Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Θεμελιώδη Θέματα Επιστήμης Υπολογιστών, 2024-25

2η σειρά γραπτών ασκήσεων

(αλγοριθμικές τεχνικές – αλγόριθμοι γράφων – αριθμητικοί αλγόριθμοι)

Άσκηση 1. (Ελάχιστο πλήθος στάσεων)

Δίνεται κατευθυνόμενος γράφος με n κορυφές (οι πόλεις μιας χώρας) και m ακμές (δρόμους) που συνδέουν πόλεις μεταξύ τους. Δεν συνδέονται υποχρεωτικά όλες οι πόλεις με δρόμο. Μπορεί ανάμεσα σε δύο πόλεις να υπάρχουν δρόμοι και προς τις δύο κατευθύνσεις, αλλά δεν είναι απαραίτητο. Τα βάρη των ακμών αντιπροσωπεύουν τις αποστάσεις μεταξύ δύο πόλεων (αποστάσεις θετικές). Ένα αυτοκίνητο ξεκινά από την πόλη s και θέλει να φτάσει στην πόλη t. Το αυτοκίνητο έχει αυτονομία κίνησης, ως προς τη βενζίνη που μπορεί να αποθηκεύσει, k χιλιόμετρα. Σε κάθε πόλη υπάρχει η δυνατότητα ανεφοδιασμού (υπάρχει ακριβώς ένα βενζινάδικο). Δεδομένου ότι η καθυστέρηση για ανεφοδιασμό είναι μεγάλη επιθυμούμε να βρούμε το ελάχιστο πλήθος στάσεων για ανεφοδιασμό που χρειάζεται να κάνει το αυτοκίνητο προκειμένου να φτάσει στον προορισμό του, και την αντίστοιχη διαδρομή.

1. Δώστε όσο το δυνατόν πιο αποδοτικό αλγόριθμο για το πρόβλημα αυτό. Εξηγήστε την ορθότητα και υπολογίστε την πολυπλοκότητα του αλγορίθμου που προτείνετε.

2. Να σημειωθεί ότι δεν θέλουμε να βρούμε τη διαδρομή με την ελάχιστη συνολική απόσταση, αλλά αυτή *με τις λιγότερες δυνατές στάσεις*. Δώστε ένα παράδειγμα στο οποίο οι δύο αυτές διαδρομές να διαφέρουν.

3. Έστω ότι θέλουμε επιπλέον να βρούμε από τις διαδρομές με τις λιγότερες στάσεις αυτή που έχει ελάχιστη συνολική απόσταση. Δώστε όσο το δυνατόν πιο αποδοτικό αλγόριθμο για το πρόβλημα αυτό, εξηγήστε την ορθότητα και υπολογίστε την πολυπλοκότητά του.

Άσκηση 2. (Μείωση αποστάσεων)

Έστω γράφημα $G(V, E)$ που αναπαριστά το οδικό δίκτυο μιας χώρας με V πόλεις (οι κορυφές του γραφήματος) και E δρόμους (οι ακμές του γραφήματος). Κάθε δρόμος e_i έχει μήκος w_i. Θέλουμε να προστεθεί ένας ακόμη δρόμος $e^{'}$ μήκους $w^{'}$ μεταξύ δύο πόλεων. Τα υποψήφια ζεύγη πόλεων για την προσθήκη του νέου δρόμου περιλαμβάνονται σε ένα σύνολο $E^{'}$ που δίνεται. Ο δρόμος που θα προστεθεί πρέπει να επιτυγχάνει τη μέγιστη μείωση απόστασης μεταξύ δύο συγκεκριμένων πόλεων v_i και v_j. Περιγράψτε έναν αποδοτικό αλγόριθμο που θα καθορίζει τις πόλεις μεταξύ των οποίων θα προστεθεί ο νέος δρόμος.

Άσκηση 3. (2ο-ΕΣΔ) Θεωρούμε μη κατευθυνόμενο συνεκτικό γράφημα $G(V, E, w)$ με θετικά βάρη στις ακμές, και υποθέτουμε ότι $|E| \geq |V|$ και ότι όλα τα βάρη των ακμών του G είναι διαφορετικά μεταξύ τους. Έστω T το σύνολο όλων των συνδετικών δέντρων του G και έστω $t \in T$ ένα ελάχιστο συνδετικό δέντρο του G. Το Δεύτερο Ελαφρύτερο Συνδετικό Δέντρο (2ο-ΕΣΔ) του G είναι ένα συνδετικό δέντρο $t' \in T$ τέτοιο ώστε $w(t') = min_{t'' \in T \setminus \{t\}}\{w(t'')\}$. Με άλλα λόγια, το δεύτερο ελάχιστο συνδετικό δέντρο t' είναι ένα συνδετικό δέντρο του G που έχει βάρος μεγαλύτερο ή ίσο από το βάρος του ΕΣΔ t και μικρότερο ή ίσο από το βάρος κάθε άλλου συνδετικού δέντρου.

1. Να δείξετε ότι το ΕΣΔ του G είναι μοναδικό, και ότι κάτι τέτοιο δεν ισχύει απαραίτητα για το 2ο-ΕΣΔ του G.

2. Έστω t το ΕΣΔ του G και $t^{'}$ το 2ο-ΕΣΔ.Να δείξετε ότι τα t και $t^{'}$ διαφέρουν κατά μία μόνο ακμή, δηλαδή ότι υπάρχουν ακμές $e \in t$ και $e^{'} \notin t$ τέτοιες ώστε $t^{'} = t \cup \{e^{'}\} \setminus \{e\}$

3. Έστω t ένα συνδετικό δέντρο του G και, για οποιοδήποτε ζεύγος κορυφών $u, v \in V$, έστω e_{uv}^{max} η ακμή μεγίστου βάρους στο στο μοναδικό $u - v$ μονοπάτι στο t. Να διατυπώσετε αλγόριθμο χρόνου $O(|V|^2)$ που δέχεται ως είσοδο το t και προσδιορίζει την ακμή e_{uv}^{max} για όλα τα ζεύγη κορυφών $u, v \in V$.

4. Να διατυπώσετε αποδοτικό αλγόριθμο που να υπολογίζει το 2ο-ΕΣΔ του G. Εξηγήστε την ορθότητα και βρείτε την πολυπλοκότητα του αλγορίθμου που προτείνετε.

Άσκηση 4. (Κατάργηση κύκλων) Έστω μη κατευθυνόμενο γράφημα $G(V, E, w)$ με θετικά βάρη w στις ακμές του. Θέλουμε να αφαιρέσουμε ένα σύνολο ακμών, με το ελάχιστο δυνατό συνολικό βάρος, ώστε το γράφημα να μην έχει πλέον κύκλους.
Να προτείνετε αποδοτικό αλγόριθμο που να υπολογίζει τις ακμές που πρέπει να αφαιρεθούν. Εξηγήστε την ορθότητα και υπολογίστε την πολυπλοκότητα του αλγορίθμου σας.

Άσκηση 5. (Εύρεση ΜΚΔ) Θεωρήστε τον παρακάτω αλγόριθμο για εύρεση ΜΚΔ που είναι γνωστός ως Binary GCD.

bgcd(a, b): (* υποθέτουμε $a, b > 0$ *)
- Αν $a = b$ επίστρεψε a
- αν a, b άρτιοι επίστρεψε $2 \cdot$ bgcd$(a/2, b/2)$
- αν a είναι άρτιος και b περιττός επίστρεψε bgcd$(a/2, b)$, και αντίστοιχα αν b άρτιος και a περιττός
- αν a, b περιττοί επίστρεψε bgcd$(\min(a, b), |a - b|/2)$

(α) Αποδείξτε την ορθότητα του Binary GCD.
(β) Ποια είναι η πολυπλοκότητά του και γιατί;
(γ) Υλοποιήστε τον και συγκρίνετε την απόδοτικότητά του με αυτήν του Ευκλείδειου αλγόριθμου. Δοκιμάστε τους δύο αλγορίθμους με τουλάχιστον 10 ζεύγη πολύ μεγάλων αριθμών.

Άσκηση 6. (Επαναλαμβανόμενος τετραγωνισμός – έλεγχοι πρώτων αριθμών)
(α) Γράψτε πρόγραμμα σε γλώσσα της επιλογής σας (θα πρέπει να υποστηρίζει πράξεις με αριθμούς 100δων ψηφίων) που να ελέγχει αν ένας αριθμός είναι πρώτος με τον έλεγχο (test) του Fermat:
Αν n πρώτος τότε για κάθε a τ.ώ. $1 < a < n - 1$, ισχύει

$$a^{n-1} \bmod n = 1$$

Αν λοιπόν, για δεδομένο n βρεθεί a ώστε να μην ισχύει η παραπάνω ισότητα τότε ο αριθμός n είναι οπωσδήποτε σύνθετος. Αν η ισότητα ισχύει για το συγκεκριμένο a, τότε η δοκιμή πρέπει να επαναληφθεί με νέο a, καθώς υπάρχει περίπτωση ο αριθμός να είναι σύνθετος και παρ'όλα αυτά η ισότητα να ισχύει για κάποιες τιμές του a. Μια ενδιαφέρουσα ιδιότητα λέει ότι, αν ο n είναι σύνθετος, η πιθανότητα να ισχύει η ισότητα είναι $\leq 1/2$ (αυτό ισχύει για όλα τα n εκτός από κάποιες 'παθολογικές'

περιπτώσεις, που λέγονται αριθμοί Carmichael, δείτε ερώτημα (β) παρακάτω). Έτσι, μπορούμε να αυξήσουμε σημαντικά την πιθανότητα επιτυχίας (δηλ. της επιβεβαίωσης της συνθετότητας του αριθμού n) επαναλαμβάνοντας μερικές φορές τη δοκιμή (τυπικά 30 φορές) με διαφορετικό a. Αν όλες τις φορές βρεθεί να ισχύει η παραπάνω ισότητα τότε λέμε ότι το n "περνάει το test" και ανακηρύσσουμε το n πρώτο αριθμό· αν έστω και μία φορά αποτύχει ο έλεγχος, τότε είμαστε βέβαιοι ότι ο αριθμός είναι σύνθετος.

Το πρόγραμμά σας θα πρέπει να δουλεύει σωστά για αριθμούς χιλιάδων ψηφίων. Δοκιμάστε την με τους αριθμούς:

$$67280421310721, 170141183460469231731687303715884105721, 2^{2281} - 1$$

Σημείωση: το $a^{2^{2281}-2}$ έχει 'αστρονομικά' μεγάλο πλήθος ψηφίων (δεν χωράει να γραφτεί ούτε σε ολόκληρο το σύμπαν!), ενώ το $a^{2^{2281}-2} \bmod (2^{2281} - 1)$ είναι σχετικά "μικρό" (έχει μερικές χιλιάδες δεκαδικά ψηφία *μόνο* :-)) οπότε είναι δυνατόν να το υπολογίσουμε (με λίγη προσοχή).

(β) Υπάρχουν (λίγοι) σύνθετοι που έχουν την ιδιότητα να περνούν τον έλεγχο Fermat για κάθε a που είναι σχετικά πρώτο με το n, οπότε για αυτούς το test θα αποτύχει όσες δοκιμές και αν γίνουν (εκτός αν πετύχουμε κατά τύχη a που δεν είναι σχετικά πρώτο με το n, πράγμα αρκετά απίθανο για αρκετά μεγάλο n). Αυτοί οι αριθμοί λέγονται *Carmichael* – δείτε και `http://en.wikipedia.org/wiki/Carmichael_number`. Ελέγξτε τη συνάρτησή σας με *αρκετά μεγάλους* αριθμούς Carmichael που θα βρείτε π.χ. στη σελίδα
`http://de.wikibooks.org/wiki/Pseudoprimzahlen:_Tabelle_Carmichael-Zahlen`. Τι παρατηρείτε;

(γ) Μελετήστε και υλοποιήστε τον έλεγχο Miller-Rabin (π.χ. από τις σημειώσεις που θα βρείτε στη σελίδα του μαθήματος στο Helios) που αποτελεί βελτίωση του ελέγχου του Fermat και δίνει σωστή απάντηση με πιθανότητα τουλάχιστον 1/2 για *κάθε* φυσικό αριθμό (οπότε με 30 επαναλήψεις έχουμε αμελητέα πιθανότητα λάθους για κάθε αριθμό εισόδου). Δοκιμάστε τον με διάφορους αριθμούς Carmichael. Βλέπετε κάτι περίεργο; Πώς το εξηγείτε;

(γ) Γράψτε πρόγραμμα που να βρίσκει όλους τους πρώτους αριθμούς Mersenne, δηλαδή της μορφής $n = 2^x - 1$ με $1 < x < 200$ (σημειώστε ότι αν το x δεν είναι πρώτος, ούτε το $2^x - 1$ είναι πρώτος – μπορείτε να το αποδείξετε;). Αντιπαραβάλετε με όσα αναφέρονται στην ιστοσελίδα `https://www.mersenne.org/primes/`.

Άσκηση 7. (Αριθμοί Fibonacci)

(α) Υλοποιήστε και συγκρίνετε τους εξής αλγορίθμους για υπολογισμό του n-οστού αριθμού Fibonacci: αναδρομικό με memoization, επαναληπτικό, και τον αλγόριθμο με χρήση πινάκων 2×2.

Υλοποιήστε τους αλγορίθμους σε γλώσσα που να υποστηρίζει πολύ μεγάλους ακεραίους (100δων ψηφίων), π.χ. σε Python. Χρησιμοποιήστε τον πολλαπλασιασμό ακεραίων που παρέχει η γλώσσα. Τι συμπεραίνετε;

(β) Δοκιμάστε να λύσετε το παραπάνω πρόβλημα με ύψωση σε δύναμη, χρησιμοποιώντας τη σχέση του F_n με το ϕ (χρυσή τομή). Τι παρατηρείτε;

(γ) Υλοποιήστε συνάρτηση, όσο το δυνατόν πιο αποδοτική, που να δέχεται σαν είσοδο δύο θετικούς ακεραίους n, k και να υπολογίζει τα k λιγότερο σημαντικά ψηφία του n-οστού αριθμού Fibonacci.

Θεωρώντας $n = 10^i, k = 17$ βρείτε το μεγαλύτερο i για το οποίο η συνάρτησή σας δίνει σωστό αποτέλεσμα εντός χρόνου 1 sec.

(δ) Αναζητήστε και εξετάστε τη μέθοδο Fast Doubling για να επιλύσετε το ερώτημα (γ). Συγκρίνετέ την με τη μέθοδο χρήσης πινάκων 2×2 θεωρητικά και υπολογιστικά.

Άσκηση 8. (Αλγόριθμοι Διαίρει και Βασίλευε)
[άσκηση από το βιβλίο *Algorithms*, των Dasgupta, Papadimitriou, Vazirani]

Μία ακολουθία στοιχείων $A[1], \ldots, A[n]$ λέμε ότι έχει ένα πλειοψηφούν στοιχείο (majority element) αν περισσότερα από τα μισά στοιχεία της ακολουθίας είναι ίδια. Με δεδομένη μια ακολουθία, ο σκοπός είναι να σχεδιαστεί ένας αποδοτικός αλγόριθμος ο οποίος να προσδιορίζει αν η ακολουθία έχει πλειοψηφούν στοιχείο, και αν ναι να βρίσκει αυτό το στοιχείο. Τα στοιχεία της ακολουθίας δεν προέρχονται υποχρεωτικά από κάποιο διατεταγμένο πεδίο τιμών όπως οι ακέραιοι, και έτσι δεν μπορούν να υπάρχουν συγκρίσεις της μορφής "ισχύει $A[i] > A[j]$;". (Για παράδειγμα μπορείτε να θεωρήσετε τα στοιχεία της ακολουθίας είναι αρχεία GIF). Ωστόσο μπορείτε να απαντάτε σε σταθερό χρόνο σε ερωτήσεις της μορφής: "ισχύει $A[i] = A[j]$;".

1. Δείξτε πως θα λύσετε αυτό το πρόβλημα σε χρόνο $O(n \log n)$. (Υπόδειξη: Χωρίστε τη ακολουθία A σε δύο ακολουθίες A_1 και A_2 με το μισό μέγεθος. Η γνώση των πλειοψηφούντων στοιχείων των A_1 και A_2 σας βοηθά να βρείτε το πλειοψηφούν στοιχείο του A; (Αν ναι, μπορείτε να χρησιμοποιήσετε μία προσέγγιση 'διαίρει και βασίλευε'.)

2. Μπορείτε να βρείτε έναν αλγόριθμο γραμμικού χρόνου;

   (Υπόδειξη: σκεφτείτε την παρακάτω προσέγγιση 'διαίρει και βασίλευε':

   - Συνδυάστε τα στοιχεία του A με αυθαίρετο τρόπο, για να δημιουργήσετε $n/2$ ζεύγη.
   - Εξετάστε κάθε ζεύγος: αν τα δύο στοιχεία είναι διαφορετικά, απορρίψτε τα και τα δύο. Αν είναι ίδια, διατηρήστε μόνο το ένα από αυτά.

     Δείξτε ότι μετά από αυτή τη διαδικασία έχουν απομείνει το πολύ $n/2$ στοιχεία, και ότι έχουν ένα πλειοψηφούν στοιχείο σε περίπτωση που η ακολουθία A έχει τέτοιο στοιχείο.)

Προθεσμία υποβολής και οδηγίες. Οι απαντήσεις θα πρέπει να υποβληθούν έως τις 8/12/2024, και ώρα 23:59, σε ηλεκτρονική μορφή, στο Helios (προσπαθήστε το τελικό αρχείο να είναι μεγέθους <5MB συνολικά). Αποδεκτά format: pdf, png, jpg, gif, και zip ή gz που να περιέχει κάποια από τα προηγούμενα.

Δεν θα διορθωθούν εργασίες που στέλνονται με email.

Συνιστάται *θερμά* να αφιερώσετε ικανό χρόνο για να λύσετε τις ασκήσεις μόνοι σας προτού καταφύγετε σε οποιαδήποτε *θεμιτή* βοήθεια (διαδίκτυο, βιβλιογραφία, συζήτηση με συμφοιτητές). Σε κάθε περίπτωση, οι απαντήσεις θα πρέπει να είναι *αυστηρά* ατομικές και να περιλαμβάνουν αναφορές σε κάθε πηγή που χρησιμοποιήσατε.

Για να βαθμολογηθείτε θα πρέπει να παρουσιάσετε σύντομα τις λύσεις σας σε ημέρα και ώρα που θα ανακοινωθεί αργότερα. Για απορίες / διευκρινίσεις: στείλτε μήνυμα στη διεύθυνση `focs@corelab.ntua.gr`.