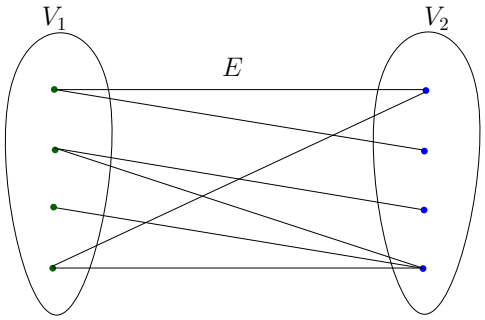
Άσκηση 1 [31.03.2022]

Παράδοση: Τετάρτη 04.05.2022, 11:00πμ

Η παρούσα άσκηση αφορά στον έλεγχο διμερότητας ενός γραφήματος. Σκοπός της άσκησης είναι η εξοικείωση με τη C++, με τους αλγόριθμους πιστοποίησης, με τη LEDA και την πειραματική αξιολόγηση.

Ένα μη κατευθυνόμενο γράφημα $G = (V, E)$ λέγεται *διμερές (bipartite)* όταν $V = V_1 \cup V_2$ και $V_1 \cap V_2 = \emptyset$ (δηλαδή όταν οι κορυφές του μπορούν να χωριστούν σε δύο ξένα σύνολα V_1, V_2), έτσι ώστε κάθε ακμή $e \in E$ να συνδέει μία κορυφή του V_1 με μία κορυφή του V_2. Στο Σχήμα 1 φαίνεται ένα παράδειγμα διμερούς γραφήματος.



Σχήμα 1: Ένα Διμερές Γράφημα

Στα πλαίσια της παρούσας άσκησης, καλείστε:

α) Να υλοποιήσετε έναν αλγόριθμο πιστοποίησης (certifying algorithm) και έναν ελεγκτή (checker) για τον έλεγχο διμερότητας ενός δεδομένου μη κατευθυνόμενου συνεκτικού γραφήματος $G = (V, E)$.

β) Να αξιολογήσετε πειραματικά τον αλγόριθμο πιστοποίησης σε διάφορες κατηγορίες γραφημάτων και να τον συγκρίνετε με τον αντίστοιχο της LEDA.

Πιο συγκεκριμένα, ζητούνται τα εξής.

Στο **ερώτημα α)** καλείστε να υλοποιήσετε δύο φάσεις. Η πρώτη φάση αφορά στην υλοποίηση ενός αλγορίθμου πιστοποίησης ελέγχου διμερότητας, ενώ η δεύτερη φάση αφορά στην υλοποίηση ενός ελεγκτή.

Στην πρώτη φάση, επιλέγεται τυχαία μία κορυφή $s \in V$ του συνεκτικού γραφήματος $G = (V, E)$ και ξεκινάει μία Αναζήτηση Πρώτα κατά Πλάτος (ΑΠΠ). Η ΑΠΠ με ρίζα την s κατατάσσει τις κορυφές σε επίπεδα με την s στο επίπεδο 0 (L_0), τις γειτονικές κορυφές της στο επίπεδο 1 (L_1), κλπ. Μετά την ΑΠΠ χρωματίζονται οι κορυφές ως εξής: η κορυφή s πράσινη, όλες οι κορυφές του επιπέδου L_1 μπλε, όλες οι κορυφές του επιπέδου L_2 πράσινες, κ.ο.κ., χρωματίζοντας μπλε τις κορυφές των επιπέδων με περιττή αρίθμηση και πράσινες εκείνες των επιπέδων με άρτια αρίθμηση, μέχρι να χρωματιστούν όλες. Αυτή η διαδικασία μπορεί να υλοποιηθεί τροποποιώντας την υλοποίηση ενός αλγορίθμου ΑΠΠ, και εμπλουτίζοντάς την με έναν πίνακα Color. Κάθε φορά που εκτελείται ένα βήμα στον αλγόριθμο ΑΠΠ, στο οποίο

αφαιρείται μία κορυφή v από την ουρά Q, πραγματοποιείτε την απόδοση τιμής $Color[v] =$ πράσινο ή $Color[v] =$ μπλε. Προς διευκόλυνση, δίνεται ο παρακάτω ψευδοκώδικας ενός αλγορίθμου ΑΠΠ.

```
1  procedure BFS(G,v):
2      create an empty queue Q
3      insert v onto Q
4      mark v
5      while Q is not empty:
6          t = Q.pop()
7          for all edges e = (t,u) ∈ E do
8              if u is not marked:
9                  mark u
10                 insert u onto Q
11     return none
```

Στον ψευδοκώδικα πρέπει να προστεθεί η χρήση του πίνακα color όπως περιγράφηκε παραπάνω. Ο ψευδοκώδικας πρέπει να «κωδικοποιηθεί» σε C++ με χρήση LEDA, υλοποιώντας την συνάρτηση

```
list<node> my_BFS(const graph& G, node s, node_array<int>& dist,
                  node_array<edge>& pred)
```

Τα ορίσματα της my_BFS είναι το δεδομένο G, μία αρχική κορυφή s, ένα node array dist το οποίο αποθηκεύει το επίπεδο στο οποίο βρίσκεται η κορυφή v, και ένα node array pred που για κάθε κορυφή v αποθηκεύει την ακμή που δείχνει στον προκάτοχο της v στο δέντρο που δημιουργείται από την ΑΠΠ. Τέλος, η συνάρτηση επιστρέφει μία λίστα από όλες τις κορυφές που επισκέφθηκε η ΑΠΠ.

Σην δεύτερη φάση, θα υλοποιήσετε μία συνάρτηση

```
bool my_bipar_checker(const graph& G, list<node>& V1, list<node>& V2)
```

η οποία έχει ορίσματα το δεδομένο γράφημα G και δύο node arrays $V1$,$V2$. Η συνάρτηση θα καλεί την my_BFS, και εν συνεχεία θα ελέγχει είτε αν προκύπτει ένας *έγκυρος* χρωματισμός του G με χρώματα πράσινο/μπλε στον οποίο κάθε ακμή έχει αντίθετα χρώματα, είτε αν υπάρχει κάποια ακμή με άκρα του ιδίου χρώματος που καθιστά το G μη διμερές. Η my_bipar_checker θα επιστρέφει false, αν το G δεν είναι διμερές εμφανίζοντας έναν κύκλο περιττού μεγέθους στο G, και true αν το G είναι διμερές, επιστρέφοντας τις κορυφές που ανήκουν στο σύνολο V_1 και εκείνες που ανήκουν στο σύνολο V_2, χρησιμοποιώντας τον χρωματισμό κορυφών της my_BFS.

Η υλοποίησή σας πρέπει να συγκριθεί με τον αντίστοιχο ελεγκτή της LEDA

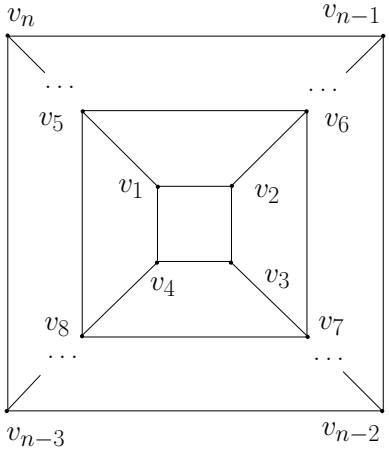
```
bool Is_Bipartite(const graph& G, list<node>& A, list<node>& B)
```

που παίρνει τα ίδια ορισματα και επιστρέφει true, αν το G είναι διμερές και false αν δεν είναι. Αν το G δεν είναι διμερές τότε επιστρέφεται ένας κύκλος περιττού μεγέθους στο G. Αν

το G είναι διμερές τότε επιστρέφονται και τα δύο σύνολα A και B. Προσοχή: η συνάρτηση Is_Bipartite της LEDA παίρνει ως όρισμα κατευθυνόμενα γραφήματα. Άρα όταν θα την χρησιμοποιήσετε θα πρέπει να μετατρέψετε τα γραφήματά σας. Η μετατροπή γίνεται ως εξής: για κάθε ακμή $e = (i, j)$ που δημιουργείτε για το (κατευθυνόμενο) γράφημά σας, θα δημιουργείτε και την αντίθετή της $e' = (j, i)$. Έτσι το προκύπτον κατευθυνόμενο γράφημα θα είναι ουσιαστικά μη κατευθυνόμενο.

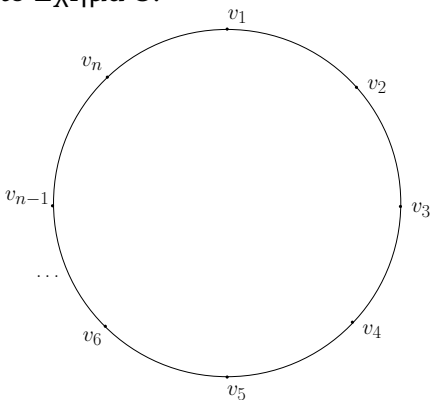
Για το **ερώτημα β)** θα γίνει συγκριτική πειραματική αξιολόγηση των συναρτήσεων Is_Bipartite (της LEDA) και της my_bipar_checker (που υλοποιήσατε) στις παρακάτω κατηγορίες γραφημάτων.

- Συνθετικά γραφήματα εμφωλιασμένων τετραγώνων n κορυφών και m ακμών, για τα οποία ισχύει $(n, m) \in \{(12000, 23996), (48000, 95996), (100000, 199996)\}$, και τα οποία κατασκευάζονται όπως φαίνεται στο Σχήμα 2.



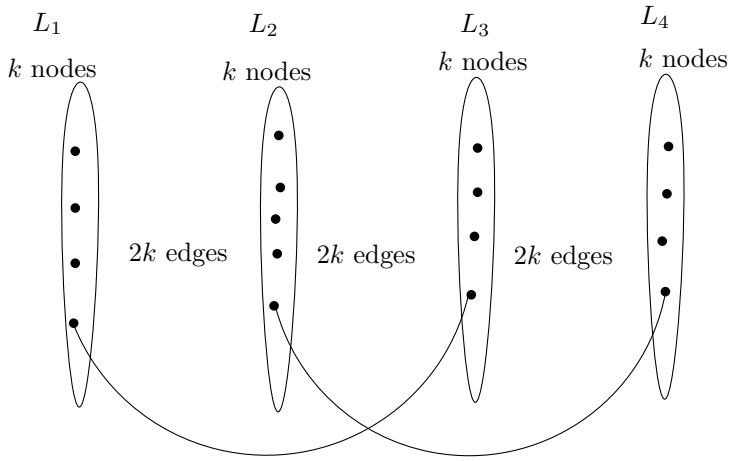
Σχήμα 2: Συνθετικό Γράφημα Εμφωλιασμένων Τετραγώνων

- Γραφήματα Δακτυλίου n κορυφών και $m = n$ ακμών, όπου n είναι περιττός και $(n, m) \in \{(20001, 20001), (500001, 50001), (100001, 100001)\}$, και τα οποία κατασκευάζονται όπως φαίνεται στο Σχήμα 3.



Σχήμα 3: Γράφημα Δακτυλίου

- Συνθετικά γραφήματα που αποτελούνται από 4 επίπεδα k κορυφών και $2k$ ακμών μεταξύ δύο διαδοχικών επιπέδων, για τα οποία ισχύει $k \in \{600, 1200, 1800\}$, και τα οποία κατασκευάζονται ως εξής (δείτε Σχήμα 4).



Σχήμα 4: Συνθετικό Γράφημα 4 επιπέδων

Έστω $v_1^i, v_2^i, \ldots, v_k^i$ οι κορυφές του επιπέδου L_i, $1 \leq i \leq 4$. Οι $2k$ ακμές μεταξύ δύο διαδοχικών επιπέδων L_i, L_{i+1}, $1 \leq i \leq 3$, δημιουργούνται ως εξής. Επιλέγονται πρώτα οι k ακμές (v_j^i, v_j^{i+1}), $1 \leq j \leq k$. Στη συνέχεια, επιλέγεται μια τυχαία κορυφή u του επιπέδου L_i και δημιουργούνται οι υπόλοιπες k ακμές (u, v_j^{i+1}), $1 \leq j \leq k$. Τέλος, μετά την κατασκευή των ακμών μεταξύ δύο διαδοχικών επιπέδων, δημιουργούνται τυχαία δύο επιπλέον ακμές: η πρώτη μεταξύ των επιπέδων L_1 και L_3, και η δεύτερη μεταξύ των επιπέδων L_2 και L_4. Η πρώτη (δεύτερη) ακμή δημιουργείται επιλέγοντας τυχαία μια κορυφή $u \in L_1$ ($u \in L_2$) και μια κορυφή $v \in L_3$ ($v \in L_4$), δημιουργώντας την ακμή (u, v).

Για τον ελεγκτή που υλοποιήσατε, δώστε την θεωρητική του πολυπλοκότητα, και μετρήστε τον πραγματικό χρόνο εκτέλεσης του, χρησιμοποιώντας τις κατάλληλες συναρτήσεις μέτρησης χρόνου.

Για κάθε στιγμιότυπο γραφημάτων (από κάθε οικογένεια) θα τρέχετε τους ελεγκτές my_bipar_checker και Is_Bipartite, και θα αναφέρετε τους χρόνους που χρειάστηκε ο κάθε αλγόριθμος για να επιστρέψει την απάντηση. Συγκρίνετε τους χρόνους, σχολιάστε και εξηγήστε τα αποτελέσματα και τυχόν διαφορές που παρατηρείτε στα πειράματα με τα παραπάνω γραφήματα.

Προσοχή: οι απαντήσεις διμερότητας των συναρτήσεων my_bipar_checker και Is_Bipartite πρέπει να ταυτίζονται!

Bonus: Θα δοθεί βαθμολογία bonus σε όποιον δημιουργήσει μια επιπλέον οικογένεια μη τετριμμένων συνθετικών γραφημάτων στην οποία η ιδιότητα της διμερότητας δεν είναι προφανές ότι ισχύει (όπως π.χ. εκείνης του Σχήματος 4) εκτελώντας πειράματα και για την οικογένεια

αυτή.

Υποβολή Προγραμματιστικής Άσκησης:

Η παράδοση της εργασίας θα πραγματοποιηθεί ηλεκτρονικά, μέσω της σελίδας του μαθήματος στο eclass, υποβάλοντας ένα συμπιεσμένο αρχείο που περιέχει όλα τα παραδοτέα της εργασίας:

1. **report.pdf**
   Ένα pdf αρχείο που περιέχει την αναφορά της εργασίας σας, στην οποία περιγράφονται οι βασικές αποφάσεις της υλοποίησής σας, τα δεδομένα δοκιμής που χρησιμοποιήσατε και η πειραματική αξιολόγηση.

2. **Makefile**
   Το αρχείο Makefile που χρησιμοποιήσατε.

3. **src/**
   Ένα φάκελο που θα περιέχει όλα τα αρχεία του πηγαίου κώδικά σας. Ο πηγαίος κώδικας που δίνετε για τις υλοποιήσεις και πειραματικές αξιολογήσεις πρέπει να είναι σωστά δομημένος, στοιχισμένος και σχολιασμένος. Ο κώδικάς σας πρέπει να εκτελείται στο σύστημα diogenis.

4. **README**
   Ένα αρχείο που θα περιέχει τα στοιχεία σας (Όνομα, Επώνυμο, ΑΜ, email)

Παρατήρηση 1: Μπορείτε να χρησιμοποιήσετε όποιους τύπους της LEDA κρίνετε απαραίτητο.

Παρατήρηση 2: Για περαιτέρω διευκρινήσεις σχετικά με την άσκηση, μπορείτε να επικοινωνήσετε με τους Νίκο Ζαχαράτο (zacharato@ceid.upatras.gr) και Παρασκευή Μαχαίρα (machaira@ceid.upatras.gr).