Αν. Καθηγητής Π. Λουρίδας

Τμήμα Διοικητικής Επιστήμης και Τεχνολογίας

Οικονομικό Πανεπιστήμιο Αθηνών

Χρωματικότητα

Έστω ότι πρέπει να καταρτήσετε το πρόγραμμα εξετάσεων στο πανεπιστήμιο. Η εξέταση ενός μαθήματος δεν θα πρέπει να συμπέσει χρονικά με την εξέταση ενός άλλου μαθήματος του ίδιου έτους. Πώς μπορείτε να προγραμματίσετε τις εξετάσεις ώστε να αποφευχθούν τέτοιες συγκρούσεις;

Ένας τρόπος είναι να χρησιμοποιήσετε έναν γράφο. Οι κόμβοι του γράφου είναι τα εξεταζόμενα μαθήματα. Δύο μαθήματα συνδέονται μεταξύ τους στον γράφο αν ανήκουν στο ίδιο έτος. Για τη διεξαγωγή των εξετάσεων υπάρχουν συγκεκριμένες χρονικά διαστήματα, για παράδειγμα το διάστημα 1 μπορεί να είναι Δευτέρα 11-13, το διάστημα 2 Δευτέρα 13-15, κ.ό.κ. Για να προγραμματίσετε τότε τις εξετάσεις θα πρέπει να αποδώσετε σε κάθε μάθημα, δηλαδή κόμβο, ένα χρονικό διάστημα έτσι ώστε δύο γειτονικοί κόμβοι στο γράφο να μην έχουν το ίδιο χρονικό διάστημα.

Το πρόβλημα αυτό είναι το πρόβλημα του *χρωματισμού γράφου* (graph coloring). Θέλουμε να αποδώσουμε σε κάθε κόμβο μία ετικέτα, την οποία αποκαλούμε *χρώμα*, έτσι ώστε δύο γειτονικοί κόμβοι να μην έχουν το ίδιο χρώμα.

Βεβαίως, μια λύση είναι να αποδώσουμε σε κάθε κόμβο ένα διαφορετικό χρώμα, αλλά αυτό δεν είναι πολύ οικονομική λύση. Ιδανικά, θα θέλαμε να χρησιμοποιήσουμε τον ελάχιστο αριθμό χρωμάτων. Στην περίπτωση των εξετάσεων, αυτό θα σήμαινε ότι οι εξετάσεις θα ολοκληρωθούν χρησιμοποιώντας τα ελάχιστα διαθέσιμα χρονικά διαστήματα.

Αν ένας γράφος μπορεί να χρωματιστεί με k διαφορετικά χρώματα, ονομάζεται k-χρωματικός. Ο ελάχιστος αριθμός k χρωμάτων με τον οποίο μπορεί να χρωματιστεί ένας γράφος G όνομαζεται *χρωματικότητα* του γράφου, ή *χρωματικός δείκτης* (chromatic index) του γράφου και συμβολίζεται με $\chi(G)$. Παρά το ότι δεν είναι πάντα πρακτικά δυνατόν να βρούμε το χρωματικό δείκτη του γράφου και να τον χρωματίσουμε με τον ελάχιστο αριθμό χρωμάτων, ερευνητές έχουν προτείνει διάφορες μεθόδους με τις οποίες μπορούμε να χρωματίσουμε έναν γράφο με έναν αριθμό χρωμάτων μικρότερο από τον αριθμό των κόμβων, αν και μεγαλύτερο από τον χρωματικό δείκτη.

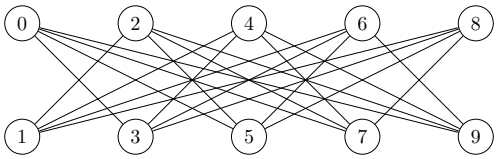
Πριν προχωρήσουμε, ας δούμε έναν ορισμό που θα χρησιμοποιήσουμε στη συνέχεια. Αν έχουμε έναν γράφο $G = (V, E)$, όπου V το σύνολο των κόμβων και E το σύνολο των ακμών, τότε ένας επαγόμενος υπογράφος H (induced subgraph) του G είναι ένας γράφος που παράγεται από ένα υποσύνολο των κόμβων και όλων των ακμών μεταξύ των κόμβων αυτού του υποσυνόλου. Αν το υποσύνολο των κόμβων είναι το $U \subset V$, ο επαγόμενος υπογράφος είναι ο $H(U, F)$ όπου:

$$F = \{(u, w) \mid u \in U, w \in U, (u, v) \in E\}$$

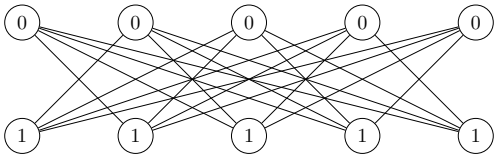
Ο Άπληστος Αλγόριθμος και ο Αλγόριθμος DSatur

Ο απλούστερος αλγόριθμος είναι να συμπεριφερθούμε *άπληστα* (greedy). Διατάσσουμε τους κόμβους του γράφου σε μια σειρά και τους παίρνουμε έναν-έναν. Χρωματίζουμε κάθε κόμβο με το μικρότερο χρώμα που δεν έχουμε χρωματίσει τους γείτονές του. Η υλοποίηση του αλγόριθμου είναι απλή (υπάρχει και στο σχετικό άρθρο στη Wikipedia). Η πολυπλοκότητα του αλγορίθμου είναι καλή: αν έχουμε n κόμβους και m συνδέσμους, ο αλγόριθμος απαιτεί χρόνο $O(n + m)$. Ο αριθμός των χρωμάτων που τελικά θα χρησιμοποιήσουμε εξαρτάται από τη διάταξη των κόμβων. Αν επιλέξουμε τη βέλτιστη διάταξη, τότε ο άπληστος αλγόριθμος θα χρησιμοποιήσει τον ελάχιστο αριθμό χρωμάτων, δηλαδή τον χρωματικό αριθμό του γράφου. Δυστυχώς όμως, δεν ξέρουμε ποια είναι αυτή η βέλτιστη διάταξη. Αν ο γράφος έχει n κόμβους, υπάρχουν $n!$ δυνατές διατάξεις των κόμβων του, οπότε αν το n δεν είναι μικρό είναι ανέφικτο να χρησιμοποιήσουμε τον άπληστο αλγόριθμο για να βρούμε το βέλτιστο χρωματισμό του γράφου.

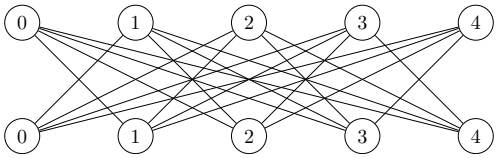
Για παράδειγμα, δείτε τον παρακάτω γράφο, ο οποίος είναι ένας *στεμματικός γράφος* (crown graph). Ένας στεμματικός γράφος είναι ένας γράφος με κόμβους $U = u_1, u_2, \ldots, u_n$ και $V = v_1, v_2, \ldots, v_n$ ο οποίος έχει έναν σύνδεσμο μεταξύ κάθε ζεύγους u_i, u_j με $i \neq j$.



Αν χρησιμοποιήσουμε τον άπληστο αλγόριθμο και χρωματίσουμε τους κόμβους από πάνω προς τα κάτω και από αριστερά προς τα δεξιά, δηλαδή $[0, 2, 4, 6, 8, 1, 3, 5, 7, 9]$ θα πάρουμε τον χρωματισμό που φαίνεται παρακάτω. Ο χρωματισμός αυτός είναι βέλτιστος και χρησιμοποιεί δύο χρώματα. Στην εικόνα έχουμε αντικαταστήσει τα ονόματα των κόμβων με έναν αριθμό που αντιστοιχεί στο χρώμα με τον οποίο τον έχουμε χρωματίσει. Στη συνέχεια όταν ζωγραφίζουμε ένα γράφο, οι κόμβοι του θα περιέχουν είτε τα ονόματά τους είτε τα χρώματά τους, αναλόγως τα συμφραζόμενα.



Αν όμως χρησιμοποιήσουμε τον άπληστο αλγόριθμο και χρωματίζουμε τους κόμβους με αύξουσα σειρά, δηλαδή $[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$ θα πάρουμε τον άλλο χρωματισμό που φαίνεται παρακάτω. Γενικότερα, ο άπληστος αλγόριθμος μπορεί στη χειρότερη περίπτωση να χρειαστεί $n/2$ χρώματα.



Εφόσον δεν γνωρίζουμε εκ των προτέρων ποια είναι η σειρά με την οποία πρέπει να χρωματίσουμε τους κόμβους, μπορούμε να ακολουθήσουμε *ευρεστικές μεθόδους* (heuristic methods). Μια τέτοια μέθοδος είναι να χρωματίσουμε τους κόμβους με φθίνουσα σειρά του βαθμού τους, δηλαδή πρώτα τον κόμβο με τον μεγαλύτερο βαθμό, μετά τον κόμβο με τον αμέσως μικρότερο βαθμό, κ.ο.κ. Σε περίπτωση που κόμβοι έχουν τον ίδιο βαθμό, τους παίρνουμε με αύξουσα σειρά με βάση το όνομά τους. Αυτή είναι η μέθοδος των Welsh και Powell και μας δίνει και ένα όριο στον χρωματικό αριθμό του γράφου. Αν με $d(v)$ συμβολίζουμε το βαθμό ενός κόμβου και έχουμε ονομάσει τους n κόμβους του γράφου έτσι ώστε $d(v_1) \geq d(v_2) \geq \cdots \geq d(v_n)$, τότε:

$$\chi(G) \leq \max_{i=1,2,\ldots,n} [\min(d(v_i) + 1, i)]$$

Ένας άλλος αλγόριθμος που μοιάζει με τον άπληστο αλγόριθμο είναι ο αλγόριθμος DSatur που αναπτύχθηκε από τον Daniel Brélaz (1979). Σε αυτόν τον αλγόριθμο, για να επιλέξουμε τον κόμβο που θα χρωματίσουμε, εξετάζουμε τους κόμβους που δεν έχουν χρωματιστεί και επιλέγουμε να χρωματίσουμε αυτόν που έχει τα περισσότερα χρώματα στη γειτονιά του. Αν υπάρχουν ισοπαλίες, επιλέγουμε τον κόμβο με το μεγαλύτερο βαθμό στον επαγόμενο υπογράφο των αχρωμάτιστων κόμβων. Ο αριθμός των χρωμάτων της γειτονιάς ενός κόμβου ονομάζεται *βαθμός κορεσμού* (degree of saturation), εξού και το όνομα του αλγορίθμου.

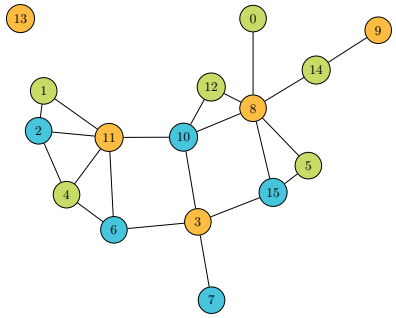
Ο Αλγόριθμος του Johnson

Μια διαφορετική προσέγγιση είναι να χρωματίσουμε τους κόμβους του γράφου βρίσκοντας *ανεξάρτητα σύνολα* (independent set) των κόμβων του γράφου. Ένα ανεξάρτητο σύνολο των κόμβων του γράφου είναι ένα σύνολο κόμβων, κανένας εκ των οποίων δεν είναι γείτονας ενός άλλου κόμβου στο σύνολο. Η προσέγγιση αυτή προτάθηκε από τον David S. Johnson (1974), και είναι ως εξής:

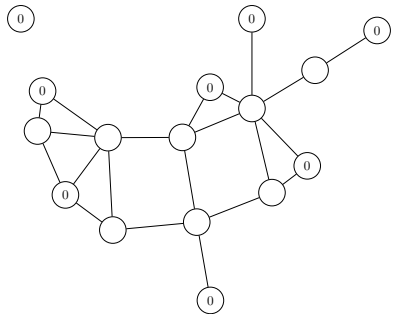
1. Έστω W το σύνολο των κόμβων που δεν έχουν χρωματιστεί ακόμα. Αν $W = \emptyset$ σταματάμε, αλλιώς $U \leftarrow W$.
2. Έστω u ένας κόμβος με τον ελάχιστο βαθμό στον επαγόμενο υπογράφο του U. Χρωματίζουμε τον u με το χρώμα i. Θέτουμε $U \leftarrow U - \{u\} - N(u)$.
3. Αν $U = \emptyset$, θέτουμε $i \leftarrow i + 1$ και επιστρέφουμε στο βήμα 2. Αλλιώς επιστρέφουμε στο βήμα 1.

Οι κόμβοι που χρωματίζονται στο βήμα 2 με το χρώμα i αποτελούν ανεξάρτητο σύνολο. Ο αλγόριθμος του Johnson έχει πολυπλοκότητα $O(n^2)$ και αποδεικνύεται ότι μπορεί να χρωματίσει κάθε k-χρωματικό γράφο χρησιμοποιώντας το πολύ $3n/\log_k(n)$ χρώματα.

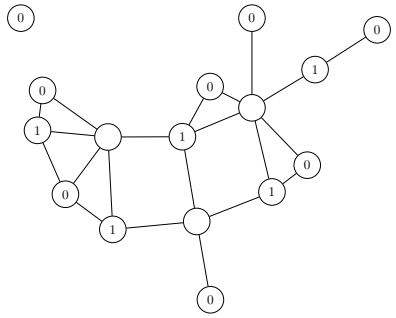
Στην παρακάτω εικόνα βλέπετε έναν 3-χρωματικό γράφο.



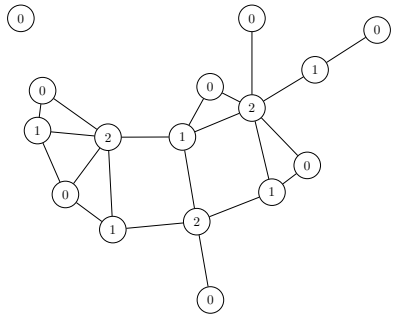
Αν χρησιμοποιήσουμε τον αλγόριθμο του Johnson, θα χρωματίσουμε πρώτα τους κόμβους $13, 0, 5, 7, 9, 12, 1, 4$, με αυτή τη σειρά, με το χρώμα 0. Οι κόμβοι αυτοί αποτελούν όπως μπορείτε να επιβεβαιώσετε ένα ανεξάρτητο σύνολο.



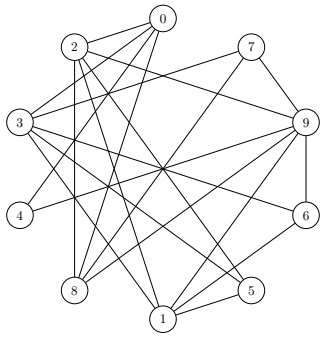
Στη συνέχεια, θα χρωματίσουμε τους κόμβους $2, 6, 10, 14, 15$, με αυτή τη σειρά, με το χρώμα 1. Οι κόμβοι αυτοί αποτελούν ένα δεύτερο ανεξάρτητο σύνολο.



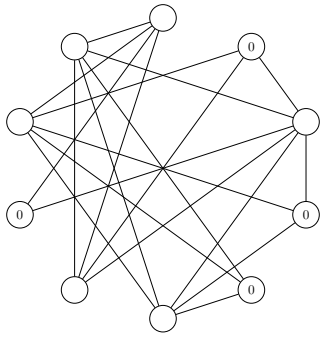
Τέλος, θα χρωματίσουμε τους κόμβους $3, 8, 11$, με αυτή τη σειρά, με το χρώμα 2. Οι κόμβοι αυτοί αποτελούν ένα τρίτο ανεξάρτητο σύνολο.



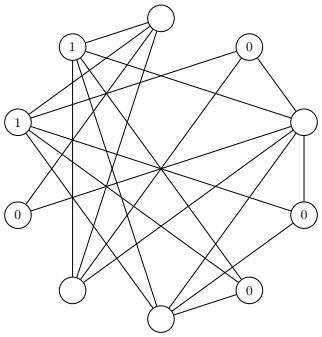
Ας δούμε τι γίνεται με έναν 4-χρωματικό γράφο. Αυτός είναι ένας γράφος Erdős-Renyi, ο οποίος δημιουργείται με μια στοχαστική διαδικασία, στην οποία δημιουργούμε ένα σύνδεσμο μεταξύ δύο συνδέσμων με μια πιθανότητα p που επιλέγουμε (στην περίπτωσή μας, $p = 0{,}4$).



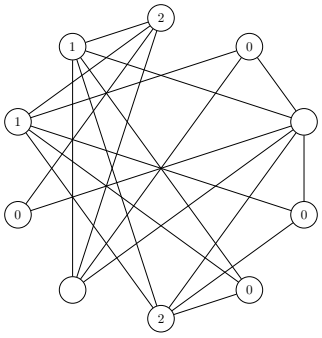
Στο γράφο αυτό, θα χρωματίσουμε πρώτα τους κόμβους $4, 6, 5, 7$, με αυτή τη σειρά, με το χρώμα 0.



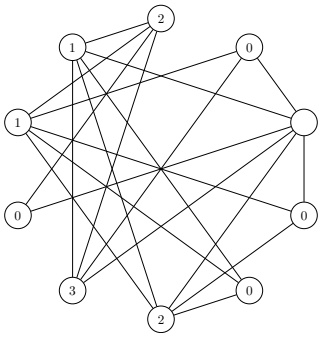
Στη συνέχεια, θα χρωματίσουμε τους κόμβους $3, 2$, με αυτή τη σειρά, με το χρώμα 1.



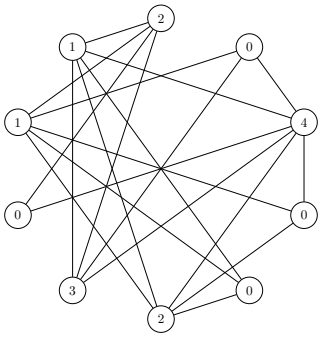
Κατόπιν θα χρωματίσουμε τους κόμβους 0, 1, με αυτή τη σειρά, με το χρώμα 2.



Τώρα ο κόμβος 8 θα χρωματιστεί με το χρώμα 3.



Τέλος, ο κόμβος 9 θα χρωματιστεί με το χρώμα 4.



Επομένως χρειαστήκαμε πέντε χρώματα συνολικά.

Ο Αλγόριθμος του Wigderson

Ένας διαφορετικός αλγόριθμος για τον χρωματισμό γράφων προτάθηκε από τον Avi Wigderson (1982, 1983). Ο αλγόριθμος αυτός είναι αναδρομικός. Αν ένας γράφος είναι k-χρωματικός, ο αλγόριθμος αναδρομικά προσπαθεί να χρωματίσει γράφους οι οποίοι είναι $(k-1)$-χρωματικοί, έως ότου καταλήξει να χρωματίζει γράφους οι οποίοι είναι 2-χρωματικοί.

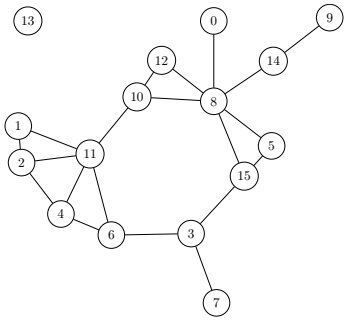
Ο αλγόριθμος καλείται με `Wigderson`(G, k, i). Οι παράμετροι είναι ο γράφος G που θέλουμε να χρωματίσουμε, ο αριθμός k των χρωμάτων που πιστεύουμε ότι μπορούν να χρησιμοποιηθούν, και το χρώμα i με το οποίο θέλουμε να ξεκινήσουμε. Ο αλγόριθμος θα χρωματίσει το γράφο G με τα χρώματα $i, i+1, ...$ Στον αλγόριθμο, με $\Delta(G)$ ονομάζουμε το μέγιστο βαθμό του γράφου G. Δηλαδή, αν $d(v)$ είναι ο βαθμός του κόμβου vμ $\Delta(G) = \max_{v \in V}\{(d(v)\}$. Επιπλέον, ορίζουμε τη συνάρτηση $f_k(n) = n^{1-1/(k-1)}$.

1. Έστω n ο αριθμός των κόμβων στο γράφο G.
2. Αν $k = 2$, χρωματίζουμε τον G με τα χρώματα i και $i+1$ και επιστρέφουμε 2. Αν δεν γίνεται να χρωματιστεί ο G με δύο χρώματα, τότε επιστρέφουμε -1.
3. Αν $k \geq \lg n$, χρωματίζουμε τον G με τα n χρώματα $i, i+1, ... i+n-1$, δηλαδή κάθε κόμβος παίρνει διαφορετικό χρώμα, και επιστρέφουμε n.
4. Όσο $\Delta(G) \geq \lceil f_k(n) \rceil$:
    1. Επιλέγουμε έναν κόμβο, έστω u, με βαθμό $\Delta(G)$.
    2. Παίρνουμε τον γράφο H που επάγεται από τη γειτονιά του u.
    3. Καλούμε αναδρομικά $j \leftarrow$ `Wigderson`$(H, k-1, i)$.
    4. Χρωματίζουμε τον u με $i+j$.
    5. $i \leftarrow i+j$.
    6. Αφαιρούμε από τον G τον u και τη γειτονιά του.
5. Χρωματίζουμε τους αχρωμάτιστους κόμβους του G χρησιμοποιώντας τον αλγόριθμο άπληστου χρωματισμού ξεκινώντας από το χρώμα i. Έστω ότι απαιτούνται s χρώματα.

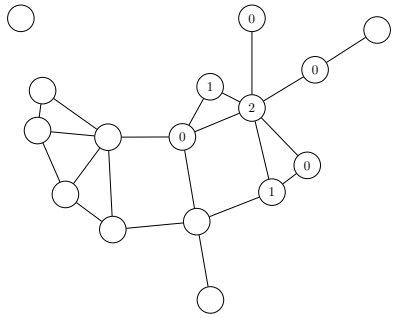
6. Επιστρέφουμε το συνολικό αριθμό $i + s$ των χρωμάτων που χρησιμοποιήσαμε.

Για να χρωματίσουμε ένα γράφο με δύο χρώματα, όπως θέλουμε να κάνουμε στο βήμα 2, αρκεί να κάνουμε μια κατά βάθος διάσχιση και να χρωματίζουμε κάθε κόμβο v με το άλλο χρώμα από αυτό που χρωματίσαμε τον γείτονά του u από τον οποίο φτάσαμε στον εν λόγω κόμβο ακολουθώντας το σύνδεσμο $u \to v$. Ταυτόχρονα θα πρέπει να ελέγχουμε ότι όλοι οι χρωματισμένοι γείτονες του v έχουν το χρώμα του u και όχι του v. Αν αυτό δεν συμβαίνει, τότε ο γράφος μας δεν μπορεί να χρωματιστεί με k χρώματα, δηλαδή δεν είναι k-χρωματικός, και γι' αυτό επιστέφουμε -1.

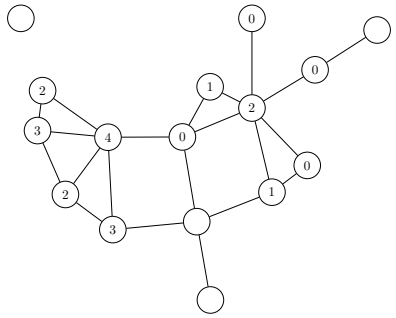
Ας χρησιμοποιήσουμε τον αλγόριθμο του Wigderson στον προηγούμενο γράφο, για ευκολία τον παραθέτουμε και εδώ:



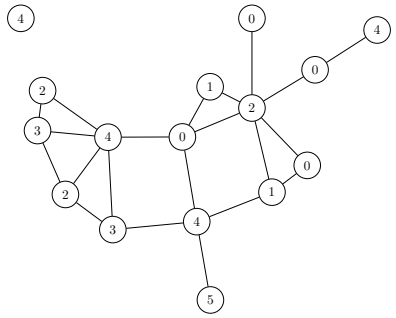
Ζητάμε από τον αλγόριθμο να χρωματίσει τον γράφο με τρία χρώματα. Ο χρωματισμός του γράφου θα ξεκινήσει από τον γράφο που επάγεται από τη γειτονιά του κόμβου 8. Θα χρωματίσει με τα δύο πρώτα χρώματα (0 και 1) τη γειτονιά και στη συνέχεια με το επόμενο χρώμα (2) τον ίδιο τον κόμβο 8.



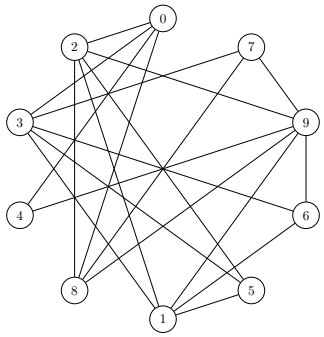
Στη συνέχεια ο αλγόριθμος θα χειριστεί το γράφο που επάγεται από τους κόμβους που δεν έχουν χρωματιστεί και θα αναλάβει τον κόμβο 11 και τη γειτονιά του. Θα χρωματίσει τη γειτονιά του με τα δύο επόμενα χρώματα ξεκινώντας από εκεί που είχαμε μείνει (άρα τα χρώματα 2 και 3) και στη συνέχεια τον ίδιο τον κόμβο 11 με το επόμενο χρώμα (4).



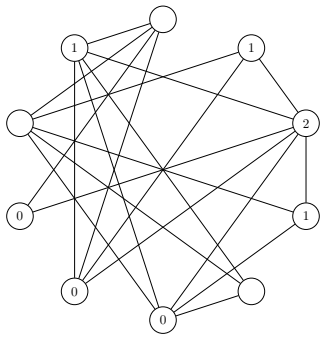
Απομένουν τέσσερεις κόμβοι οι οποίοι δεν έχουν χρωματιστεί και ο βαθμός των οποίων στο απομεινάρι του γράφου είναι μικρότερος από $\lceil f_k(n) \rceil = \lceil f_3(16) \rceil = \lceil 16^{1-1/(3-1)} \rceil = \lceil 16^{1/2} \rceil = \lceil \sqrt{16} \rceil = 4$. Οι κόμβοι αυτοί θα χρωματιστούν με τον αλγόριθμο άπληστου χρωματισμού ξεκινώντας από το χρώμα 4, οπότε τελικά θα χρειαστούν έξι χρώματα για τον πλήρη χρωματισμό όλου του γράφου.



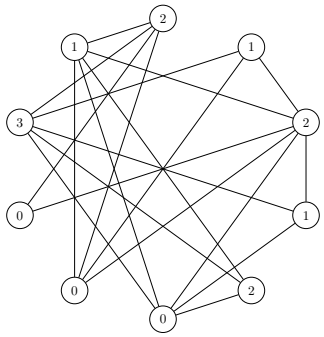
Στην περίπτωση αυτή ο αλγόριθμος του Wigderson χρησιμοποιεί περισσότερα χρώματα από τον αλγόριθμο του Johnson. Ας δούμε όμως τι γίνεται με το δεύτερο γράφο στον οποίο εφαρμόσαμε τον αλγόριθμο του Johnson, που για ευκολία τον παραθέτουμε πάλι εδώ:



Ζητάμε από τον αλγόριθμο να χρωματίσει τον γράφο με τρία χρώματα. Ο χρωματισμός θα ξεκινήσει από τον γράφο που επάγεται από τη γειτονιά του κόμβου 9, η οποία θα χρωματιστεί με τα δύο πρώτα χρώματα (0 και 1) ενώ ο 9 θα χρωματιστεί με το χρώμα 2.



Απομένουν τρεις κόμβοι που δεν έχουν χρωματιστεί και ο βαθμός των οποίων στο απομεινάρι του γράφου είναι μικρότερος από $\lceil f_k(n) \rceil = \lceil f_3(10) \rceil = \lceil 10^{1-1/(3-1)} \rceil = \lceil 10^{1/2} \rceil = \lceil \sqrt{10} \rceil = 4$. Οι κόμβοι αυτοί θα χρωματιστούν με τον αλγόριθμο άπληστου χρωματισμού ξεκινώντας από το χρώμα 2, οπότε τελικά θα χρειαστούν τέσσερα χρώματα για τον πλήρη χρωματισμό όλου του γράφου.



Απαιτήσεις Προγράμματος

Κάθε φοιτητής θα εργαστεί σε αποθετήριο στο GitHub. Για να αξιολογηθεί μια εργασία θα πρέπει να πληροί τις παρακάτω προϋποθέσεις:

- Για την υποβολή της εργασίας θα χρησιμοποιηθεί το ιδιωτικό αποθετήριο του φοιτητή που δημιουργήθηκε για τις ανάγκες του μαθήματος και του έχει αποδοθεί. Το αποθετήριο αυτό έχει όνομα του τύπου `username-algo-assignments`, όπου `username` είναι το όνομα του φοιτητή στο GitHub. Για παράδειγμα, το σχετικό αποθετήριο του διδάσκοντα θα ονομαζόταν `louridas-algo-assignments` και θα ήταν προσβάσιμο στο https://github.com/dmst-algorithms-course/louridas-algo-assignments. *Τυχόν άλλα αποθετήρια απλώς θα αγνοηθούν.*
- Μέσα στο αποθετήριο αυτό θα πρέπει να δημιουργηθεί ένας κατάλογος `assignment-2022-4`.
- Μέσα στον παραπάνω κατάλογο το πρόγραμμα θα πρέπει να αποθηκευτεί με το όνομα `graph_coloring.py`.
- Δεν επιτρέπεται η χρήση έτοιμων βιβλιοθηκών γράφων ή τυχόν έτοιμων

υλοποιήσεων των αλγορίθμων, ή τμημάτων αυτών, εκτός αν αναφέρεται ρητά ότι επιτρέπεται. Για την υλοποίηση του άπληστου αλγορίθμου μπορείτε να χρησιμοποιήσετε, αν σας βολεύει, το άρθρο της Wikipedia.

- Επιτρέπεται η χρήση δομών δεδομένων της Python όπως στοίβες, λεξικά, σύνολα, κ.λπ.
- Επιτρέπεται η χρήση των παρακάτω βιβλιοθηκών ή τμημάτων τους όπως ορίζεται:
  - `sys.argv`
  - `argparse`
  - `collections.defaultdict`
  - `math.ceil`
- Το πρόγραμμα θα πρέπει να είναι γραμμένο σε Python 3.

Το πρόγραμμα θα καλείται ως εξής (όπου `python` η κατάλληλη εντολή στο εκάστοτε σύστημα):

```
python graph_coloring.py {johnson,wigderson} num_colors start_color input_file
```

Η σημασία των παραμέτρων είναι η εξής:

- Αν δωθεί η παράμετρος `johnson` θα χρησιμοποιηθεί ο αλγόριθμος του Johnson.
- Αν δωθεί η παράμετρος `wigderson` θα χρησιμοποιηθεί ο αλγόριθμος του Wigderson.
- Η παράμετρος `num_colors` δίνει τον αριθμό χρωμάτων με τον οποίο ξέρουμε ή πιστεύουμε ότι μπορεί να χρωματιστεί ο γράφος.
- Η παράμετρος `start_color` δίνει τον αριθμό του χρώματος με το οποίο ξεκινάμε τον χρωματισμό (δεν είναι απαραίτητο ότι θα ξεκινάμε πάντα από το 0).
- Η παράμετρος `input_file` δίνει το όνομα του αρχείου που περιγράφει το γράφο. Το αρχείο περιέχει γραμμές οι οποίες περιέχουν μία ή δύο τιμές. Αν μια γραμμή περιέχει δύο τιμές, περιγράφει ένα σύνδεσμο μεταξύ αυτών των κόμβων. Αν μια γραμμή περιέχει μια τιμή, περιγράφει έναν κόμβο.

Το πρόγραμμα στην έξοδο θα εκτυπώνει το χρωματισμό του γράφου, δίνοντας τους κόμβους σε αύξουσα σειρά, και τον αριθμό χρωμάτων που χρησιμοποιήθηκαν.

Λεπτομέρειες Υλοποίησης

Στον άπληστο αλγόριθμο, η διάταξη των κόμβων θα είναι φθίνουσα ως προς τον βαθμό τους. Σε περίπτωση ισοβαθμίας, θα είναι αύξουσα ως προς τον αριθμό του κόμβου. Για παράδειγμα, αν οι κόμβοι 1 και 2 έχουν βαθμό 5, θα πάρουμε πρώτα τον 1 και μετά τον 2.

Στον αλγόριθμο του Johnson, στο βήμα 2, επιλέγουμε τον κόμβο με τον ελάχιστο βαθμό. Σε περίπτωση ισοβαθμίας, επιλέγουμε τον κόμβο με τον μικρότερο αριθμό.

Στον αλγόριθμο του Wigderson, στο βήμα 4.1, επιλέγουμε τον κόμβο με τον μεγαλύτερο βαθμό. Σε περίπτωση ισοβαθμίας, επιλέγουμε τον κόμβο με τον μικρότερο αριθμό.

Όταν διασχίζουμε έναν γράφο, θα επισκεπτόμαστε τους γείτονές του σε αύξουσα σειρά ως προς τον αριθμό τους.

Παραδείγματα

Παράδειγμα 1

Αν ο χρήστης του προγράμματος δώσει:

```
python graph_coloring.py johnson 3 0 graph_3.txt
```

τότε το πρόγραμμα θα διαβάσει τον γράφο από το αρχείο graph_3.txt. Ο γράφος αυτός είναι ο πρώτος 3-χρωματικός γράφος που εξετάσαμε. Το πρόγραμμα θα χρησιμοποιήσει τον αλγόριθμο του Johnson, θα ξεκινήσει το χρωματισμό από το χρώμα 0 και στην έξοδο θα εμφανίσει:

```
0: 0
1: 0
2: 1
3: 2
4: 0
5: 0
6: 1
7: 0
8: 2
9: 0
10: 1
11: 2
12: 0
13: 0
14: 1
15: 1
3
```

Η έξοδος έχει τους κόμβους σε αύξουσα σειρά με το χρώμα τους και στην τελευταία εγγραφή εμφανίζει τον αριθμό των χρωμάτων που απαιτήθηκαν.

Παράδειγμα 2

Αν ο χρήστης του προγράμματος δώσει:

```
python graph_coloring.py wigderson 3 0 graph_3.txt
```

τότε το πρόγραμμα θα διαβάσει τον γράφο από το αρχείο graph_3.txt, θα χρησιμοποιήσει τον αλγόριθμο του Wigderson, θα ξεκινήσει το χρωματισμό από το χρώμα 0 και στην έξοδο θα εμφανίσει:

```
0: 0
1: 2
2: 3
3: 4
4: 2
5: 0
6: 3
7: 5
8: 2
9: 4
10: 0
11: 4
12: 1
13: 4
14: 0
15: 1
6
```

Παράδειγμα 3

Αν ο χρήστης του προγράμματος δώσει:

```
python graph_coloring.py johnson 3 10 erdos_renyi_10.txt
```

τότε το πρόγραμμα θα διαβάσει τον γράφο από το αρχείο erdos_renyi_10.txt. Ο γράφος αυτός είναι ο γράφος Erdős-Renyi που είδαμε. Το πρόγραμμα θα χρησιμοποιήσει τον αλγόριθμο του Johnson ξεκινώντας από το χρώμα χρώμα 10, και στην έξοδο θα εμφανίσει:

```
0: 12
1: 12
2: 11
3: 11
4: 10
5: 10
6: 10
7: 10
8: 13
9: 14
5
```

Παράδειγμα 4

Αν ο χρήστης του προγράμματος δώσει:

```
python graph_coloring.py wigderson 3 10 erdos_renyi_10.txt
```

τότε το πρόγραμμα θα διαβάσει την ακολουθία από το αρχείο `erdos_renyi_10.txt`, θα χρησιμοποιήσει τον αλγόριθμο του Wigderson ξεκινώντας από το χρώμα 10, και στην έξοδο θα εμφανίσει:

```
0: 12
1: 10
2: 11
3: 13
4: 10
5: 12
6: 11
7: 11
8: 10
9: 12
4
```

Περισσότερες Πληροφορίες

Brélaz, Daniel. 1979. "New Methods to Color the Vertices of a Graph." *Communications of the ACM* 22 (4): 251–56. https://doi.org/10.1145/359094.359101.

Johnson, D S. 1974. "Worst Case Behavior of Graph Coloring Algorithms." In *Proceedings of the 5th South-Eastern Conference on Combinatorics, Graph Theory, and Computing*, 513–28. Winnipeg, Canada: Utilitas Mathematica Publishing.

Wigderson, Avi. 1982. "A New Approximate Graph Coloring Algorithm." In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, 325–29. STOC '82. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/800070.802207.

———. 1983. "Improving the Performance Guarantee for Approximate Graph Coloring." *Journal of the ACM* 30 (4): 729–35. https://doi.org/10.1145/2157.2158.

Καλή Επιτυχία!