ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΛΓΟΡΙΘΜΟΙ ΚΑΙ ΠΟΛΥΠΛΟΚΟΤΗΤΑ

3η ΣΕΙΡΑ ΓΡΑΠΤΩΝ ΑΣΚΗΣΕΩΝ

Διδάσκοντες:

Άρ. Παγουρτζής

Δ. Φωτάκης

Δ. Σούλιου

Π. Γροντάς

Ειρήνη Δόντη

ΑΜ 03119839

7ο εξάμηνο

Αθήνα 2023

Άσκηση 1: Υπολογισιμότητα

(a)

Ένας αλγόριθμος που ημιποφασίζει το πρόβλημα των Διοφαντικών Εξισώσεων είναι ο αλγόριθμος απλοποίησης από τη θεωρία των αριθμών, ο οποίος βασίζεται στη χρήση συναρτήσεων απλοποίησης για να απλοποιήσει την εξίσωση και να δείξει ότι η εξίσωση έχει ακέραιες λύσεις αν και μόνο αν η απλοποιημένη εξίσωση έχει ακέραιες λύσεις (δε δίνει συγκεκριμένη λύση).

(b)

Θεωρούμε δεδομένη μία πολυωνυμική εξίσωση με ακέραιους συντελεστές και πεπερασμένο αριθμό αγνώστων και τη μετατρέπουμε σε Διοφαντικό σύστημα εξισώσεων. Χρησιμοποιούμε μία διαδικασία απόφασης για την αριθμητική Presburger όπως ο αλγόριθμος εξάλειψης ποσοτικοποιητή, για να αποφασίσουμε αν το σύστημα έχει ακέραιες λύσεις. Η διαδικασία απόφασης για την αριθμητική Presburger μπορεί να αναχθεί στο πρόβλημα τερματισμού, κωδικοποιώντας την είσοδο σε μηχανή Turing και ελέγχοντας αν αυτή η μηχανή σταματά σε συγκεκριμένη είσοδο. Επομένως, αν συμβεί μία λύση για το πρόβλημα Τερματισμού, μπορεί να λυθεί το πρόβλημα Διοφαντικών Εξισώσεων σε πολυωνυμικό χρόνο.

Άσκηση 2: Πολυπλοκότητα - Αναγωγές

(a)

Εάν ένα πρόβλημα είναι C-πλήρες σε μία σχέση με μία αναγωγή $\leq_R$, τότε το συμπληρωματικό πρόβλημα Π’ είναι επίσης C-πλήρες ως προς το $\leq_R$. Οπότε, το Π’ είναι τουλάχιστον τόσο δύσκολο όσο οποιοδήποτε άλλο πρόβλημα στην κατηγορία C και μπορεί να αναχθεί σε Π’ σε πολυωνυμικό χρόνο χρησιμοποιώντας την αναγωγή $\leq_R$ Εφόσον Π’και Π είναι C-πλήρη, τότε το Π’ ανήκει στο coC-πλήρες ως προς την αναγωγή.

(b)

Το πρόβλημα της ταυτολογίας είναι coNP επειδή μπορεί να επαληθευτεί σε πολυωνυμικό χρόνο ότι ένας προτασιακός τύπος δεν είναι ταυτολογία. Επίσης, είναι coNP-πλήρες, επειδή είναι τουλάχιστον τόσο δύσκολο όσο οποιοδήποτε άλλο πρόβλημα στο coNP, δηλαδή οποιοδήποτε πρόβλημα στο coNP μπορεί να αναχθεί στο πρόβλημα της Ταυτολογίας σε πολυωνυμικό χρόνο. Αυτό συμβαίνει, γιατί μπορούμε να κατασκευάσουμε έναν προτασιακό τύπο που αντιπροσωπεύει το συμπλήρωμα του προβλήματος και στη συνέχεια να ελέγξουμε αν είναι ταυτολογία για οποιοδήποτε πρόβλημα coNP. Αν δεν είναι, τότε το αρχικό πρόβλημα έχει λύση και αν είναι ταυτολογία, τότε το αρχικό πρόβλημα δεν έχει λύση.

(c)

Αν ένα πρόβλημα στο coNP είναι NP-complete, τότε είναι και NP-hard. Εφόσον το πρόβλημα είναι NP, τότε μπορεί να μειωθεί σε coNP-complete και συνεπώς κάθε πρόβλημα στο NP είναι στο coNP. Επιπλέον, αν ένα coNP-complete είναι στο coNP, κάθε πρόβλημα στο coNP είναι στο NP. Επειδή κάθε NP πρόβλημα είναι coNP και κάθε coNP είναι NP, τότε coNP = NP.

(d)

Για να αποδείξουμε ότι ένα NAE3SAT είναι NP-complete, μπορούμε να χρησιμοποιήσουμε την αναγωγή. Αρχικά, η NAE3SAT ανήκει στην κλάση NP, όπως και η SAT. Θα αποδείξουμε ότι η NAE3SAT μπορεί να αναχθεί σε 3-SAT σε πολυωνυμικό χρόνο και εφόσον η 3-SAT είναι NP-complete, τότε και η NAE3SAT είναι NP-complete. Δημιουργούμε μία νέα μεταβλητή για κάθε literal στον αρχικό τύπο. Για κάθε πρόταση:

Αν το πρώτο literal είναι αληθές, τότε το δεύτερο και το τρίτο πρέπει να είναι ψευδές. Αν το τρίτο literal είναι αληθές, τότε το πρώτο και το δεύτερο πρέπει να είναι ψευδές. Επίσης, αν το δεύτερο literal είναι αληθές, τότε το πρώτο και το τρίτο πρέπει να είναι ψευδές. Οι νέες μεταβλητές και οι περιορισμοί μπορούν να προστεθούν στον αρχικό τύπο σε πολυωνυμικό χρόνο. Οπότε, δημιουργείται ένας τύπος που έχει ίδια λύση με τον αρχικό τύπο, αφού όλοι οι περιορισμοί είναι ισοδύναμοι με την αρχική συνθήκη. Ο νέος τύπος είναι σε 3-CNF και όλοι οι περιορισμοί έχουν τη μορφή του 3-SAT, εφόσον ισχύει ότι ένα ακριβώς από τα τρία literals πρέπει να είναι αληθές. Οπότε, το πρόβλημα NAE3SAT μπορεί να μειωθεί σε 3-SAT σε πολυωνυμικό χρόνο και συνεπώς είναι NP-complete.

(e)

Μπορούμε να επιβεβαιώσουμε ότι το πρόβλημα Επιλογής Αντιπροσώπων είναι NP-complete μέσω της αντιστοιχίας σε Set Cover πρόβλημα. Για το Set Cover Θεωρούμε σύνολο S, υποσύνολα x1,…,xm του S και 1<k<m. Υπάρχουν το πολύ k υποσύνολα που η ένωση τους είναι το S. Παρατηρούμε ότι το πρόβλημα Επιλογής Αντιπροσώπων είναι ουσιαστικά ένα Set Cover πρόβλημα για το οποίο γνωρίζουμε ότι είναι NP-πλήρες. Οπότε και το πρόβλημα Επιλογής Αντιπροσώπων είναι NP-πλήρες.

<u>Άσκηση 3: Προσεγγιστικοί Αλγόριθμοι – Vertex Cover</u>

Ο αλγόριθμος είναι ένας αλγόριθμος 2 προσεγγίσεων για το πρόβλημα του Vertex Cover. Συγκεκριμένα, **το** σύνολο **C** που επιστρέφεται από τον αλγόριθμο **καλύπτει όλες τις ακμές** στο γράφημα G. Αυτό συμβαίνει, καθώς ο αλγόριθμος επιλέγει επανειλημμένα μία ακάλυπτη ακμή και προσθέτει τουλάχιστον ένα από τα τελικά σημεία του στο C μέχρι να καλυφθούν όλες οι ακμές. **Το συνολικό βάρος του συνόλου C που επιστρέφεται δεν είναι περισσότερο από το διπλάσιο του βάρους του βέλτιστου καλύμματος κορυφής**, καθώς ο αλγόριθμος εκχωρεί ένα βάρος c(e) σε κάθε ακμή e, το οποίο είναι ίσο με το ελάχιστο των βαρών των τελικών σημείων της ακμής και το συνολικό βάρος του C είναι το άθροισμα αυτών των βαρών για όλες τις ακμές. Εφόσον σε κάθε ακμή εκχωρείται ένα βάρος που δεν υπερβαίνει το μισό βάρος των τελικών σημείων του, το συνολικό βάρος του C δεν είναι μεγαλύτερο από το διπλάσιο του βάρους βέλτιστου καλύμματος κορυφής. **Το συνολικό βάρος του καλύμματος βέλτιστης κορυφής δεν είναι μικρότερο από το συνολικό βάρος του C**, επειδή ο αλγόριθμος μειώνει μόνο τα βάρη των κορυφών. Οπότε, το βάρος του C είναι το χαμηλότερο όριο στο βάρος του βέλτιστου καλύμματος κορυφής.

Ένα παράδειγμα γράφου στο οποίο ο αλγόριθμος υπολογίζει ένα Vertex Cover με συνολικό βάρος διπλάσιο από αυτό της βέλτιστης λύσης είναι ένα γράφημα με δύο κορυφές και μία ακμή. Η μία κορυφή πρέπει να έχει βάρος 1 και η άλλη να έχει βάρος 2. Οπότε, ο αλγόριθμος θα επέλεγε την κορυφή με βάρος 1 και θα επέστρεφε ένα κάλυμμα κορυφής βάρους 1, ενώ το βέλτιστο κάλυμμα θα ήταν η κορυφή με βάρος δύο.

Άσκηση 4: Πιθανοτικοί Αλγόριθμοι – Έλεγχος Ταξινόμησης

(α)

Επιλέγονται τυχαία k θέσεις στον πίνακα και ελέγχουμε αν τα στοιχεία σε αυτές τις θέσεις είναι στη σωστή σειρά. Ο αλγόριθμος μπορεί να αποφανθεί στην περίπτωση πινάκων που δεν είναι σχεδόν ταξινομημένοι, αλλά έχουν μικρό αριθμό τυχαία τοποθετημένων στοιχείων που ικανοποιούν το κριτήριο.

Για παράδειγμα, θεωρούμε τον πίνακα:

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 294, 295, 296, 297, 298, 299, 300].

Στην περίπτωση αυτή υπάρχουν n/4 στοιχεία (75) τα οποία έχουν τοποθετηθεί τυχαία, αλλά τα υπόλοιπα είναι ταξινομημένα. Ο αλγόριθμος ελέγχει Ω(n) θέσεις ώστε η πιθανότητα λάθους να γίνει μικρότερη από 10%.

(β)

Η δυαδική αναζήτηση χωρίζει επανειλημμένα το διάστημα αναζήτησης στη μέση, απορρίπτοντας το μισό μέρος που δεν περιέχει το στοιχείο στόχο. Όταν καλείται η BINARY-SEARCH(A, x, 1, n) και επιστρέφει τη θέση k, τότε το διάστημα αναζήτησης έχει περιοριστεί στο διάστημα [k,k] και το x βρίσκεται στη θέση k. Επίσης, όταν καλείται η BINARY-SEARCH(A, y, 1, n) και επιστρέφει τη θέση ℓ, τότε το διάστημα αναζήτησης έχει περιοριστεί στο διάστημα [ℓ, ℓ] και το y βρίσκεται στη θέση ℓ. Εφόσον k < ℓ, σημαίνει ότι το x βρέθηκε πριν τη διαδικασία αναζήτησης, κάτι που μπορεί να συμβεί μόνο αν x < y. Οπότε, αν k < ℓ, τότε x < y.

(γ)

Πρέπει να αποδείξουμε ότι αν ένας πίνακας A δεν είναι σχεδόν ταξινομημένος, τότε με μεγάλη πιθανότητα, σε τουλάχιστον μία από τις τυχαία επιλεγμένες θέσεις a_i, δε θα ικανοποιεί τη συνθήκη ai = BINARY-SEARCH(A, A[a_i], 1, n). Για να αποδειχθεί το ζητούμενο, χρησιμοποιούμε το γεγονός ότι αν το A δεν είναι σχεδόν ταξινομημένος, τότε υπάρχουν τουλάχιστον n/4 στοιχεία που πρέπει να αφαιρεθούν για να ταξινομηθεί το A. Αν επιλέξουμε τυχαία k θέσεις a1, a2, …, ak τότε η πιθανότητα να αντιστοιχούν όλα τα στοιχεία που πρέπει να αφαιρεθούν για να ταξινομηθεί το A είναι $(n/4)^k$. Η παραπάνω πιθανότητα γίνεται 10% για k=4.