Σπάρτη 2013

Γιαννούλα Αντωνοπούλου

Α.Μ.: 2007028

Φοιτήτρια του τμήματος
Τεχνολογίας Πληροφορικής και
Τηλεπικοινωνιών

ΑΤΕΙ ΚΑΛΑΜΑΤΑΣ – ΠΑΡΑΡΤΗΜΑ ΣΠΑΡΤΗΣ

ΤΜΗΜΑ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«ΤΟ ΠΡΟΒΛΗΜΑ ΤΟΥ ΠΛΑΝΟΔΙΟΥ ΠΩΛΗΤΗ»

Επιβλέπων : Καραγιώργος Γρηγόριος

Ευχαριστίες

Θεωρώ υποχρέωσή μου να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Γρηγόριο Καραγιώργο, για την πολύτιμη καθοδήγησή του.

Επίσης, οφείλω να αφιερώσω την πτυχιακή μου εργασία στους γονείς μου που μου συμπαραστάθηκαν σε όλα τα χρόνια της φοίτησής μου στο Α.Τ.Ε.Ι. Καλαμάτας – Παράρτημα Σπάρτης.

Περίληψη

Στην παρούσα πτυχιακή εργασία θα ασχοληθούμε με το πρόβλημα του πλανόδιου πωλητή. Για τη μελέτη του προβλήματος του πλανόδιου πωλητή μας χρειάζονται μερικοί αλγόριθμοι γράφων και κάποιες έννοιες για προβλήματα βελτιστοποίησης.

Η σχέση βέλτιστου αποτελέσματος – χρόνου έχει απασχολήσει και συνεχίζει να απασχολεί τους σχεδιαστές ενσωματωμένων κυκλωμάτων και όλους όσους ασχολούνται με την επιστήμη υπολογιστών αλλά και για άλλους τομείς όπως η βιολογία, η κοινωνιολογία κ.α.

Νέες τεχνολογίες αναπτύσσονται συνέχεια με σκοπό να βελτιστοποιήσουν τα προγράμματα και να τα κάνουν να τρέχουν σε όσο το δυνατό λιγότερο χρόνο.

Πολλά είναι τα προβλήματα στα οποία η σχέση βέλτιστου αποτελέσματος – χρόνου είναι αμφιλεγόμενη και ένα από αυτά είναι το πρόβλημα του πλανόδιου πωλητή. Στην παρούσα εργασία ορίσουμε τον αλγόριθμο του πλανόδιου πωλητή και θα προσπαθήσουμε να το αναλύσουμε είτε προσεγγιστικά είτε βελτιωτικά.

Περιεχόμενα

Περιεχόμενα Σχημάτων

Κεφάλαιο 1. Αλγόριθμοι

1.1 Εισαγωγή.

Αλγόριθμος ορίζεται μια πεπερασμένη σειρά ενεργειών, ή ένα σύνολο οδηγιών αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος. Πιο απλά ***αλγόριθμο*** ονομάζουμε μία σειρά από εντολές που έχουν αρχή και τέλος, είναι σαφείς και εκτελέσιμες που σκοπό έχουν την επίλυση κάποιου προβλήματος.

Πιο αναλυτικά ένας αλγόριθμος πρέπει να πληροί τα ακόλουθα κριτήρια:

1) ***Είσοδος***: Για να λειτουργήσει πρέπει να εισαχθούν Ν δεδομένα από κάποια εξωτερική πηγή.
2) ***Έξοδος***: Με το τέλος του αλγόριθμου πρέπει να παράγεται τουλάχιστον ένα αντικείμενο σαν αποτέλεσμα.
3) ***Καλά ορισμένος***: Η κάθε οδηγία πρέπει να είναι απόλυτα καθορισμένη, κατανοητή και να μην αφήνει κανένα περιθώριο αμφισβήτησης (πχ σε κάποια απόφαση).
4) ***Πεπερασμένος***: Να τελειώνει σε πεπερασμένο αριθμό βημάτων.
5) ***Αποτελεσματικός***: Κάθε οδηγία του αλγόριθμου πρέπει να είναι απόλυτα βασική και να μπορεί να εκτελεσθεί από μία υπολογιστική μηχανή. Δηλαδή το τρίτο κριτήριο δεν είναι αρκετό: πρέπει η οδηγία επιπλέον να είναι έτσι ώστε να φέρνει κάποιο αποτέλεσμα.
6) ***Γενικός***: Εάν είναι δυνατόν, ο αλγόριθμος πρέπει να λύνει πολλά στιγμιότυπα και όχι ένα και μοναδικό πρόβλημα. Για παράδειγμα, καλός είναι ο αλγόριθμος που επιλύει την εξίσωση πρώτου βαθμού $ax + b = 0$ και όχι μία συγκεκριμένη εξίσωση σαν την $5x + 6 = 0$.

Τέλος, ένας αλγόριθμος πρέπει να περιγράφεται αναλυτικά και με τέτοιο τρόπο ώστε να είναι απόλυτα κατανοητός ακόμη και σε κάποιον που δεν ξέρει το πρόβλημα που επιλύει.

Η λέξη αλγόριθμος προέρχεται από μία μελέτη του Πέρση μαθηματικού του 8ου αιώνα μ.Χ. Αλ Χουαρίζμι (Abu Ja'far Mohammed ibn Musa Al-Khowarismi), η οποία περιείχε συστηματικές τυποποιημένες λύσεις αλγεβρικών προβλημάτων και αποτελεί ίσως την πρώτη πλήρη πραγματεία άλγεβρας. Πέντε αιώνες αργότερα η μελέτη μεταφράστηκε στα Λατινικά και άρχιζε με τη φράση "Algorithmi dixit ..." (ο Αλγόριθμος λέει ...). Έτσι η λέξη αλγόριθμος καθιερώθηκε αργά τα επόμενα χίλια χρόνια με την έννοια «συστηματική διαδικασία αριθμητικών χειρισμών». Τη σημερινή της σημασία την οφείλει στη γρήγορη ανάπτυξη των ηλεκτρονικών υπολογιστών στα μέσα του 20ου αιώνα.

Το πιο γνωστό παράδειγμα για να γίνει η έννοια του αλγορίθμου εύκολα αντιληπτή είναι το εξής: Αν κάποιος επιθυμεί να γευματίσει θα πρέπει να εκτελέσει κάποια συγκεκριμένα βήματα: να συγκεντρώσει τα υλικά, να προετοιμάσει τα σκεύη μαγειρικής, να παρασκευάσει το φαγητό, να στρώσει το τραπέζι, να ετοιμάσει τη σαλάτα, να γευματίσει, να καθαρίσει το τραπέζι και να πλύνει τα πιάτα. Προφανώς, η προηγούμενη αλληλουχία οδηγεί στο επιθυμητό αποτέλεσμα. Δεν είναι όμως η μοναδική για την επίτευξη του σκοπού, αφού μπορεί να αλλάξει η σειρά των βημάτων (π.χ. πρώτα να ετοιμάσει τη σαλάτα και μετά να στρώσει το τραπέζι). Ωστόσο το νόημα είναι πως η κατάτμηση μιας σύνθετης εργασίας σε διακριτά βήματα που εκτελούνται διαδοχικά, είναι ο πιο πρακτικός τρόπος επίλυσης πολλών προβλημάτων.

Ανάλογα με την τεχνική επίλυσης ενός προβλήματος οι αλγόριθμοι διακρίνονται σε:[1]

- ***Αναδρομικοί (recursive)***: αλγόριθμοι που χρησιμοποιούν αναδρομικές λύσεις προβλημάτων, πχ πολυώνυμα Hermite, υπολογισμός παραγοντικού, κ.α.
- ***Διαίρει και Βασίλευε (divide and conquer)***: επιλύουν το πρόβλημα αναγάγοντάς το σε μικρότερα ανάλογα προβλήματα, πχ quick sort, merge sort, κα.
- ***Άπληστοι (greedy)***: επιλύουν προβλήματα επιλέγοντας κάθε φορά την τοπικά βέλτιστη λύση προσδοκώντας την συνολικά βέλτιστη, πχ πρόβλημα επιστροφής ρέστων, χρονικός προγραμματισμός, κ.α.

- ***Δυναμικού Προγραμματισμού (dynamic programming)***: συνήθως αναδρομικοί αλγόριθμοι οι οποίοι χρησιμοποιούν τις ενδιάμεσα παραγόμενες λύσεις, πχ πολλαπλασιασμός αλυσίδας πινάκων, κα.
- ***Παράλληλοι (parallel)***: εύρεση λύσης όπου το πρόβλημα δεν λύνεται σειριακά αλλά πολλές σχέσεις του εκτελούνται παράλληλα, πχ πολλά προβλήματα πινάκων, τεχνικές τύπου «διαίρει και βασίλευε», κ.α.

Επίσης, ανάλογα με την λύση που επιτυγχάνουν μπορούν να διακριθούν σε:

- ***Βέλτιστοι ή Άριστοι (optimal)***: εύρεση της βέλτιστης λύσης του προβλήματος, πχ επίλυσης μίας εξίσωσης δευτέρου βαθμού.
- ***Προσεγγιστικοί ή ευρεστικοί (approximation – heuristics)***: εύρεση «καλών» λύσεων σε άλυτα ή πολύ δύσκολα προβλήματα, πχ χρονοπρογραμματισμός εργασιών, χρωματισμός χάρτη κ.α.

1.2 Γράφοι

1.2.1 Ορισμοί

Πρώτος που χρησιμοποίησε σε θεωρητικό επίπεδο γράφους ήταν ο Euler το 1736 στην δημοσίευση με τίτλο "Solutio Problematis ad Geometrian Situs Pertinentis". Προσπάθησε, στην αρχή από απλή μαθηματική περιέργεια, να λύσει το πρόβλημα: «Πως μπορεί κάποιος να διασχίσει τις επτά γέφυρες του ποταμού Konigsberg σε μία βόλτα αλλά να περάσει μόνο και μία μόνο φορά από την καθεμία» [6].

Γράφος G είναι μία δομή (ένα σύνολο) που ορίζεται ως G = (V, E), όπου V είναι ένα πεπερασμένο σύνολο κορυφών (κόμβων) με δείκτες φυσικούς αριθμούς από το 1 μέχρι το n, E είναι ένα πεπερασμένο σύνολο ζευγών κορυφών, που ανήκουν στο V. Αυτά τα ζεύγη ονομάζονται ***ακμές***, |E| = m. Εάν οι ακμές δεν είναι ταξινομημένες, δηλαδή αν για i∈V, j∈V, η ακμή (i, j) και η (j, i) ανήκουν και οι δύο στο E, τότε το γράφημα ονομάζεται ***μη κατευθυνόμενο***. Ενώ αν κάθε ακμή (i, j) έχει κατεύθυνση τότε οι κόμβοι ορίζονται να αρχίζουν από την i και να τελειώνουν στην j.

Σε αυτή την περίπτωση το γράφημα ονομάζεται ***κατευθυνόμενο***. Οι κατευθυνόμενες ακμές συχνά ονομάζονται ***τόξα***. Ένα ***μονοπάτι*** είναι μία ακολουθία ακμών (i_1,i_2), $(i_2,i_3),\ldots,$ (i_{k-1},i_k) που ανήκουν στο E και οι οποίες συνδέουν τους κόμβους i_1 και i_k που ανήκουν στο V, το πλήθος αυτών των ακμών ορίζουν το ***μήκος*** του. Ένα γράφημα είναι ***συνδεδεμένο*** εάν υπάρχει ένα μονοπάτι μεταξύ οποιωνδήποτε δύο κορυφών του. Σε διαφορετική περίπτωση ονομάζεται ***μη συνδεδεμένο***. Η απόσταση μεταξύ δύο κορυφών v και u, συμβολίζεται σαν d(v,u) και ορίζεται σαν το μήκος του συντομότερου μονοπατιού μεταξύ των δύο αυτών κορυφών. Εάν δεν υπάρχει τέτοιο μονοπάτι, αυτή η απόσταση ορίζεται σαν άπειρη. ***Διάμετρος ενός γράφου G***, ονομάζεται η μέγιστη απόσταση μεταξύ δύο οποιονδήποτε κορυφών αυτού του γράφου και συμβολίζεται σαν diam(G). Ο αριθμός των ακμών που έχει μία κορυφή ονομάζεται βαθμός της κορυφής και συμβολίζεται σαν $d(v_i)$. Εάν ισχύει $d(v_i) = 0$ τότε αυτή η κορυφή ονομάζεται απομονωμένη κορυφή [6].

1.2.2 Αναπαράσταση γράφων

Η αναπαράσταση γραφημάτων έγινε αποδέκτης μεγάλης προσοχής από την αρχή της μελέτης της θεωρίας των υπολογιστών .Αυτό που γενικά απαιτείται είναι μία καλή και αποδοτική αναπαράσταση: καλή για την αποδοτικότητα των αλγορίθμων και αποδοτική και από την άποψη του χώρου για την αποθήκευση των δεδομένων και για τον χρόνο υπολογισμού που χρειάζεται για την παραγωγή της αναπαράστασης.

Τα γραφήματα και τα δίκτυα αναπαριστούνται επισήμως μαθηματικά ***με πίνακες γειτνίασης***[6].

Ο ***πίνακας γειτνίασης*** είναι ένας nxn πίνακας. Κάθε μη μηδενικό στοιχείο του πίνακα φανερώνει την ύπαρξη μίας ακμής μεταξύ των κορυφών της αντίστοιχης γραμμής και κορυφής. Εάν η ακμή χαρακτηρίζεται και από κάποια παράμετρο, όπως βάρος ,χωρητικότητα, κόστος κτλ , τότε το αντίστοιχο στοιχείο του πίνακα θα είναι μία εγγραφή που θα περιέχει αυτή την τιμή της παραμέτρου.
Ο πίνακας γειτνίασης ενός μη κατευθυνόμενου γραφήματος είναι συμμετρικός, ενώ των κατευθυνόμενων δεν είναι.

Αυτοί οι πίνακες είναι πολύ σημαντικοί καθώς με πράξεις πινάκων πάνω σε αυτούς πολλοί άλλοι πίνακες, που αφορούν διαφορετικές ιδιότητες των γραφημάτων μπορούν εύκολα να παραχθούν, όπως για παράδειγμα κύκλοι, επικαλύπτοντα δέντρα κτλ. Το μόνο μειονέκτημα αυτής της αναπαράστασης είναι ότι απαιτεί $\Omega(n^2)$ αποθηκευτικό χώρο ενώ συνήθως ο αριθμός των ακμών είναι μικρότερος του n^2, και φυσικά η προσπέλαση του πίνακα απαιτεί $O(n^2)$ χρόνο.

Η αναπαράσταση με μία λίστα που θα στηρίζεται στην διασύνδεση των κόμβων αποδεικνύεται ο καλύτερος τρόπος για την αποθήκευση ενός δικτύου καθώς από αυτή μπορούμε εύκολα να αποκτήσουμε εάν χρειαστεί και τον πίνακα γειτνίασης και των πίνακα ακμών, και έτσι και όποιον άλλον πίνακα μας είναι απαραίτητος. Αυτή η πράξη έχει πολυπλοκότητα O(n) και σε χρόνο και σε μνήμη.

Η ***λίστα γειτνίασης*** έχει ως εξής:
Μελετούμε ένα δίκτυο n κόμβων ,άρα n γραμμών. Κάθε στοιχείο της γραμμής i περιέχει μία εγγραφή της ακόλουθης δομής [(j), $p_{(i,j)}(1)$… $p_{(i,j)}(k)$], όπου j είναι ένας κόμβος που συνδέεται με ένα τόξο (που έχει ως αρχή το i και τέλος το j) με το i και $p_{(i,j)}(.)$ είναι οι παράμετροι του τόξου. Εάν έχουμε στη διάθεση μας την αναπαράσταση του γραφήματος σε λίστα γειτνίασης κάθε πίνακας μπορεί να κατασκευαστεί [6].

1.3 Πρόβλημα πλανόδιου πωλητή – Πολυπλοκότητα

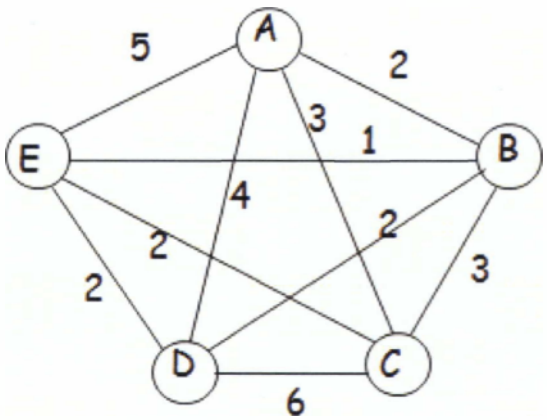
Στην παρούσα πτυχιακή εργασία θα ασχοληθούμε με το πρόβλημα του πλανόδιου πωλητή μας ζητά να βρούμε την μικρότερη διαδρομή που περνά από κάθε πόλη (κόμβο ενός γραφήματος) μόνο μια φορά και γυρνά σε αυτήν από την οποία ξεκίνησε. [21]

Το πρόβλημα αυτό δεν μπορεί να κατηγοριοποιηθεί εύκολα καθώς πλήθος αλγορίθμων διαφορετικών τεχνικών μπορούν να παράγουν λύση. [21]

Εξαντλητική αναζήτηση.

Μπορούμε να θεωρήσουμε πως ο αλγόριθμος για το πρόβλημα του πλανόδιου πωλητή μπορεί να ανήκει σε αυτή την κατηγορία γιατί μπορεί να υλοποιηθεί υπολογίζοντας όλα τα δυνατά μονοπάτια που ενώνουν όλες τις πόλεις και έπειτα να επιλέγουμε από αυτά το συντομότερο [21].

Για το παρακάτω γράφημα θα έχουμε:



Σχήμα 1.3.1 Παράδειγμα γραφήματος για το πρόβλημα του πλανόδιου πωλητή.

Μήκος(ABCDEA) = 2 + 3 + 6 + 2 + 5 = 18

Μήκος(ACBDEA) = 3 + 3 + 2 + 2 + 5 = 15

.....

Μήκος(ABDECA) = 2 + 2 + 2 + 2 + 3 = 11 ←

Μήκος(AEBCDA) = 5 + 1 + 3 + 6 + 4 = 19

Με βάση την τεχνική της εξαντλητικής αναζήτησης λοιπόν ο αλγόριθμος για n πόλεις υπολογίζει το μήκος (n-1)! Διαδρομών. Έτσι η πολυπλοκότητα είναι $\Omega(2^n)$ [21].

NP-Complete πρόβλημα.

Το πρόβλημα του πλανόδιου πωλητή ανήκει σε αυτή την κλάση πολυπλοκότητας καθώς θεωρείται δύσκολο πρόβλημα με εκθετικούς αλγορίθμους και χωρίς να γνωρίζουμε αν υπάρχει πολυωνυμική λύση για αυτό. Ιδιότητες που ικανοποιεί το συγκεκριμένο πρόβλημα και μπορεί να ανήκει σε αυτήν την κατηγορία είναι η όχι αποφασισμένη πολυωνυμικότητα και η πληρότητα. Η όχι αποφασισμένη πολυωνυμικότητα εμφανίζεται στο πρόβλημα του πλανόδιου πωλητή ως ένας μη ντετερμινιστικός πολυωνυμικός αλγόριθμος (όχι αποφασιστικός δηλαδή) που προσπαθεί να βρει λύσεις χρησιμοποιώντας τεχνάσματα όπως να επισκέπτεται την επόμενη πόλη με βάση την κοντινότερη απόσταση από την τρέχουσα πόλη. Η πληρότητα είναι η ιδιότητα που έχει ένα NP-πλήρες πρόβλημα να μετατρέπεται σε ένα άλλο NP-complete σε πολυωνυμικό χρόνο. Τότε, αν βρεθεί βέλτιστος πολυωνυμικός αλγόριθμος για ένα από αυτά τα προβλήματα, τότε υπάρχει τέτοιος αλγόριθμος για όλα. Για να λυθεί ένα NP-complete πρόβλημα αρκεί να μετασχηματιστεί στο πρόβλημα που λύνεται πολυωνυμικά Η πολυπλοκότητα της λύσης είναι το άθροισμα της πολυπλοκότητας μετατροπής και της λύσης του πρώτου προβλήματος. Στο σύνολο η πολυπλοκότητα θα είναι πολυωνυμική.

Ευρετικός – άπληστος αλγόριθμος.

Αλγόριθμοι απληστίας (greedy algorithms). Οι αλγόριθμοι απληστίας προσπαθούν να οδηγήσουν σε μια εφικτή λύση του προβλήματος, αλλά πολλές φορές χρειάζονται πάρα πολύ μεγάλο χρόνο γιατί είναι μυωπικοί αλγόριθμοι, δηλαδή βλέπουν μόνο μπροστά.

Σε αυτήν την κατηγορία ανήκουν προβλήματα που καταλήγουν σε λύση κάνοντας επιλογή διαλέγοντας αυτή με το μικρότερο κόστος. Έτσι για το TSP πρόβλημα (το πρόβλημα του πλανόδιου πωλητή) χρησιμοποιώντας αυτή την τεχνική και για το σχήμα 1.3.1 παίρνουμε σαν λύση την διαδρομή (BECADB) με μήκος ίσο με:

$$1 + 2 + 3 + 4 + 2 = 12.$$

Αυτή η τεχνική μπορεί να μας δίνει λύση δεν είναι όμως πάντα σίγουρα η βέλτιστη. Και πράγματι η βέλτιστη διαδρομή είναι η (BACEDB) με μήκος ίσο με:

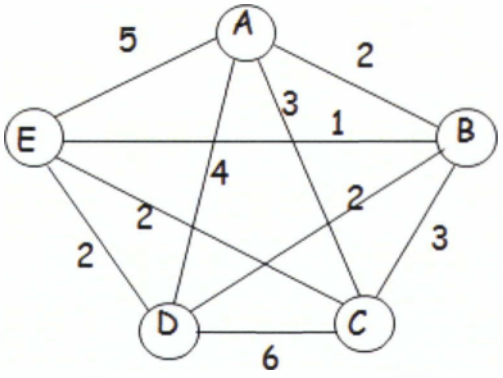
$$2 + 3 + 2 + 2 + 2 = 11.$$

Σε αυτή την περίπτωση ο αλγόριθμος είναι εκθετικής πολυπλοκότητας:
$f(n) = a^n$, $1 < a$, πολυπλοκότητα εκθετική [18].

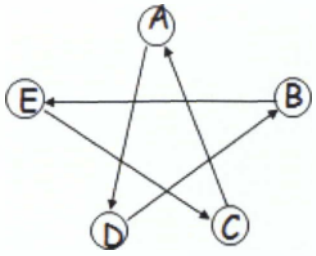
Τοπική αναζήτηση.

Το TSP πρόβλημα μπορεί να ενταχτεί σε αυτή την κατηγορία στην οποία ανήκουν προβλήματα βελτιστοποίησης. Σε αυτές τις περιπτώσεις σε ήδη υπάρχουσες μη βέλτιστες λύσεις ,που έχουν προκύψει από άπληστους αλγορίθμους, εφαρμόζονται αλγόριθμοι που τις βελτιώνουν. Αυτή η τεχνική μπορεί να εφαρμοστεί επαναληπτικά όσο η λύση βελτιώνεται. Η πολυπλοκότητα των μετατροπών πρέπει να είναι πολυωνυμική.

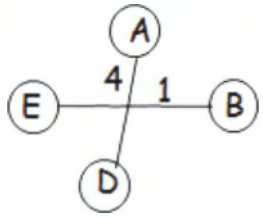
Έτσι για το προηγούμενο παράδειγμα του άπληστου αλγορίθμου που μας έδωσε λύση με μήκος μονοπατιού 12 η βελτιστοποίηση με τοπική αναζήτηση έχει ως εξής:



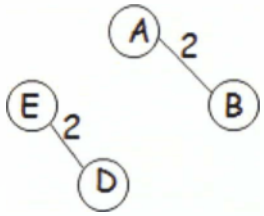
Σχήμα 1.3.1 Παράδειγμα γραφήματος για το πρόβλημα του πλανόδιου πωλητή



Σχήμα 1.3.2 Αποτέλεσμα άπληστου αλγορίθμου για το γράφημα του σχήματος 1.3.1

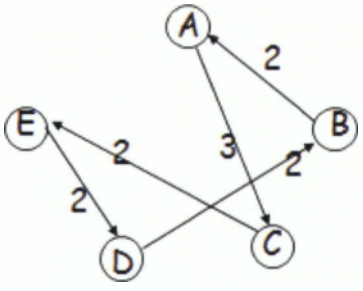


Σχήμα 1.3.3 Τοπικό κόστος ίσο με 5



Σχήμα 1.3.4 Βελτιστοποίηση του 1.3.3. Τοπικό κόστος ίσο με 4.

Έτσι η βέλτιστη λύση είναι αυτή του παρακάτω σχήματος και έχει μήκος διαδρομής ίσο με 11 < 12 :



Σχήμα 1.3.5 Αποτέλεσμα του TSP με την τοπική αναζήτηση (βελτιστοποίηση)

Δυναμικός προγραμματισμός

Με αυτή την τεχνική το αρχικό πρόβλημα διασπάται σε μικρότερα προβλήματα που λύνονται ευκολότερα. Οι λύσεις τους αποθηκεύονται και επαναχρησιμοποιούνται πολλές φορές όταν χρειάζονται για να λυθούν μεγαλύτερα υπο-προβλήματα και τελικά το αρχικό πρόβλημα Συχνά εκφράζεται με αναδρομική φόρμουλα που δηλώνει πως συνδυάζονται τα μικρότερα προβλήματα για να λύσουν ένα μεγαλύτερο.

Για το πρόβλημα του πλανόδιου πωλητή ο βέλτιστος αλγόριθμος έχει πολυπλοκότητα Ω((n-1)!). Αν όμως τα ίδια μερικά μονοπάτια χρησιμοποιούνται πολλές φορές τότε ο δυναμικός προγραμματισμός μπορεί να είναι γρηγορότερος!

Κεφάλαιο 2. Αλγόριθμος Hamilton

2.1 Κύκλος Hamilton: Ιστορία – Εισαγωγή

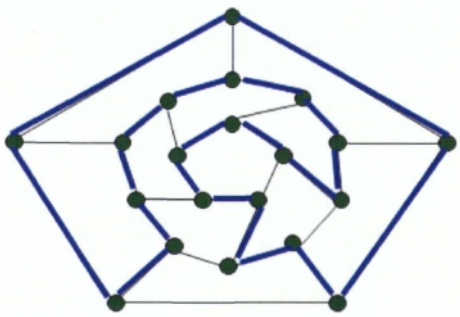
Ο Sir William Rowan Hamilton (4 Αυγούστου 1805 - 2 Σεπτεμβρίου 1865) ήταν ένας Ιρλανδός φυσικός, αστρονόμος και μαθηματικός, ο οποίος είχε σημαντικές συνεισφορές στην κλασική μηχανική, στην οπτική, και στην άλγεβρα. Οι μελέτες του στα μηχανικά και οπτικά συστήματα οδήγησαν στην ανακάλυψη νέων μαθηματικών εννοιών και τεχνικών. Η μεγαλύτερη συμβολή του ίσως είναι η αναδιατύπωση της νευτώνειας μηχανικής, που σήμερα ονομάζεται μηχανική Hamilton. Το έργο αυτό έχει αποδειχθεί κεντρικής σημασίας για την σύγχρονη μελέτη του ηλεκτρομαγνητισμού, καθώς και στην ανάπτυξη της κβαντικής μηχανικής [10].

Ως μαθηματικός το 1856 ανέπτυξε γύρω από την θεωρία των γραφημάτων, τη δική του θεωρία που ονομάζεται Χαμιλτονιανή. Το 1859 κατασκεύασε ένα παιχνίδι που λέγονταν «Το δωδεκάεδρο του Ταξιδιώτη» (Σχ.2.1.1). Βγήκε στην αγορά του Λονδίνου και κόστιζε μόνο 95 λίρες. Το παιχνίδι αυτό αποτελούνταν από ένα ξύλινο δωδεκάεδρο με τα ονόματα είκοσι πόλεων στις είκοσι κορυφές του, τοποθετημένες με αλφαβητική σειρά, ξεκινώντας από Β (Βρυξέλλες) και καταλήγοντας σε Ζ (Ζανζιβάρη).

Οι ακμές του δωδεκάεδρου ήταν τονισμένες με μαύρες γραμμές και αναπαριστούσαν τις αποστάσεις ανάμεσα στις πόλεις. Το ζητούμενο ήταν να επισκεφτεί κάποιος όλες τις πόλεις από μια και μόνο φορά. Δηλαδή έπρεπε να βρεθεί μια διαδρομή που θα περνάει από όλες τις πόλεις αλλά δεν θα περνάει από καμία πόλη 2 φορές. Το ανάλογο του δωδεκάεδρου στο χαρτί είναι το παρακάτω σχήμα.

Σχήμα 2.1.1 Το δωδεκάεδρο του Ταξιδιώτη

Μεταφέροντας το παιχνίδι σε γράφο (Σχ.2.1.2) το ερώτημα γίνεται εάν είναι δυνατόν σε κάθε γράφο να βρεθεί κύκλος (=κλειστό μονοπάτι) που να περνά από όλες τις κορυφές μια και μόνο φορά. Αν βρεθεί τότε αυτός θα ονομάζεται **Κύκλος Hamilton** [10].



Σχήμα 2.1.2 Κύκλος Hamilton

2.2 Θεωρήματα κύκλου Hamilton

Θεώρημα 2.1: Κάθε πλήρης γράφος είναι Hamilton.

Θεώρημα 2.2: Κάθε πλήρης γράφος με n κορυφές (n περιττός) έχει (n-1)/2 Hamilton κύκλους ξένους ως προς ακμές.

Θεώρημα 2.3 (Dirac 1952): Κάθε απλός γράφος με $n \geq 3$ και $d(G) \geq n/2$ είναι Hamiltonian.

Θεώρημα 2.4 (Ore 1960): Κάθε απλός γράφος με $n \geq 3$ και $d(x)+d(y) \geq n$ για κάθε ζεύγος μη γειτονικών κορυφών x,y είναι Hamiltonian.

Θεώρημα 2.5: Κάθε απλός γράφος με $n \geq 3$ και $d(x)+d(y) \geq n$ για δύο διακριτές μη γειτονικές κορυφές x,y είναι Hamiltonian, αν ο γράφος $G+(x,y)$ είναι Hamiltonian.

Κλείσιμο-closure γράφου είναι ένας γράφος c(G) με επιπλέον ακμές για τα ζεύγη μη γειτονικών κορυφών x και y, όπου ισχύει $d(x)+d(y) \geq n$.

Θεώρημα 2.6 (Bondy-Chvatal 1976): Κάθε απλός γράφος είναι Hamiltonian, αν και μόνον αν έχει κλείσιμο Hamiltonian.

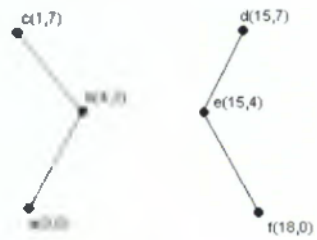
2.3 Άπληστος Αλγόριθμος εύρεσης κύκλων Hamilton

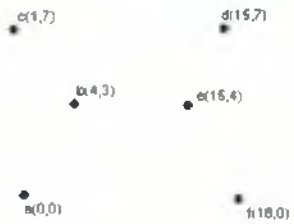
Αρχικά θεωρούμε ένα στιγμιότυπο ενός ολοκληρωμένου γράφου με τις πόλεις να παριστάνονται σαν κορυφές με απόσταση $d(c_i , c_j)$ ανάμεσα σε κάθε ζεύγος πόλεων $\{c_i , c_j\}$. Στην περίπτωση αυτή το μονοπάτι είναι ο Hamiltonian κύκλος (δηλαδή ο κύκλος που επισκέπτεται κάθε κορυφή ακριβώς μια φορά και επιστρέφει στην αρχική κορυφή) του γράφου, π.χ., μια συνδεδεμένη συλλογή από κορυφές με κάθε πόλη να έχει βαθμό 2. Φτιάχνουμε τον κύκλο προχωρώντας μια κορυφή τη φορά, ξεκινώντας από τη κοντινότερη κορυφή, και προσθέτοντας κάθε φορά μια από τις πιο κοντινές κορυφές που έχουν απομείνει.

Μια κορυφή θεωρείται διαθέσιμη όταν δεν είναι ήδη μέρος της διαδρομής και αν την προσθέσουμε δε θα αυξήσουμε το βαθμό της κορυφής, που την προσθέτουμε, σε 3 ή αν δε δημιουργεί κύκλο μικρότερου μεγέθους από τον αριθμό των κορυφών του γράφου N.

Όλο αυτό μπορεί να φανεί καλύτερα με ένα παράδειγμα:

Στο Σχήμα 2.3.1 βλέπουμε 6 πόλεις μαζί με τις συντεταγμένες τους. Οι αποστάσεις μεταξύ των πόλεων είναι {((d,e),3) , ((b,c),5) ,((a,b),5) , ((e,f),5) , ((a,c),7.08) , ((d,f), 58) , ((b,e), 22) , ((b,d), 137) , ((c,d),14) ,… ((a,f),18)}.





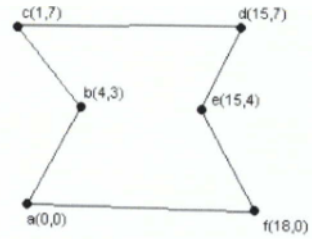
Σχήμα 2.3.1 Διάταξη πόλεων

Ο αλγόριθμος δουλεύει ως εξής:

επιλέγουμε (d,e)
επιλέγουμε (a,b)
επιλέγουμε (b,c)
επιλέγουμε (e,f)
δεν επιλέγουμε (a,c), δημιουργεί κύκλο με τα (a,b) και (b,c)
δεν επιλέγουμε (d,f), δημιουργεί κύκλο με τα (d,e) και (e,f)
δεν επιλέγουμε (b,e), γιατί κάνει το βαθμό του b ίσο με 3
δεν επιλέγουμε (b,d), γιατί κάνει το βαθμό του b ίσο με 3
επιλέγουμε (c,d)
...
...
...

επιλέγουμε (a,f)

Και έχουμε την τελική διαδρομή που φαίνεται στο Σχήμα 2.3.2.



Σχήμα 2.3.2 Τελική διαδρομή

Κεφάλαιο 3. Αλγόριθμος Prim

Στην επιστήμη των υπολογιστών, ο αλγόριθμος Prim είναι ένας άπληστος αλγόριθμος που βρίσκει ένα ελάχιστο spanning-tree για ένα συνδεδεμένο σταθμισμένο μη-κατευθυνόμενο γράφημα. Αυτό σημαίνει ότι βρίσκει ένα υποσύνολο των ακμών το οποίο σχηματίζει ένα δέντρο που περιλαμβάνει κάθε κορυφή, όπου το συνολικό βάρος όλων των ακμών στο δέντρο είναι το ελάχιστο δυνατό.

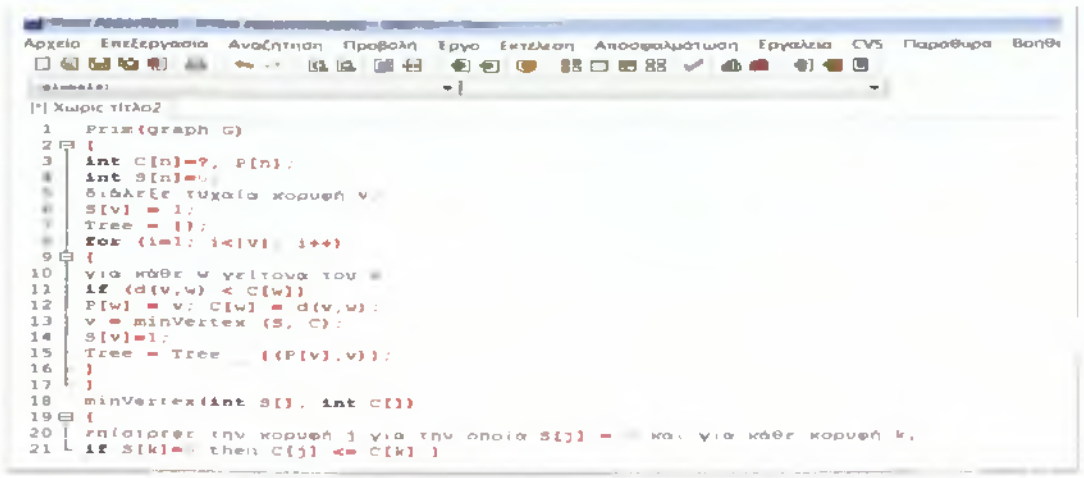
3.1 Ο αλγόριθμος

Η ιδέα του αλγορίθμου Prim, είναι να διατηρεί κάθε στιγμή ένα μοναδικό MST (minimum spanning tree – ελάχιστο συνδετικό δέντρο) το οποίο και θα επεκτείνει ακμή προς ακμή.

Ξεκινώντας από το στοιχειώδες δέντρο της μίας κορυφής και διατηρώντας μία ουρά προτεραιότητας για τις κορυφές που δεν έχουν συμπεριληφθεί ακόμα στο MST κάθε φορά επιλέγει ποια θα είναι η επόμενη κορυφή που θα εισαχθεί στο υπό κατασκευήν ελάχιστο επικαλύπτον δέντρο. Η προτεραιότητα της κάθε κορυφής είναι το βάρος της ελαφρύτερης ακμής που την συνδέει με το τρέχον MST.

Ο αλγόριθμος Prim σε κάθε βήμα του εξάγει από την ουρά προτεραιότητας την κορυφή, υ, με την μικρότερη προτεραιότητα και την εισάγει στο MST και έπειτα ανανεώνει την προτεραιότητα των κορυφών u που είναι γειτονικές με την υ. Ειδικότερα το βάρος, w, της ακμής (υ,u) συγκρίνεται με την προτεραιότητα της κορυφής u στην ουρά προτεραιότητας και γίνεται ανανέωση της προτεραιότητάς της εάν το w είναι μικρότερο από την τρέχουσα προτεραιότητα [14].

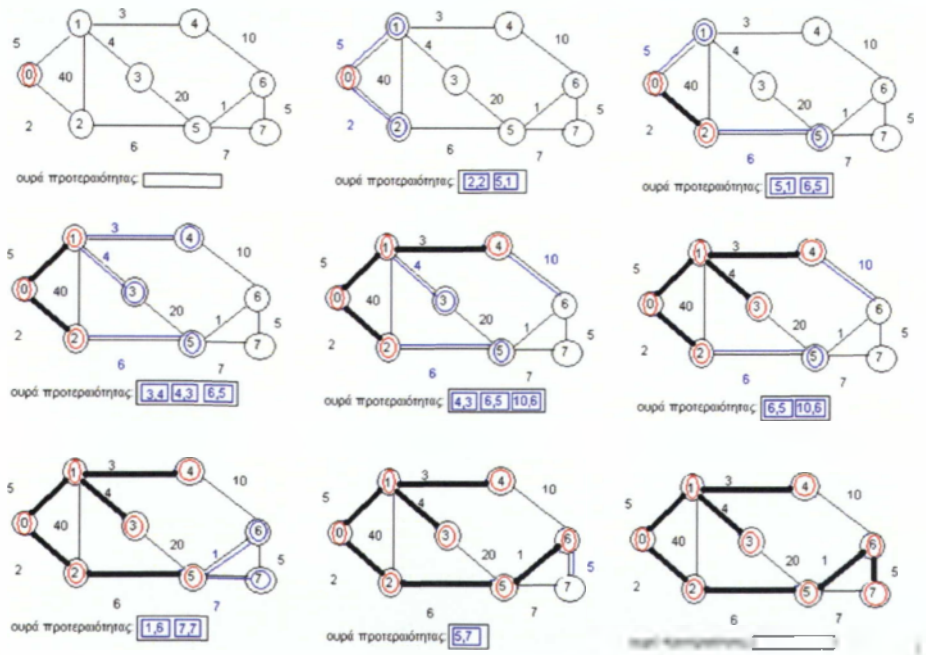
Παρακάτω (σχήμα 3.1) παρουσιάζουμε τον αλγόριθμο του Prim:



Σχήμα 3.1 Ψευδοκώδικας αλγορίθμου Prim

3.2 Παράδειγμα αλγορίθμου Prim

Στο παρακάτω σχήμα παρουσιάζεται ένα παράδειγμα του αλγόριθμου Prim [6]



Σχήμα 3.2 Παράδειγμα αλγόριθμου Prim

Κεφάλαιο 4. Αλγόριθμοι Προσέγγισης και Βελτιστοποίησης

4.1 Αλγόριθμοι Βελτιστοποίησης

Ας υποθέσουμε ότι εργαζόμαστε σε ένα πρόβλημα βελτιστοποίησης στο οποίο κάθε δυνητική λύση έχει ένα θετικό κόστος, και θέλουμε να βρούμε μια σχεδόν-βέλτιστη λύση. Ανάλογα με το πρόβλημα, μπορούμε να ορίσουμε ως βέλτιστη λύση αυτή με το μέγιστο δυνατό κόστος ή ένα με το ελάχιστο δυνατό κόστος, δηλαδή, το πρόβλημα μπορεί να είναι είτε μεγιστοποίησης είτε ελαχιστοποίησης.

Λέμε ότι ένας αλγόριθμος για ένα πρόβλημα που έχει λόγο προσέγγισης n αν, για κάθε είσοδο μεγέθους n, το κόστος C της παραγόμενης λύσης από τον αλγόριθμο είναι παράγοντα n του κόστους, C* της βέλτιστης λύσης.

$$\text{Max}(C/C^*, C^*/C) \leq \rho(n).$$

Εάν ένας αλγόριθμος επιτυγχάνει λόγο προσέγγισης του N εμείς τον αποκαλούμε N-προσεγγιστικό αλγόριθμο. Για ένα πρόβλημα μεγιστοποίησης, $0 < C \leq C^*$, και ο λόγος C^*/C δίνει τον παράγοντα με τον οποίο το κόστος μιας βέλτιστης λύσης είναι μεγαλύτερο από το κόστος μιας προσεγγιστικής λύσης. Παρομοίως, για ένα πρόβλημα ελαχιστοποίησης, $0 < C^* \leq C$, ο λόγος C/C^* δίδει τον παράγοντα για τον οποίο το κόστος της προσεγγιστική λύσης είναι μεγαλύτερο από το κόστος της βέλτιστης λύσης. Επειδή υποθέτουμε ότι όλες οι λύσεις έχουν θετικό κόστος, οι λόγοι αυτοί είναι πάντα σαφώς καθορισμένοι. Ο λόγος προσέγγισης ενός προσεγγιστικού αλγόριθμου δεν είναι ποτέ μικρότερος από 1, καθόσον $C/C^* \leq 1$ συνεπάγεται $C^*/C \geq 1$. Επομένως, ένας 1-προσεγγιστικός αλγόριθμος παράγει μια βέλτιστη λύση, και ένας προσεγγιστικός αλγόριθμος με μεγάλο βαθμό προσέγγισης μπορεί να επιστρέψει μια λύση που είναι πολύ χειρότερη από την βέλτιστη.

Τα προβλήματα που εμφανίζονται στη φύση μπορεί να είναι δύσκολα προβλήματα Βελτιστοποίησης ή να μοντελοποιούνται από συνεχείς συναρτήσεις που εμφανίζουν ένα μεγάλο πλήθος βέλτιστων λύσεων κάνοντας κατά αυτόν τον τρόπο δύσκολη, αν

όχι αδύνατη, την επίλυσή τους. Για το λόγο αυτό αναπτύχτηκαν οι αλγόριθμοι βελτιστοποίησης ειδικά για φυσικά, κοινωνικά ή βιολογικά φαινόμενα. Οι αλγόριθμοι αυτοί έχουν την ιδιότητα να εντοπίσουν την περιοχή των βέλτιστων λύσεων [13].
Γενικά σε ένα πρόβλημα βελτιστοποίησης, ο σκοπός μας να προσδιορίσουμε ένα διάνυσμα λύσεων το οποίο θα ικανοποιεί μια συνάρτηση κόστους ή ένα αντικειμενικό κριτήριο.

Η διαδικασία εύρεσης λύσεων στους αλγορίθμους βελτιστοποίησης πραγματοποιείται πάντοτε μέσω μιας επαναληπτικής διαδικασίας. Σε κάθε επανάληψη δημιουργούνται από τον αλγόριθμο μια ή περισσότερες λύσεις από τον αντίστοιχο αριθμό ατόμων (individuals) που ανήκουν στο χώρο των πιθανών λύσεων. Η καταλληλότητα των λύσεων αυτών εκτιμάται μέσω της συναρτήσεως κόστους. Πολλές φορές οι αλγόριθμοι αυτοί κατά τη διαδικασία εύρεσης λύσεων προσαρμόζονται συνεχώς στο χώρο των λύσεων μεταβάλλοντας τις παραμέτρους τους.

Η παραπάνω γενική διαδικασία συνεχίζεται μέχρις ότου ολοκληρωθεί ένας συγκεκριμένος αριθμός επαναλήψεων ή μέχρις ότου ικανοποιηθεί ένα κριτήριο σύγκλισης.

Τα τρία κυριότερα μειονεκτήματα των στοχαστικών αλγορίθμων είναι τα εξής:

- πολλές φορές δυσκολεύονται να προσεγγίσουν την ακριβή τιμή της βέλτιστης λύσης του προβλήματος αν και συνήθως εντοπίζουν γρήγορα την περιοχή στην οποία αυτή βρίσκεται.
- απαιτούν μεγάλο αριθμό εκτιμήσεων της αντικειμενικής συνάρτησης – και επομένως έχουν υψηλό υπολογιστικό κόστος – προκειμένου να δώσουν ένα καλό αποτέλεσμα.
- συνήθως, έχουν ένα μεγάλο αριθμό παραμέτρων ο οποίος κάνει δύσκολη τη βέλτιστη ρύθμιση του αλγορίθμου.

Η σχέση μεταξύ μέγιστου αριθμού επαναλήψεων και ποιότητας της βέλτιστης λύσης είναι άμεση και πρέπει να λαμβάνουμε υπόψη μας το πρόβλημα που θέλουμε να

επιλύσουμε. Προφανώς, θα πρέπει να γνωρίζουμε εκ των προτέρων αν απλά επιθυμούμε μια καλή λύση ή αν είναι απαραίτητος ο εντοπισμός της βέλτιστης λύσης.

Ένα σημαντικό χαρακτηριστικό των αλγορίθμων βελτιστοποίησης που πρέπει να λαμβάνεται υπόψη από το χρήστη είναι η επαναληψιμότητα της απόδοσης του αλγορίθμου. Κάθε εκτέλεσή τους με διαφορετικές αρχικές συνθήκες δίνει και διαφορετικό αποτέλεσμα. Υπολογίζοντας τη μέση τιμή της βέλτιστης λύσης και την τυπική απόκλιση αυτής για ένα μεγάλο αριθμό εκτελέσεων του αλγορίθμου, θα πρέπει η τυπική απόκλιση να είναι σχετικά μικρή. Το γεγονός αυτό προσφέρει τη σιγουριά στο χρήστη ότι ο αλγόριθμος που χρησιμοποιεί θα έχει την αναμενόμενη απόδοση στις περισσότερες εκτελέσεις του.

4.2 Παράγοντας προσέγγισης-Προβλήματα μεγιστοποίησης-ελαχιστοποίησης

Αναφορικά στην απόδοση των προσεγγιστικών αλγόριθμων, μελετάμε την χειρότερη περίπτωση, όχι μόνο ως προς την χρονική πολυπλοκότητα, αλλά και ως προς την βέλτιστη λύση του προβλήματος. Δηλαδή ενδιαφερόμαστε για το πόσο κοντά στη βέλτιστη λύση ($OPT_{\Pi}(I)$), είναι η λύση που μας δίνει ένας προσεγγιστικός αλγόριθμος ($SOL_A(I)$). Ένας αλγόριθμος Α, που επιλύει ένα πρόβλημα ελαχιστοποίησης Π, έχει παράγοντα-πηλίκο προσέγγισης (approximation factor-ratio) έναν πραγματικό αριθμό $\rho \geq 1$, αν για κάθε στιγμιότυπο I του Π ισχύει [16]:

$$SOL_A I \leq \rho \cdot OPT_{\Pi} I \Rightarrow SOL_A(I) / OPT_{\Pi}(I) \leq \rho$$

Αντίστοιχα για ένα πρόβλημα μεγιστοποίησης θα ισχύει:

$$OPT_{\Pi}(I) \geq SOL_A(I) \geq \rho \cdot OPT_{\Pi}(I) \Rightarrow OPT_{\Pi}(I) / SOL_A(I) \leq \rho$$

Και στις δύο περιπτώσεις χαρακτηρίζουμε τον Α σαν ρ-προσεγγιστικό αλγόριθμο.

Εναλλακτικά, μπορούμε να ορίσουμε τον παράγοντα προσέγγισης σαν

- $\rho = max\ SOL_A\ (I)\ /\ OPT_{\Pi}\ (I), \forall I \in F_{\Pi}\ I$ για προβλήματα ελαχιστοποίησης και
- $\rho = max\ OPT_{\Pi}\ (I)\ /\ SOL_A\ (I)\ , \forall I \in F_{\Pi}\ I$ για προβλήματα μεγιστοποίησης.

Από τον παραπάνω ορισμό, φαίνετε πως μας ενδιαφέρει η τιμή του ρ στην χειρότερη περίπτωση (worst case), δηλαδή για το στιγμιότυπο I εκείνο, όπου η τιμή $SOL_A\ (I)$ απέχει την μεγαλύτερη απόσταση από την τιμή της βέλτιστης λύσης για το Π ($OPT_{\Pi}\ (I)$). Όσο η τιμή του ρ πλησιάζει στην μονάδα, τόσο ο αλγόριθμος Α γίνεται αποδοτικότερος, αφού η λύση που μας επιστρέφει πλησιάζει την βέλτιστη λύση. Ένας αλγόριθμος που έχει ρ = 1, παύει να είναι προσεγγιστικός, καθώς επιστρέφει πάντα την βέλτιστη λύση. Εν γένει ο παράγοντας ρ μπορεί να μην είναι ένας σταθερός αριθμός για κάποιον αλγόριθμο Α, αλλά να είναι συνάρτηση της εισόδου του. Αν θεωρήσουμε ότι |I| = n, τότε στην γενική περίπτωση ο παράγοντας προσέγγισης θα είναι $\rho_A\ (n)$ [16]. Τέλος, να τονίσουμε ότι ο αλγόριθμος που θα ασχοληθούμε ανήκει στα προβλήματα ελαχιστοποίησης.

4.3 Πλανόδιος πωλητής

Το πρόβλημα του πλανόδιου πωλητή αφορά στην εύρεση μιας διαδρομής που πρέπει να ακολουθήσει ένας πωλητής ο οποίος κινείται επάνω σε ένα δίκτυο πόλεων, ούτως ώστε να επισκέπτεται το σύνολο των πόλεων ελαχιστοποιώντας το κόστος της διαδρομής που ακολουθεί [11]. Το δίκτυο των πόλεων αναπαρίσταται με ένα γράφο, οι κόμβοι του οποίου αναπαριστούν τις πόλεις ενώ οι ακμές του το οδικό δίκτυο που συνδέει τις πόλεις αυτές. Για την επίλυση του συγκεκριμένου προβλήματος, έχουν προταθεί διάφοροι αλγόριθμοι οι οποίοι ως ένα βαθμό επιλύουν ικανοποιητικά το πρόβλημα. Ωστόσο, η πολυπλοκότητα επίλυσης του προβλήματος, ιδιαίτερα όταν αυτό αφορά μεγάλους γράφους, είναι μη- πολυωνυμική καθώς η μόνη μέθοδος που εγγυάται τη λύση του προβλήματος συνίσταται στον εξαντλητικό υπολογισμό του κόστους (exhaustion method) του συνόλου των πιθανών διαδρομών και στην επιλογή της συντομότερης. Αυτό όμως καθίσταται εξαιρετικά πολύπλοκο για μεγάλους γράφους, διότι αυξάνεται πολύ ο χρόνος επεξεργασίας των δεδομένων. Συνεπώς, δεν υφίσταται μια συγκεκριμένη ενιαία μέθοδος που να λύνει οποιαδήποτε περίπτωση του προβλήματος του πλανόδιου πωλητή.[12]

Υπάρχουν μόνο ad hoc διαδικασίες που βασίζονται σε ευρετικούς κανόνες οι οποίες όμως λύνουν προσεγγιστικά το πρόβλημα καθώς δε δίνουν τη βέλτιστη λύση.

Μια ευρετική λύση του προβλήματος θα μπορούσε να διατυπωθεί ως ακολούθως:

- Έστω ένας κόμβος ε, ο οποίος επιλέγεται ως κόμβος εκκίνησης και ονομάζεται κόμβος *ν*.
- Εν συνεχεία, αναζητείται κατά προτεραιότητα η ακμή *(ν, u)* από τον κόμβο *ν* η οποία δεν έχει διαπεραστεί και έχει το ελάχιστο κόστος διάσχισης.

Αφού, επισημανθεί ως διαπερασμένη, ο κόμβος *u* μετονομάζεται σε κόμβο *ν* και η διαδικασία επαναλαμβάνεται μέχρις ότου προσπελαθούν όλοι οι κόμβοι και ο αλγόριθμος επιστρέψει στο αρχικό σημείο εκκίνησης

4.4 Προσεγγιστικοί αλγόριθμοι

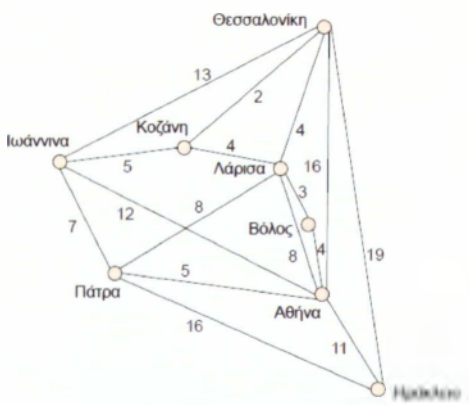
Όπως γνωρίζουμε υπάρχουν προβλήματα τα οποία είναι υπολογιστικά δυσεπίλυτα, καθώς ο χρόνος που απαιτείται από κάποιον αλγόριθμο για να επιστρέψει μία λύση είναι ιδιαίτερα μεγάλος, καθιστώντας συχνά τον υπολογισμό λύσης πρακτικά ανέφικτο. Πολλές φορές καλούμαστε να επιλύσουμε τέτοια προβλήματα βελτιστοποίησης, έχοντας περιορισμένο χρόνο. Στις περιπτώσεις εκείνες που μας ενδιαφέρει να βρούμε μία λύση του προβλήματος, σε μικρό χρονικό διάστημα, και είμαστε ελαστικοί ως προς το αν η λύση που θα βρούμε θα είναι βέλτιστη ή όχι μπορούμε να χρησιμοποιήσουμε αλγορίθμους που βρίσκουν μία σχεδόν βέλτιστη (near- optimal) λύση σε πολυωνυμικό χρόνο. Τέτοιοι αλγόριθμοι, που βρίσκουν για κάποιο πρόβλημα μία λύση η οποία εν γένει δεν είναι βέλτιστη αλλά είναι κοντά στην βέλτιστη, ονομάζονται προσεγγιστικοί αλγόριθμοι [16].

Στη συνέχεια θα συμβολίζουμε με Π το πρόβλημα βελτιστοποίησης, το οποίο καλούμαστε να επιλύσουμε και με Α τον αλγόριθμο τον οποίο θα χρησιμοποιήσουμε για την επίλυση του. Το Π μπορεί να είναι ένα πρόβλημα ελαχιστοποίησης ή μεγιστοποίησης. Για κάθε στιγμιότυπο I του Π, θα υπάρχει ένα σύνολο από εφικτές λύσεις (feasible solution), $F_{\Pi}(I)$. Θα υπάρχει αλγόριθμος πολυωνυμικού χρόνου, οποίος δεχόμενος μια πιθανή λύση s και το στιγμιότυπο I, μπορεί να αποφανθεί αν

$s \in F_\Pi(I)$. Για κάθε εφικτή λύση $s \in F_\Pi(I)$, υπάρχει αντικειμενική συνάρτηση C_Π, υπολογίσιμη σε πολυωνυμικό χρόνο, η οποία αναθέτει έναν μη-αρνητικό ρητό αριθμό σε κάθε ζεύγος (I,s). Η τιμή $C_\Pi(I,s)$ που προκύπτει από την εφαρμογή του αλγόριθμου A, είναι η λύση που παίρνουμε από τον A, και την συμβολίζουμε $SOL_A(I)$. Για ένα πρόβλημα ελαχιστοποίησης, η βέλτιστη λύση είναι η εφικτή λύση της οποίας η τιμή της αντικειμενικής συνάρτησης C_Π είναι η ελάχιστη. Στην περίπτωση που έχουμε πρόβλημα μεγιστοποίησης, αντίστοιχα είναι η εφικτή λύση με την μέγιστη τιμή αντικειμενικής συνάρτησης. Και στις δύο περιπτώσεις συμβολίζουμε την βέλτιστη λύση με $OPT_\Pi(I)$. Το μέγεθος του στιγμιότυπου I θα συμβολίζουμε με |I| και θα αντιστοιχεί στο πλήθος των bits που χρειαζόμαστε για την αναπαράσταση του I σε δυαδική μορφή [16].

4.5 Το πρόβλημα του πλανόδιου πωλητή με τριγωνική ανισότητα

Ο πωλητής πρέπει να επισκεφτεί τις πόλεις που αναπαριστώνται στο παρακάτω σχήμα (βλ. σχήμα 4.5.1) και να καταλήξει στην πόλη όπου ξεκίνησε την περιοδεία του, ελαχιστοποιώντας το συνολικό κόστος μετακίνησης



Σχήμα 4.5.1 Πόλεις

Για το παραπάνω πρόβλημα έχουμε:

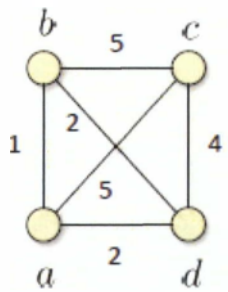
- Ένα πλήρες γράφημα G = (V,E) και
- μία συνάρτηση κόστους c : E ->Z, c(u,υ) ≥ 0, ∀{u,υ} ϵ E.

Για την περιοδεία του πωλητή θα μας χρειαστεί να βρούμε έναν κύκλο Hamilton, δηλαδή έναν κύκλο που θα επισκέπτεται κάθε κόμβο ακριβώς μία φορά ενώ το πρόβλημα βελτιστοποίησης που καλούμαστε να επιλύσουμε είναι η εύρεση μία περιοδείας δηλαδή του κύκλου Hamilton με το ελάχιστο συνολικό κόστος.

$$\sum_{i=1}^{n} c(v_{i-1}, v_i)$$

Για ένα σύνολο ακμών A υποσύνολο του E ορίζουμε το κόστος A ως:

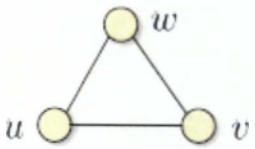
$$c(A) = \sum_{\{u,v\} \in A} c(u,v)$$



Σε πολλές περιπτώσεις το μικρότερο κόστος διαδρομής για να πάμε από μία πόλη σε άλλη είναι να πάμε απ' ευθείας χωρίς ενδιάμεσες στάσεις. Με άλλα λόγια η αποκοπή μιας ενδιάμεσης στάσης ποτέ δεν αυξάνει το κόστος διαδρομής. Έτσι μπορούμε να πούμε πως για να βρούμε μία σχεδόν βέλτιστη λύση χρειάζεται η συνάρτηση

κόστους, c, να ικανοποιεί την τριγωνική ανισότητα. Δηλαδή για κάθε τριάδα κόμβων u, υ, w να ισχύει :

$$\mathbf{c(u,w) \leq c(u,\upsilon) + c(\upsilon,w).}$$

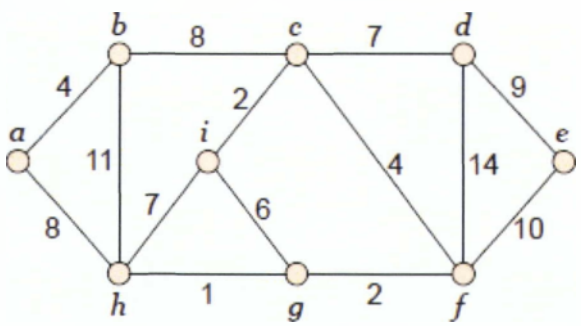


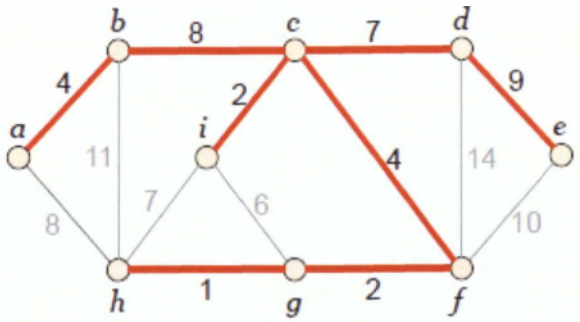
Αν οι κορυφές του γράφου είναι οι πόλεις στο επίπεδο και το κόστος διαδρομής είναι η μεταξύ τους ευκλείδεια απόσταση τότε ικανοποιείται η τριγωνική ανισότητα.

Το πρόβλημα του πλανόδιου πωλητή είναι ένα NP-πλήρες πρόβλημα ακόμα και στην περίπτωση που ικανοποιείται η τριγωνική ανισότητα. Έτσι δεν θα πρέπει να ψάχνουμε για αλγόριθμο πολυωνυμικού χρόνου που να επιλύει το πρόβλημα επακριβώς. Αντί αυτού ψάχνουμε για καλούς προσεγγιστικούς αλγορίθμους.

Για να υπολογίσουμε το κάτω φράγμα του κόστους της βέλτιστης περιοδείας θα πρέπει να υπολογίσουμε το βάρος - κόστος του ελαφρύτατου συνδετικού δέντρου, Τ, του γράφου G.

Θέλουμε δηλαδή να υπολογίσουμε το ελαφρύτατο δέντρο του παρακάτω γράφου με το ελάχιστο άθροισμα βαρών των ακμών που το αποτελούν. Το οποίο μπορεί να υπολογιστεί σε σχεδόν γραμμικό χρόνο (βλ. σχήμα 4.5.2).





Σχήμα 4.5.2 Υπολογισμός ελαφρύτατου δέντρου

Ο προσεγγιστικός αλγόριθμος έχει ως εξής:

1. Επιλέγουμε μία κορυφή r ϵ G ως αφετηρία.
2. Υπολογίζουμε ένα ελαφρύτατο συνδετικό δένδρο, Τ, του G από την κορυφή r (βλ.Σχήμα 4.5.2).
3. Επιλέγουμε αυθαίρετα ένα κόμβο r ϵ V ως ρίζα και εκτελούμε καθοδική διερεύνηση του Τ με αφετηρία το r.
4. Επιστρέφουμε την περιοδεία που σχηματίζεται από την διάταξη των κόμβων ως προς τους χρόνους ανακάλυψης της καθοδικής διερεύνησης (Κύκλος Hamilton H).

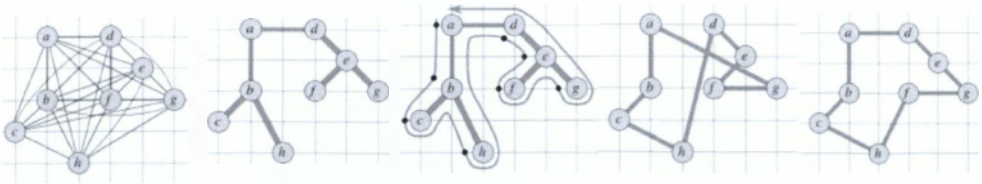
Θεώρημα:

Ο αλγόριθμος είναι 2-προσεγγιστικός πολυωνυμικού χρόνου για το πρόβλημα του περιοδεύοντος πωλητή με τριγωνική ανισότητα.

Απόδειξη:

Έστω Η* μια βέλτιστη περιοδεία και {u,υ} μια οποιαδήποτε ακμή της. Τότε το:

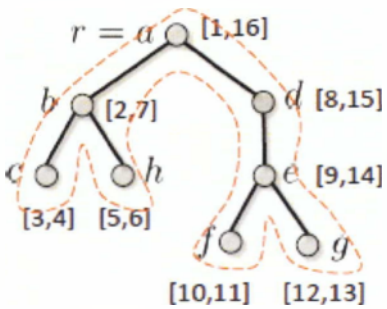
Τ' = Η* - {h,f} είναι συνδετικό δένδρο, άρα $c(H^*) \geq c(T') \geq c(T)$.



Σχήμα 4.5.3 από αριστερά προς τα δεξιά. Περιοδεία Η, Βέλτιστη περιοδεία Η* (23% συντομότερη), συνδετικό δέντρο Τ' = Η* - {h,f} , ελαφρύτατο συνδετικό δέντρο Τ.

Ως ***πλήρης διάνυση του Τ*** ονομάζουμε μία καταγραφή όλης της σειράς των επισκέψεων των κόμβων κατά την καθοδική διερεύνηση. Άρα μία πλήρης διάνυση W διατρέχει το Τ δύο φορές και $c(W) = 2c(T) \leq 2c(H^*)$ και λόγω της τριγωνικής ανισότητας $c(W) \geq c(H)$. Άρα $c(H) \leq 2c(H^*)$ Στο σχήμα 4.5.4 βλέπουμε την πλήρη διάνυση,W, του Τ η οποία έχει ως εξής:

W = (a,b,c,b,h,b,a,d,e,f,e,g,e,d,a) και π.χ c(e,f) + c(f,e) + c(e,g) ≤ c(e,f) + c(f,g).



Σχήμα 4.5.4 Πλήρης διάνυση

4.6 Το πρόβλημα του πλανόδιου πωλητή χωρίς τριγωνική ανισότητα

Αν η συνάρτηση κόστους δεν ικανοποιεί την τριγωνική ανισότητα τότε είναι αδύνατο να βρούμε σε πολυωνυμικό χρόνο μια ρ-προσεγγιστική περιοδεία, για οποιαδήποτε σταθερά $\rho \geq 1$, εκτός εάν P =NP. Η ύπαρξη ενός ρ-προσεγγιστικού αλγόριθμου πολυωνυμικού χρόνου συνεπάγεται πολυωνυμικό αλγόριθμο για το παρακάτω NP-πλήρες πρόβλημα.

Ας υποθέσουμε ότι υπάρχει ρ-προσεγγιστικός αλγόριθμος Α πολυωνυμικού χρόνου για το γενικό πρόβλημα του περιοδεύοντος πωλητή, όπου $\rho \geq 1$ ακέραια σταθερά. Μπορούμε να χρησιμοποιήσουμε τον Α για να βρούμε σε πολυωνυμικό χρόνο αν υπάρχει κύκλος Hamilton σε ένα γράφημα G = (V,E). Έστω G' = (V,E') το πλήρες γράφημα με σύνολο κόμβων V ορίζουμε τη συνάρτηση κόστους c : E' -> Z των ακμών του G'ως :

$$c(u,v) = \begin{cases} 1, & \{u,v\} \in E \\ \rho|V| + 1, & \{u,v\} \notin E \end{cases}$$

Αν το G έχει κύκλο Hamilton τότε η βέλτιστη περιοδεία H* του G' έχει κόστος:

$$c(H^*) = |V|.$$

Διαφορετικά η H* περιέχει τουλάχιστον μια ακμή εκτός Ε, άρα:

$$c(H^*) \geq |V|-1 + \rho|V| +1 = \rho|V| + |V| = (\rho +1)\,|V|.$$

Επομένως δεν μπορούμε να αποφασίσουμε σε πολυωνυμικό χρόνο αν το G περιέχει κύκλο Hamilton.

Κεφάλαιο 5. Συμπεράσματα

Με την παραπάνω μελέτη, αρχικά αναλύσαμε του βασικούς ορισμούς όπως τι είναι ο αλγόριθμος και ορισμούς γραφών καθώς και τις κλάσεις. Και πιο συγκεκριμένα δώσαμε έναν ορισμό στο πρόβλημα του πλανόδιου πωλητή και αναφερθήκαμε στην κλάση που ανήκει.

Για τους κύκλους Hamilton χρησιμοποιήσαμε έναν γενικό αλγόριθμο και στη συνέχεια έναν άπληστο αλγόριθμο.

Για τον αλγόριθμο του Prim παρατηρήσαμε πως είναι αποδοτικός όχι όμως όταν πρόκειται να χρησιμοποιήσουμε εξωτερική μνήμη, κυρίως επειδή την τρέχουσα προτεραιότητα της εκάστοτε κορυφής δεν μπορούμε γενικά να την εξασφαλίσουμε χωρίς να κάνουμε μία πράξη I/O. Με αυτό τον τρόπο μία τέτοια υλοποίηση θα οδηγούσε σε έναν Ω(E) I/O αλγόριθμο. Και έτσι καταλήξαμε σε έναν αλγόριθμο όπου το ελάχιστο επικαλύπτον δέντρο ενός μη κατευθυνόμενου βεβαρημένου γραφήματος G = (V,E) μπορεί να υπολογιστεί σε O(V + sort(E)) I/Os.

Τέλος, αναπτύξαμε το πρόβλημα του πλανόδιου πωλητή και καταλήξαμε στο συμπέρασμα ότι ο πιο αποδοτικός αλγόριθμος είναι ο προσεγγιστικός αλγόριθμος που χρησιμοποιεί ένα ελάχιστο συνδετικό δένδρο για την επίλυση του.

Βιβλιογραφία

1. **Beauquier, Berstel, Chrétienne, 1992,** *Στοιχεία αλγορίθμων*, Masson, Παρίσι.

2. **Grochemore, Rytter, 1994,** *Αλγόριθμοι κειμένου*, Oxford University Press.

3. **Stephen, GA, 1994,** *String Αλγόριθμοι Αναζήτησης*, World Scientific.

4. **Idan Szpektor, 2011,** *Boyer Moore Algorithm*, ppt Παρουσίαση.

5. **Ο. Καλαμιώτης, 2008,** *Τεχνικές αναζήτησης προτύπων σε αρχεία κειμένου*, Βόλος.

6. **Δ. Λιώλη, 2008,** *Αλγόριθμοι γραφημάτων στην εξωτερική μνήμη*, Βόλος.

7. **Σ. Ζάχος, Δ. Φωτάκης, 2009,** *Ελάχιστο Συνδετικό Δέντρο*, Εθνικό Μετσόβιο Πολυτεχνείο.

8. **Γ. Βούρος, 2006,** *Συνδετικά Δέντρα*, Πανεπιστήμιο Αιγαίου.

9. **Τ. Δημητρίου, 2001,** *Παράλληλοι Αλγόριθμοι κ Software,Minimum Spanning Trees*, Πάτρα.

10. **Π. Εφραιμίδης, 2010,** *Θεωρία Γραφημάτων*, Ξάνθη.

11. **Γ. Δατσέρης,** *Δυναμικός προγραμματισμός και εφαρμογές*, Κρήτη.

12. **Ε. Πασσάλη, 2006,** *Καλά επιλύσιμες περιπτώσεις για το πρόβλημα του Περιοδεύοντος Πωλητή*, Πάτρα.

13. **Φ. Θεός, 2001,** *Μέθοδοι ολικής ελαχιστοποίησης*, Ιωάννινα.

14. **Α. Μπαλούκας, 2007,** *Στατική και δυναμική οπτικοποίηση αλγορίθμων δικτύων*, Θεσσαλονίκη.

15. **Σ. Δημητρίου,** *Επεξεργασία και υλοποίηση Αλγορίθμων για την επίλυση προβλημάτων ζευγνυόντων δέντρων (spanning trees).* Κρήτη.

16. **Λ. Ιερεμιάδη, 2009,** *Προσεγγιστικοί αλγόριθμοι για το πρόβλημα της χρονοδρομολόγησης αποστολής δεδομένων σε δίκτυα*, Εθνικό Μετσόβιο πολυτεχνείο.

17. **Ν. Ουλκέρογλου , 2004,** *Βελτιστοποίηση προγραμματισμού παραγωγής με χρήση γενετικών αλγορίθμων,* Εθνικό Μετσόβιο πολυτεχνείο.

18. **Π. Κατσαρός, 2013,** *Άπληστοι ευριστικοί αλγόριθμοι* ,Θεσσαλονίκη.

19. **Χ. Παπαδοπούλου, 2011,** *Επεξεργασία και αναπαράσταση βέλτιστων Διαδρομών σε οδικά δίκτυα*, Εθνικό Μετσόβιο πολυτεχνείο.

20. http://en.wikipedia.org/wiki/Travelling_salesman_problem

21.**Ε. Πετράκης**, *Αλγόριθμοι και Πολυπλοκότητα.*

22. **Π. Εφραιμίδης, 2010,** *Κλάσεις πολυπλοκότητας*, Ξάνθη.