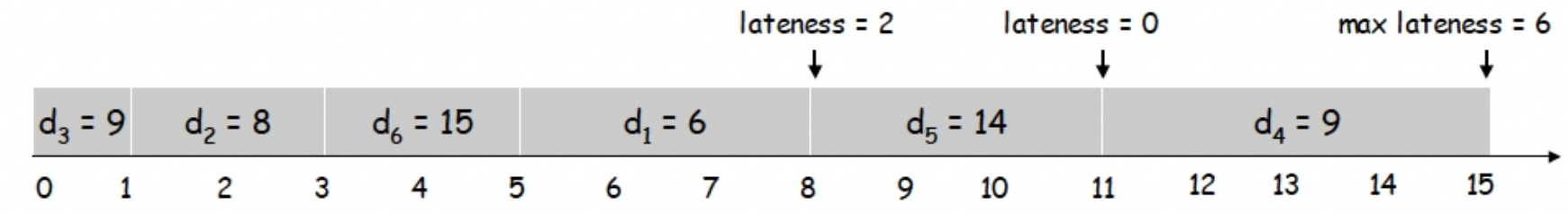
Χρονοπρογραμματισμός με ελάχιστη καθυστέρηση

Χρονοπρογραμματισμός με ελάχιστη καθυστέρηση

- Κάθε μηχανή επεξεργάζεται μία εργασία τη φορά.
- Η εργασία j απαιτεί t_j μονάδες χρόνου και έχει προθεσμία d_j.
- Άρα αν η εργασία j ξεκινήσει τη χρονική στιγμή s_j θα τελειώσει τη χρονική στιγμή $f_j = s_j + t_j$.
- Καθυστέρηση εργασίας j: $\ell_j = \max\{0, f_j - d_j\}$.

Στόχος: Χρονοπρογραμματισμός των εργασιών ελαχιστοποιώντας τη μέγιστη καθυστέρηση $L = \max_j\{\ell_j\}$.

	1	2	3	4	5	6
t_j	3	2	1	4	3	2
d_j	6	8	9	9	14	15



Ενδεχόμενα άπληστα κριτήρια

Άπληστο πρότυπο: Θεωρούμε τις εργασίες ταξινομημένες ως προς κάποια σειρά.

- Αύξουσα σειρά χρόνου επεξεργασίας των εργασιών (t_j)
- Αύξουσα σειρά προθεσμιών (d_j)
- Αύξουσα σειρά χρονικού περιθωρίου ($d_j - t_j$)

Αύξουσα σειρά χρόνου επεξεργασίας των εργασιών (t_j)

	1	2
t_j	1	10
d_j	100	10

Αύξουσα σειρά χρόνου επεξεργασίας των εργασιών (t_j)

	1	2
t_j	1	10
d_j	100	10

Δεν δουλεύει!

Αύξουσα σειρά χρονικού περιθωρίου $(d_j - t_j)$

	1	2
t_j	1	10
d_j	2	10

Αύξουσα σειρά χρονικού περιθωρίου $(d_j - t_j)$

	1	2
t_j	1	10
d_j	2	10

Δεν δουλεύει!

Αύξουσα σειρά προθεσμιών (d_j)

Ταξινόμησε τις εργασίες σε αύξουσα σειρά των προθεσμιών έτσι ώστε $d_1 \leq d_2 \leq \cdots \leq d_n$.

$t \leftarrow 0$
Για $j = 1$ έως n
    Αναθέτουμε την εργασία j στο διάστημα $[t, t + t_j]$
    $s_j \leftarrow t$, $f_j \leftarrow t + t_j$
    $t \leftarrow t + t_j$

Επίστρεψε τα διαστήματα $[s_j, f_j]$

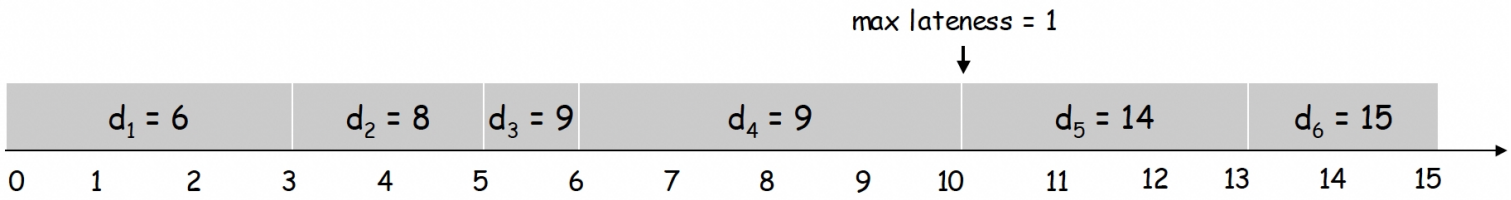
Αύξουσα σειρά προθεσμιών (d_j)

Ταξινόμησε τις εργασίες σε αύξουσα σειρά των προθεσμιών έτσι ώστε $d_1 \leq d_2 \leq \cdots \leq d_n$.

$t \leftarrow 0$
Για $j = 1$ έως n
    Αναθέτουμε την εργασία j στο διάστημα $[t, t + t_j]$
    $s_j \leftarrow t$, $f_j \leftarrow t + t_j$
    $t \leftarrow t + t_j$

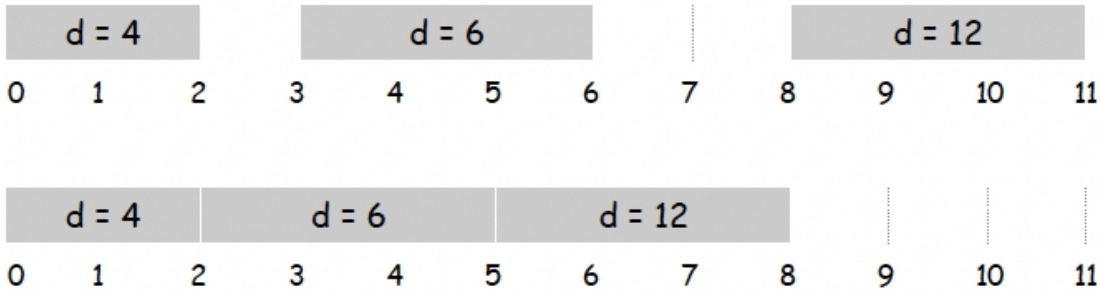
Επίστρεψε τα διαστήματα $[s_j, f_j]$

	1	2	3	4	5	6
t_j	3	2	1	4	3	2
d_j	6	8	9	9	14	15



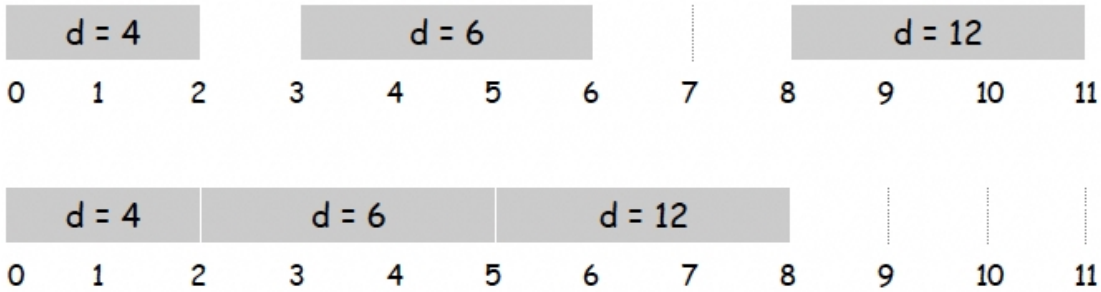
Αδρανής χρόνος

Παρατήρηση: Υπάρχει μία βέλτιστη λύση χωρίς αδρανή χρόνο (η μηχανή επεξεργάζεται πάντα κάποια εργασία).



Αδρανής χρόνος

Παρατήρηση: Υπάρχει μία βέλτιστη λύση χωρίς αδρανή χρόνο (η μηχανή επεξεργάζεται πάντα κάποια εργασία).



Παρατήρηση: Ο άπληστος αλγόριθμος δεν έχει αδρανή χρόνο.

Αντιστροφές

Μία **αντιστροφή** σε ένα χρονοδιάγραμμα S είναι ένα ζευγάρι εργασιών i και j τέτοιο ώστε $d_i < d_j$ αλλά η j προγραμματίζεται προς επεξεργασία πριν από την i.

Αντιστροφές

Μία **αντιστροφή** σε ένα χρονοδιάγραμμα S είναι ένα ζευγάρι εργασιών i και j τέτοιο ώστε $d_i < d_j$ αλλά η j προγραμματίζεται προς επεξεργασία πριν από την i.

Παρατήρηση: Ο άπληστος αλγόριθμος δεν έχει αντιστροφές.

Αντιστροφές

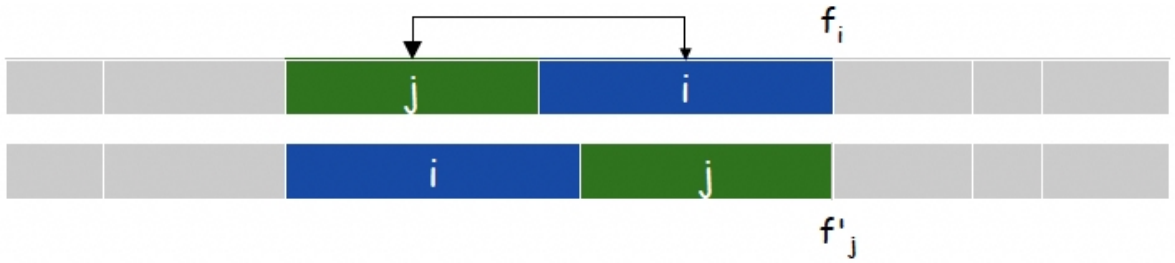
Μία **αντιστροφή** σε ένα χρονοδιάγραμμα S είναι ένα ζευγάρι εργασιών i και j τέτοιο ώστε $d_i < d_j$ αλλά η j προγραμματίζεται προς επεξεργασία πριν από την i.

Παρατήρηση: Ο άπληστος αλγόριθμος δεν έχει αντιστροφές.

Παρατήρηση: Αν ένα χρονοδιάγραμμα χωρίς αδρανή χρόνο έχει μία αντιστροφή τότε έχει μία αντιστροφή με ένα ζευγάρι αντεστραμμένων εργασιών διαδοχικά προγραμματισμένων.

Αντιστροφές

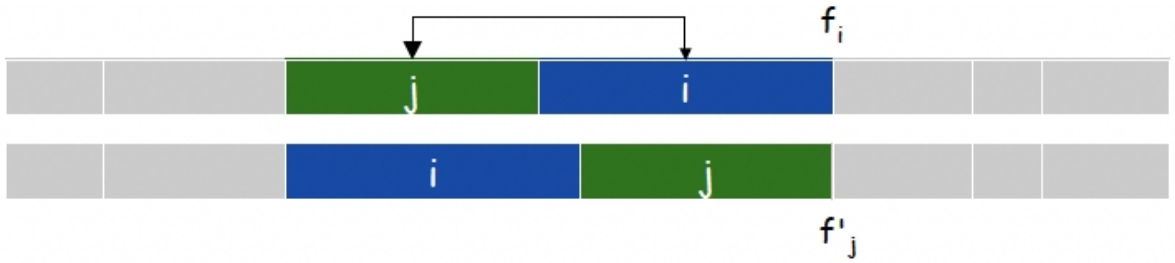
Ισχυρισμός: Η αντιμετάθεση δύο διαδοχικών, αντεστραμμένων εργασιών μειώνει τον αριθμό των αντιστροφών κατά 1 και δεν αυξάνει τη μέγιστη καθυστέρηση.



Απόδειξη: Έστω ℓ_k οι καθυστερήσεις στην αρχική λύση και έστω ℓ'_k οι καθυστερήσεις μετά την αντιμετάθεση. Τότε:

Αντιστροφές

Ισχυρισμός: Η αντιμετάθεση δύο διαδοχικών, αντεστραμμένων εργασιών μειώνει τον αριθμό των αντιστροφών κατά 1 και δεν αυξάνει τη μέγιστη καθυστέρηση.

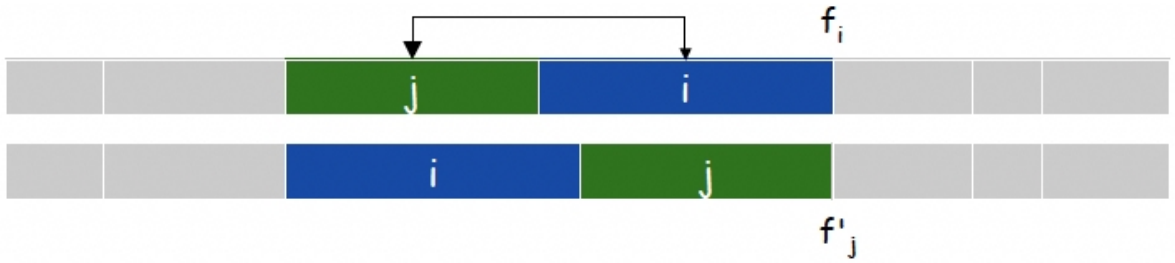


Απόδειξη: Έστω ℓ_k οι καθυστερήσεις στην αρχική λύση και έστω ℓ'_k οι καθυστερήσεις μετά την αντιμετάθεση. Τότε:

- $\ell_k = \ell'_k$ για κάθε $k \neq i, j$

Αντιστροφές

Ισχυρισμός: Η αντιμετάθεση δύο διαδοχικών, αντεστραμμένων εργασιών μειώνει τον αριθμό των αντιστροφών κατά 1 και δεν αυξάνει τη μέγιστη καθυστέρηση.

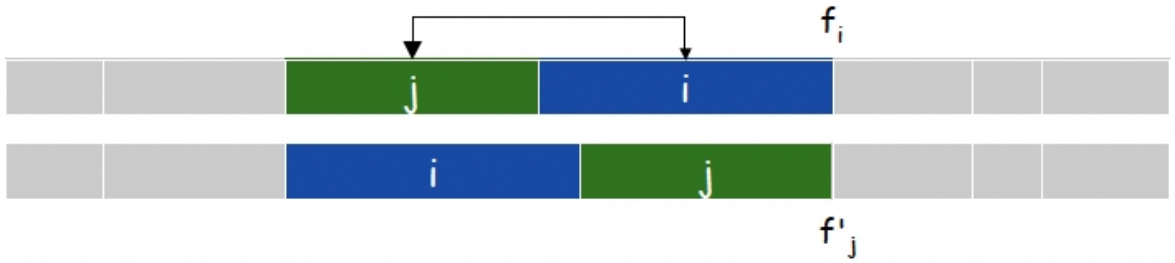


Απόδειξη: Έστω ℓ_k οι καθυστερήσεις στην αρχική λύση και έστω ℓ'_k οι καθυστερήσεις μετά την αντιμετάθεση. Τότε:

- $\ell_k = \ell'_k$ για κάθε $k \neq i, j$
- $\ell'_i \leq \ell_i$

Αντιστροφές

Ισχυρισμός: Η αντιμετάθεση δύο διαδοχικών, αντεστραμμένων εργασιών μειώνει τον αριθμό των αντιστροφών κατά 1 και δεν αυξάνει τη μέγιστη καθυστέρηση.



Απόδειξη: Έστω ℓ_k οι καθυστερήσεις στην αρχική λύση και έστω ℓ'_k οι καθυστερήσεις μετά την αντιμετάθεση. Τότε:

- $\ell_k = \ell'_k$ για κάθε $k \neq i, j$
- $\ell'_i \leq \ell_i$
- $\ell'_j = f'_j - d_j = f_i - d_j \leq f_i - d_i = \ell_i$

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

- Ορίζουμε S^* μία βέλτιστη λύση με τον ελάχιστο αριθμό αντιστροφών.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

- Ορίζουμε S^* μία βέλτιστη λύση με τον ελάχιστο αριθμό αντιστροφών.
- Υποθέτουμε ότι το S^* δεν έχει αδρανή χρόνο.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

- Ορίζουμε S^* μία βέλτιστη λύση με τον ελάχιστο αριθμό αντιστροφών.
- Υποθέτουμε ότι το S^* δεν έχει αδρανή χρόνο.
- Αν το S^* δεν έχει αντιστροφές, τότε $S = S^*$.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

- Ορίζουμε S^* μία βέλτιστη λύση με τον ελάχιστο αριθμό αντιστροφών.
- Υποθέτουμε ότι το S^* δεν έχει αδρανή χρόνο.
- Αν το S^* δεν έχει αντιστροφές, τότε $S = S^*$.
- Αν το S^* έχει τουλάχιστον μία αντιστροφή, τότε έστω i, j μία διαδοχική αντιστροφή.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

- Ορίζουμε S^* μία βέλτιστη λύση με τον ελάχιστο αριθμό αντιστροφών.
- Υποθέτουμε ότι το S^* δεν έχει αδρανή χρόνο.
- Αν το S^* δεν έχει αντιστροφές, τότε $S = S^*$.
- Αν το S^* έχει τουλάχιστον μία αντιστροφή, τότε έστω i, j μία διαδοχική αντιστροφή.

  Αντιμεταθέτωντας τις εργασίες i και j παίρνουμε μία λύση με αυστηρά μικρότερο αριθμό αντιστροφών που δεν αυξάνει τη μέγιστη καθυστέρηση.

Ανάλυση άπληστου αλγόριθμου

Θεώρημα: Η λύση S του άπληστου αλγόριθμου είναι βέλτιστη.

Θα δείξουμε το θεώρημα δείχνοντας ότι μία οποιαδήποτε βέλτιστη λύση μπορεί να μετατραπεί σε μία βέλτιστη λύση χωρίς αντιστροφές και άρα σέβεται το κριτήριο απληστίας που επιλέξαμε.

- Ορίζουμε S^* μία βέλτιστη λύση με τον ελάχιστο αριθμό αντιστροφών.
- Υποθέτουμε ότι το S^* δεν έχει αδρανή χρόνο.
- Αν το S^* δεν έχει αντιστροφές, τότε $S = S^*$.
- Αν το S^* έχει τουλάχιστον μία αντιστροφή, τότε έστω i, j μία διαδοχική αντιστροφή.

  Αντιμεταθέτωντας τις εργασίες i και j παίρνουμε μία λύση με αυστηρά μικρότερο αριθμό αντιστροφών που δεν αυξάνει τη μέγιστη καθυστέρηση.

  Άτοπο στην επιλογή του S^*.

Τεχνικές απόδειξης ορθότητας άπληστων κριτηρίων

- Η λύση του άπληστου αλγόριθμου υπερτερεί:
- Επιχείρημα ανταλλαγής:
- Δομική εξήγηση:

Τεχνικές απόδειξης ορθότητας άπληστων κριτηρίων

- Η λύση του άπληστου αλγόριθμου υπερτερεί:
  Δείχνουμε ότι μετά από κάθε βήμα του αλγόριθμου, η λύση του είναι τουλάχιστον τόσο καλή όσο η λύση οποιουδήποτε αλγόριθμου.

- Επιχείρημα ανταλλαγής:

- Δομική εξήγηση:

Τεχνικές απόδειξης ορθότητας άπληστων κριτηρίων

- Η λύση του άπληστου αλγόριθμου υπερτερεί:
  Δείχνουμε ότι μετά από κάθε βήμα του αλγόριθμου, η λύση του είναι τουλάχιστον τόσο καλή όσο η λύση οποιουδήποτε αλγόριθμου.

- Επιχείρημα ανταλλαγής:
  Μετατρέπουμε σταδιακά μία οποιαδήποτε λύση σε αυτή που υποδεικνύεται από τον άπληστο αλγόριθμο χωρίς να επηρεάζεται η βελτιστότητά της.

- Δομική εξήγηση:

Τεχνικές απόδειξης ορθότητας άπληστων κριτηρίων

- Η λύση του άπληστου αλγόριθμου υπερτερεί:
  Δείχνουμε ότι μετά από κάθε βήμα του αλγόριθμου, η λύση του είναι τουλάχιστον τόσο καλή όσο η λύση οποιουδήποτε αλγόριθμου.

- Επιχείρημα ανταλλαγής:
  Μετατρέπουμε σταδιακά μία οποιαδήποτε λύση σε αυτή που υποδεικνύεται από τον άπληστο αλγόριθμο χωρίς να επηρεάζεται η βελτιστότητά της.

- Δομική εξήγηση:
  Ανακαλύπτουμε/αποδεικνύουμε ένα δομικό όριο που να υποδηλώνει ένα κάτω/άνω φράγμα στη λύση και δείχνουμε ότι ο αλγόριθμος μας επιτυγχάνει αυτό το φράγμα.

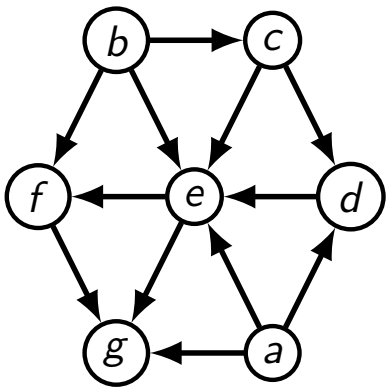
Τοπολογική Διάταξη

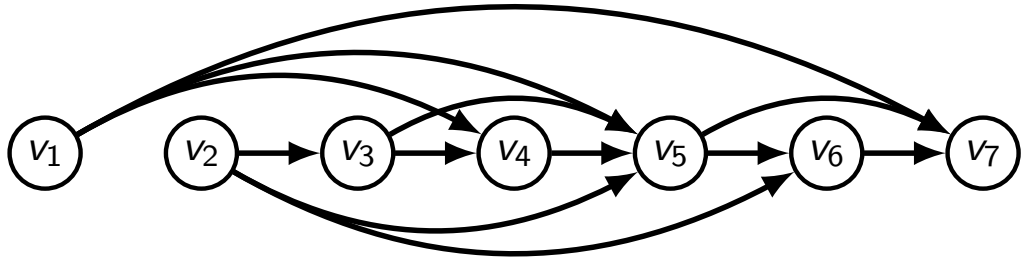
Τοπολογική Διάταξη

Μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος $G = (V, A)$ είναι μια διάταξη των κορυφών του v_1, v_2, ..., v_n τέτοια ώστε για κάθε (v_i, v_j) να ισχύει $i < j$.

Τοπολογική Διάταξη

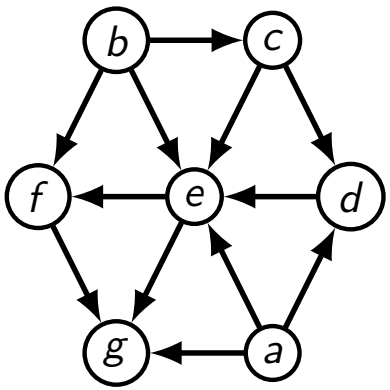
Μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος $G = (V, A)$ είναι μια διάταξη των κορυφών του v_1, v_2, ..., v_n τέτοια ώστε για κάθε (v_i, v_j) να ισχύει $i < j$.

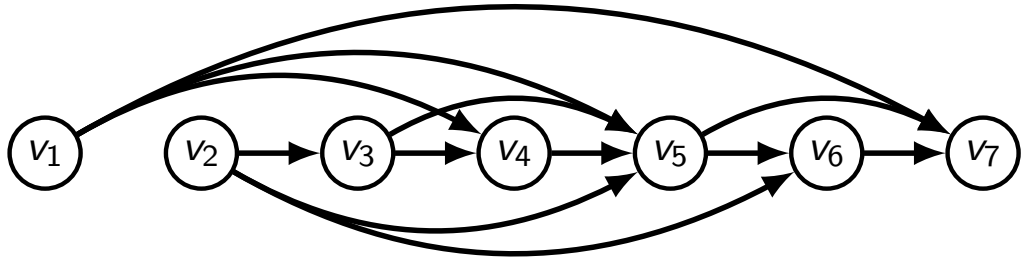




Τοπολογική Διάταξη

Μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος $G = (V, A)$ είναι μια διάταξη των κορυφών του v_1, v_2, ..., v_n τέτοια ώστε για κάθε (v_i, v_j) να ισχύει $i < j$.

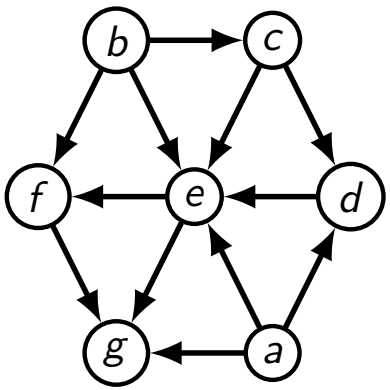


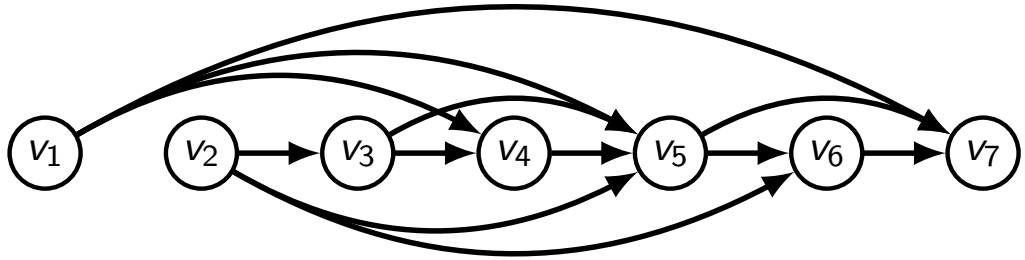


Έχουν όλα τα κατευθυνόμενα γραφήματα τοπολογική διάταξη;

Τοπολογική Διάταξη

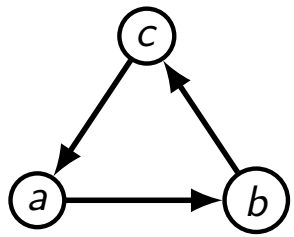
Μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος $G = (V, A)$ είναι μια διάταξη των κορυφών του v_1, v_2, ..., v_n τέτοια ώστε για κάθε (v_i, v_j) να ισχύει $i < j$.





Έχουν όλα τα κατευθυνόμενα γραφήματα τοπολογική διάταξη;

ΟΧΙ



Άκυκλα κατευθυνόμενα γραφήματα (DAG)

Θεώρημα: Ένα κατευθυνόμενο γράφημα έχει τοπολογική διάταξη αν και μόνο αν είναι άκυκλο (Directed Acyclic Graph).

Άκυκλα κατευθυνόμενα γραφήματα (DAG)

Θεώρημα: Ένα κατευθυνόμενο γράφημα έχει τοπολογική διάταξη αν και μόνο αν είναι άκυκλο (Directed Acyclic Graph).

Ερμηνεία: Η ακμή (v_i, v_j) σημαίνει ότι η εργασία v_i πρέπει να προηγηθεί της v_j.

Άκυκλα κατευθυνόμενα γραφήματα (DAG)

Θεώρημα: Ένα κατευθυνόμενο γράφημα έχει τοπολογική διάταξη αν και μόνο αν είναι άκυκλο (Directed Acyclic Graph).

Ερμηνεία: Η ακμή (v_i, v_j) σημαίνει ότι η εργασία v_i πρέπει να προηγηθεί της v_j.

Εφαρμογές - Παραδείγματα

- Γράφημα προαπαιτούμενων μαθημάτων: το μάθημα v_i πρέπει να διδαχθεί πριν από το μάθημα v_j.
- Μεταγλώττιση: το κομμάτι κώδικα v_i πρέπει να μεταγλωττιστεί πριν από το v_j.
- Διοχέτευση υπολογιστικών εργασιών: η έξοδος της εργασίας v_i χρειάζεται για τον προσδιορισμό της εισόδου της εργασίας v_j.
- Υπολογισμός παραστάσεων, π.χ. $((a + b) \cdot (c + d) + e) \cdot (a + b)$.

Απόδειξη

Αν το G έχει τοπολογική διάταξη, τότε το G είναι ένα DAG.

- Υποθέστε ότι το G έχει τοπολογική διάταξη $v_1, \ldots, v_n$ και ότι το G έχει επίσης έναν κατευθυνόμενο κύκλο C.

Απόδειξη

Αν το G έχει τοπολογική διάταξη, τότε το G είναι ένα DAG.

- Υποθέστε ότι το G έχει τοπολογική διάταξη $v_1, \ldots, v_n$ και ότι το G έχει επίσης έναν κατευθυνόμενο κύκλο C.
- Έστω v_i η κορυφή με τον μικρότερο δείκτη στον C, και έστω v_j η κορυφή ακριβώς πριν από την v_i. Άρα η (v_j, v_i) είναι μια ακμή.

Απόδειξη

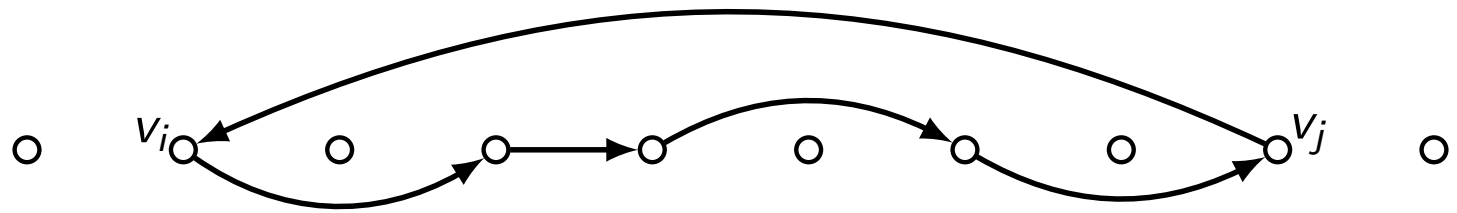
Αν το G έχει τοπολογική διάταξη, τότε το G είναι ένα DAG.

- Υποθέστε ότι το G έχει τοπολογική διάταξη $v_1, \ldots, v_n$ και ότι το G έχει επίσης έναν κατευθυνόμενο κύκλο C.
- Έστω v_i η κορυφή με τον μικρότερο δείκτη στον C, και έστω v_j η κορυφή ακριβώς πριν από την v_i. Άρα η (v_j, v_i) είναι μια ακμή.
- Από την επιλογή του i, έχουμε $i < j$.

Απόδειξη

Αν το G έχει τοπολογική διάταξη, τότε το G είναι ένα DAG.

- Υποθέστε ότι το G έχει τοπολογική διάταξη $v_1, \ldots, v_n$ και ότι το G έχει επίσης έναν κατευθυνόμενο κύκλο C.
- Έστω v_i η κορυφή με τον μικρότερο δείκτη στον C, και έστω v_j η κορυφή ακριβώς πριν από την v_i. Άρα η (v_j, v_i) είναι μια ακμή.
- Από την επιλογή του i, έχουμε $i < j$.
- Αφού η (v_j, v_i) είναι ακμή, και $v_1, \ldots, v_n$ είναι τοπολογική διάταξη, πρέπει να ισχύει $j < i$, άτοπο.



Απόδειξη

Αν το G είναι DAG, τότε το G έχει μία κορυφή χωρίς εισερχόμενες ακμές.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει μία κορυφή χωρίς εισερχόμενες ακμές.

- Ας υποθέσουμε ότι κάθε κορυφή έχει εισερχόμενες ακμές.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει μία κορυφή χωρίς εισερχόμενες ακμές.

- Ας υποθέσουμε ότι κάθε κορυφή έχει εισερχόμενες ακμές.
- Διαλέγουμε μία κορυφή v, και ακολουθούμε οπισθόδρομες ακμές από την v. Αφού η v έχει τουλάχιστον μια εισερχόμενη ακμή (u, v) μπορούμε να προχωρήσουμε οπισθόδρομα στην u.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει μία κορυφή χωρίς εισερχόμενες ακμές.

- Ας υποθέσουμε ότι κάθε κορυφή έχει εισερχόμενες ακμές.
- Διαλέγουμε μία κορυφή v, και ακολουθούμε οπισθόδρομες ακμές από την v. Αφού η v έχει τουλάχιστον μια εισερχόμενη ακμή (u, v) μπορούμε να προχωρήσουμε οπισθόδρομα στην u.
- Αφού η u έχει τουλάχιστον μια εισερχόμενη ακμή (x, u), προχωράμε οπισθόδρομα στην x.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει μία κορυφή χωρίς εισερχόμενες ακμές.

- Ας υποθέσουμε ότι κάθε κορυφή έχει εισερχόμενες ακμές.
- Διαλέγουμε μία κορυφή v, και ακολουθούμε οπισθόδρομες ακμές από την v. Αφού η v έχει τουλάχιστον μια εισερχόμενη ακμή (u, v) μπορούμε να προχωρήσουμε οπισθόδρομα στην u.
- Αφού η u έχει τουλάχιστον μια εισερχόμενη ακμή (x, u), προχωράμε οπισθόδρομα στην x.
- Συνεχίζουμε μέχρι να συναντήσουμε μία κορυφή, έστω w, δύο φορές.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει μία κορυφή χωρίς εισερχόμενες ακμές.

- Ας υποθέσουμε ότι κάθε κορυφή έχει εισερχόμενες ακμές.
- Διαλέγουμε μία κορυφή v, και ακολουθούμε οπισθόδρομες ακμές από την v. Αφού η v έχει τουλάχιστον μια εισερχόμενη ακμή (u, v) μπορούμε να προχωρήσουμε οπισθόδρομα στην u.
- Αφού η u έχει τουλάχιστον μια εισερχόμενη ακμή (x, u), προχωράμε οπισθόδρομα στην x.
- Συνεχίζουμε μέχρι να συναντήσουμε μία κορυφή, έστω w, δύο φορές.
- Έστω C η ακολουθία των κορυφών μεταξύ των δύο επισκέψεων της w. Η ακολουθία C είναι ένας κύκλος. Άτοπο.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

- Για $n = 1$ ο ισχυρισμός είναι προφανής.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

- Για $n = 1$ ο ισχυρισμός είναι προφανής.
- Έστω ένα DAG G με $k + 1$ κορυφές, για $k \geq 0$.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

- Για $n = 1$ ο ισχυρισμός είναι προφανής.
- Έστω ένα DAG G με $k + 1$ κορυφές, για $k \geq 0$.
- Από την προηγούμενη διαφάνεια το G έχει μία κορυφή, έστω v χωρίς εισερχόμενες ακμές.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

- Για $n = 1$ ο ισχυρισμός είναι προφανής.
- Έστω ένα DAG G με $k + 1$ κορυφές, για $k \geq 0$.
- Από την προηγούμενη διαφάνεια το G έχει μία κορυφή, έστω v χωρίς εισερχόμενες ακμές.
- Τότε το $G \setminus \{v\}$ είναι DAG και από την επαγωγική υπόθεση έχει μία τοπολογική διάταξη.

Απόδειξη

Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

- Για $n = 1$ ο ισχυρισμός είναι προφανής.
- Έστω ένα DAG G με $k + 1$ κορυφές, για $k \geq 0$.
- Από την προηγούμενη διαφάνεια το G έχει μία κορυφή, έστω v χωρίς εισερχόμενες ακμές.
- Τότε το $G \setminus \{v\}$ είναι DAG και από την επαγωγική υπόθεση έχει μία τοπολογική διάταξη.
- Κατασκευάζουμε μία τοπολογική διάταξη του G επαυξάνοντας την τοπολογική διάταξη του $G \setminus \{v\}$ τοποθετώντας πρώτη την v.

Απόδειξη

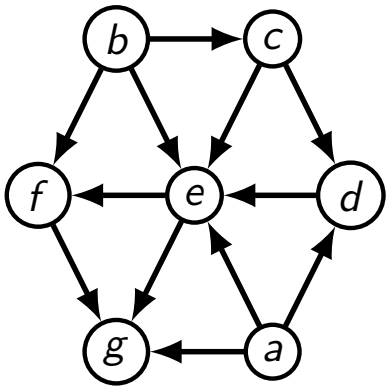
Αν το G είναι DAG, τότε το G έχει τοπολογική διάταξη.

Το αποδεικνύουμε με επαγωγή στο πλήθος των κορυφών, n, του G.

- Για $n = 1$ ο ισχυρισμός είναι προφανής.
- Έστω ένα DAG G με $k + 1$ κορυφές, για $k \geq 0$.
- Από την προηγούμενη διαφάνεια το G έχει μία κορυφή, έστω v χωρίς εισερχόμενες ακμές.
- Τότε το $G \setminus \{v\}$ είναι DAG και από την επαγωγική υπόθεση έχει μία τοπολογική διάταξη.
- Κατασκευάζουμε μία τοπολογική διάταξη του G επαυξάνοντας την τοπολογική διάταξη του $G \setminus \{v\}$ τοποθετώντας πρώτη την v.
- Αφού η v δεν έχει εισερχόμενες ακμές όλοι οι γείτονες της έπονται της v στη διάταξη και άρα είναι έγκυρη.

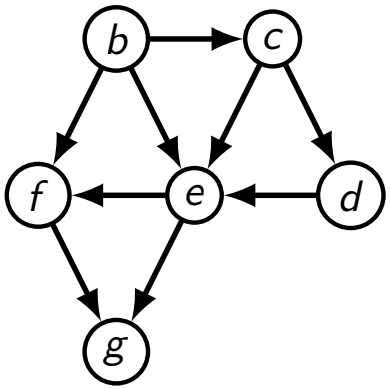
Αλγόριθμος τοπολογικής διάταξης

Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.



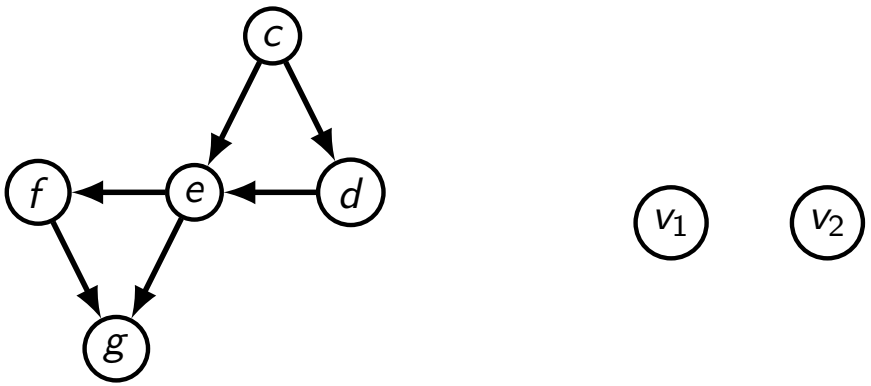
Αλγόριθμος τοπολογικής διάταξης

Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.



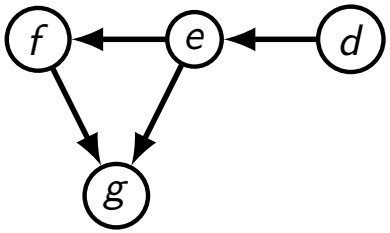
Αλγόριθμος τοπολογικής διάταξης

Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.



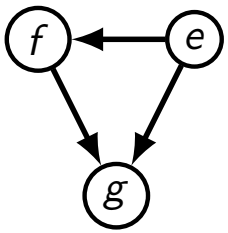
Αλγόριθμος τοπολογικής διάταξης

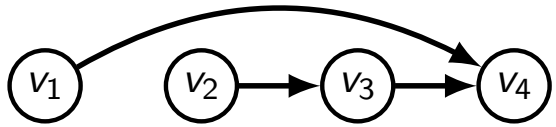
Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.



Αλγόριθμος τοπολογικής διάταξης

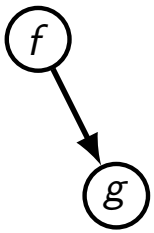
Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.

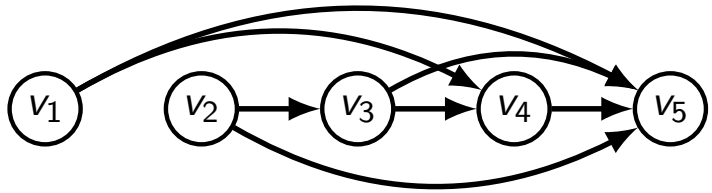




Αλγόριθμος τοπολογικής διάταξης

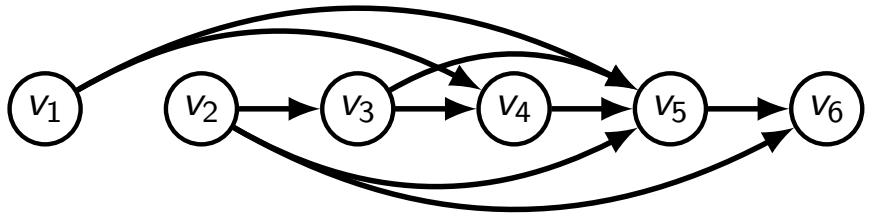
Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.





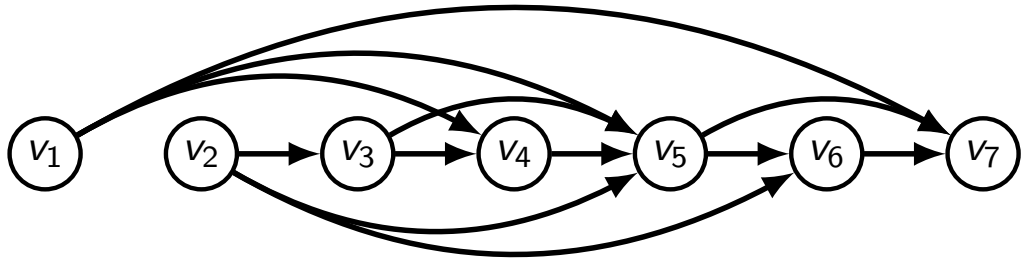
Αλγόριθμος τοπολογικής διάταξης

Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.



Αλγόριθμος τοπολογικής διάταξης

Τοπολογική διάταξη(G)
Βρες μία κορυφή v του G χωρίς εισερχόμενες ακμές
$L =$ Τοπολογική διάταξη$(G \setminus \{v\})$
Επίστρεψε $v \star L$.



Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Διατηρούμε τις παρακάτω πληροφορίες:

- $\mathrm{count}[w]$ = αριθμός απομενουσών εισερχόμενων ακμών του w
- S = σύνολο απομενουσών κορυφών χωρίς εισερχόμενες ακμές (λίστα)

Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Διατηρούμε τις παρακάτω πληροφορίες:

- $\mathrm{count}[w]$ = αριθμός απομενουσών εισερχόμενων ακμών του w
- S = σύνολο απομενουσών κορυφών χωρίς εισερχόμενες ακμές (λίστα)

Αρχικοποίηση: $\mathcal{O}(m+n)$ με μια μοναδική σάρωση του γραφήματος.

Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Διατηρούμε τις παρακάτω πληροφορίες:

- $\mathrm{count}[w]$ = αριθμός απομενουσών εισερχόμενων ακμών του w
- S = σύνολο απομενουσών κορυφών χωρίς εισερχόμενες ακμές (λίστα)

Αρχικοποίηση: $\mathcal{O}(m+n)$ με μια μοναδική σάρωση του γραφήματος.

Ενημέρωση: για τη διαγραφή της v

Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Διατηρούμε τις παρακάτω πληροφορίες:

- $\mathrm{count}[w]$ = αριθμός απομενουσών εισερχόμενων ακμών του w
- S = σύνολο απομενουσών κορυφών χωρίς εισερχόμενες ακμές (λίστα)

Αρχικοποίηση: $\mathcal{O}(m+n)$ με μια μοναδική σάρωση του γραφήματος.

Ενημέρωση: για τη διαγραφή της v

- Αφαίρεση της v από το S ($\mathcal{O}(1)$ χρόνος)

Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Διατηρούμε τις παρακάτω πληροφορίες:

- $\mathrm{count}[w]$ = αριθμός απομενουσών εισερχόμενων ακμών του w
- S = σύνολο απομενουσών κορυφών χωρίς εισερχόμενες ακμές (λίστα)

Αρχικοποίηση: $\mathcal{O}(m+n)$ με μια μοναδική σάρωση του γραφήματος.

Ενημέρωση: για τη διαγραφή της v

- Αφαίρεση της v από το S ($\mathcal{O}(1)$ χρόνος)
- Μείωση του $\mathrm{count}[w]$ για όλες τις ακμές από την v στην w, και πρόσθεση της w στο S αν το $\mathrm{count}[w]$ φτάσει στο 0

Πολυπλοκότητα

Ο αλγόριθμος βρίσκει (εφόσον υπάρχει) μια τοπολογική διάταξη ενός κατευθυνόμενου γραφήματος G σε χρόνο $\mathcal{O}(m+n)$.

Διατηρούμε τις παρακάτω πληροφορίες:

- $\mathrm{count}[w]$ = αριθμός απομενουσών εισερχόμενων ακμών του w
- S = σύνολο απομενουσών κορυφών χωρίς εισερχόμενες ακμές (λίστα)

Αρχικοποίηση: $\mathcal{O}(m+n)$ με μια μοναδική σάρωση του γραφήματος.

Ενημέρωση: για τη διαγραφή της v

- Αφαίρεση της v από το S ($\mathcal{O}(1)$ χρόνος)
- Μείωση του $\mathrm{count}[w]$ για όλες τις ακμές από την v στην w, και πρόσθεση της w στο S αν το $\mathrm{count}[w]$ φτάσει στο 0
- Εκτελούνται σε $\mathcal{O}(1)$ χρόνο ανά ακμή