Travelling Salesman Problem

Σιώρος Βασίλειος

Ανδρινοπούλου Χριστίνα

Μάϊος 2020

Contents

Chapter 1

Abstract

Chapter 2

Introduction

Το "Travelling Salesman Problem" (TSP) ή με την ελληνική του απόδοση "Πρόβλημα του πλανόδιου πωλητή" (ή εναλλακτικά πρόβλημα του περιοδεύοντος πωλητή) είναι ένα κλασσικό πρόβλημα θεωρητικής επιστήμης των υπολογιστών. Πρόκειται για ένα πρόβλημα περιήγησης. Ο πωλητής οφείλει να επισκευτεί n το πλήθος πόλεις για να πουλήσει το εμπόρευμά του. Σκοπός του προβλήματος είναι η εύρεση μίας βέλτιστης διαδρομής για τον πωλητή, με την οποία θα μπορέσει να επισκεφτεί όλες τις πόλεις που τον ενδιαφέρουν, μόνο μία φορά την κάθε μία και μάλιστα με τέτοιον τρόπο ώστε να διανύσει τη μικρότερη δυνατή απόσταση. Με άλλα λόγια, ο πωλητής πρέπει να επισκεφτεί την κάθε πόλη ακριβώς μία φορά ακολουθώντας το συντομότερο δρομολόγιο.

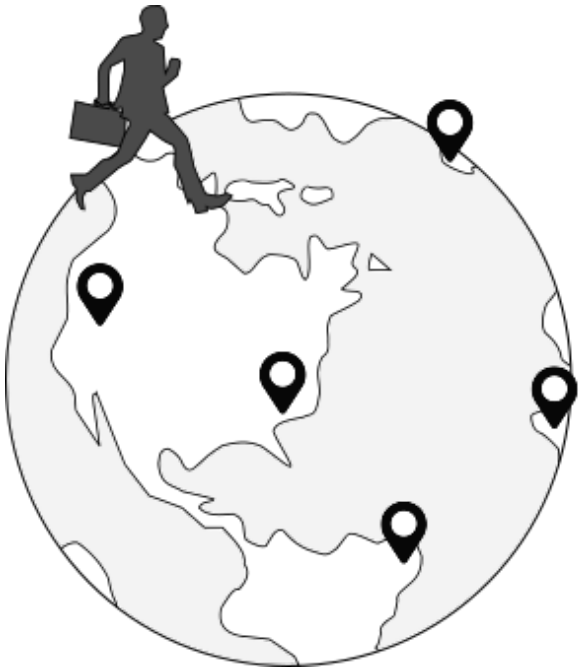

Figure 2.1: Travelling Salesman Problem - TSP
πηγή: https://www.localsolver.com/docs/last/exampletour/tsp.html1

Chapter 3

Εφαρμογές του προβλήματος του περιοδεύοντος πωλητή

Το TSP είναι ένα πολύ σημαντικό πρόβλημα στην επιστήμη της πληροφορικής, καθώς έχει μία γκάμα εφαρμογών.

Εφαρμογές που σχετίζονται με τις μεταφορές και το logistics μπορούν άμεσα να επωφεληθούν από τα ευρήματα και τη μελέτη γύρω από το TSP. Τέτοιες εφαρμογές είναι η μετακίνηση των μηχανών εφοδιασμού στους ορόφους καταστημάτων ή σε αποθήκες, η δρομολόγηση φορτηγών για παραλαβή δεμάτων, η παράδοση τροφίμων σε άτομα που δεν μπορούν να μετακινηθούν από το σπίτι, ο καθορισμός των δρομολογίων των σχολικών λεωφορείων. Μάλιστα, η τελευταία υπήρξε και η αφορμή για περεταίρω μελέτη του προβλήματος του περιοδεύοντος πωλητή το 1940 από τον Merrill Flood, μελέτη που θεωρήθηκε σταθμός για το TSP.

Ακόμα υπάρχουν εφαμογές του TSP σε διάφορους επιστημονικούς κλάδους. Στη βιολογία, το πρόβλημα του πλανόδιου πωλητή χρησιμοποιήθηκε για DNA sequencing. Με τον όρο DNA sequencing οι βιολόγοι καλούν την διαδικασία καθορισμού της σειρά των νουκλεοτιδίων στο DNA. Στην περίπτωση αυτή οι "πόλεις" του TSP είναι DNA strings, ενώ οι αποστάσεις μεταξύ των DNA strirngs υπολογίζονται με βάση μέτρα σημασιολογικής ομοιότητας (semantic similarity measures). Τα μέτρα αυτά, εν γένει, εκτιμούν την ομοιότητα δύο βιολογικών αντικειμένων.

Κάθε οντότητα κατέχει ένα συγκεκριμένο βιολογικό ρόλο, ο οποίος αποτελεί τη σημασιολογία της.

Άλλος κλάδος όπου το TSP συνέβαλε είναι η αστρονομία και το διάστημα. Στην περίπτωση αυτήν, οι επιστήμονες ήθελαν να περιορίσουν όδο είναι δυνατόν τα καύσιμα που απαιτούνται για την παρατήρηση και καταγραφή ουράνιων αντικειμένων.

Φυσικά, αυτές είναι κάποιες ενδεικτικές εφαρμογές του προβλήματος του περιοδεύοντος πωλητή. Τις επισημάνουμε καθώς στάθηκαν για εμάς κίνητρο μελέτης του προβλήματος.

Chapter 4

Πολυπλοκότητα **TSP**

Στην ενότητα αυτή θα δούμε μία βασική θεωρία γύρω από την πολυπλοκότητα των αλγορίθμων γενικά και έπειτα θα εξειδικεύσουμε στον TSP αλγόριθμο.

Ένας "εύκολος" και αποδοτικός αλγόριθμος είναι ένας αλγόριθμος όπου για μέγεθος εισόδου n έχει χρόνο εκτέλεσης χειρότερης περίπτωσης $O(n^k)$, όπου k κάποια σταθερά. Ωστόσο, δεν είναι όλα τα προβήματα ίδια. Υπάρχουν προβλήματα για τα οποία δεν έχει ακόμα ανακαλυφθεί αλγόριθμος που να τα επιλύει σε πολυωνυμικό χρόνο. Όμως, υπάρχουν αλγόριθμοι οι οποίοι αν γνωρίζουν μία πιθανή λύση των προβλημάτων αυτών μπορούν να την επαληθεύσουν σε πολυωνυμικό χρόνο. Η πρώτη κατηγορία ποβλημάτων, τα οποία μπορούν να επιλυθούν σε πολυωνυμικό χρόνο, ανήκει στην γνωσή κλάση P, ενώ η δεύτερη κατηγορία στην κλάση NP. Είναι προφανές πως αν ένα πρόβλημα ανήκει στην κλάση P, τότε θα ανήκει και στην κλάση NP, γιατί ένα πρόβλημα που ανήκει στην P μπορεί να επαληθευτεί σε πολυωνυμικό χρόνο με δεδομένη κάποια λύση. Συνεπώς, $P \subseteq NP$. Ωστόσο, δε γνωρίζουμε αν $NP \subseteq P$. Για να είμαστε πιο ακριβείς ούτε έχει επιβεβαιωθεί, ούτε έχει καταρριφθεί κάτι τέτοιο.

Το εντυπωσιακό στην περίπτωση των NP προβλημάτων είναι ότι πολλές φορές μπορεί να μοιάζουν εντυπωσιακά πολύ με κάποιο πρόβλημα που ανήκει στην κλάση P. Για παράδειγμα, δεδομένου ενός γράφου $G = (V, E)$, η εύρεση ενός κύκλου Euler, δηλαδή ενός κύκλου που περνά από κάθε ακμή του γράφου ακριβώς μία φορά χωρίς να υπάρχει περιορισμός στο πλήθος των

επισκέψεων ανά κόμβο έχει πολυπλοκότητα $O(|E|)$. Να σημειώσουμε εδώ ότι ο παραπάνω κύκλος καλείται κύκλος **Euler**. Αντιθέτως, η εύρεση αν ένα γράφημα περιέχει κύκλο **Hamilton**, δηλαδή κύκλο που διέρχεται από κάθε κόμβο του γραφήματος ακριβώς μία φορά, είναι NP-πλήρες πρόβλημα. (Για περισσότερες πληροφορίες σχετικά με τα γραφήματα και τους κύκλους **Hamilton** μπορεί κανείς να ανατρέξει στην υποενότητα "Γραφήματα" της ενότητας "Μαθηματικό υπόβαθρο").

Λήμμα 4.0.1: TSP

Το πρόβλημα του περιοδεύοντος πωλητή ανήκει στην κλάση NP.

Απόδειξη:

Για να αποδείξουμε ότι το TSP ανήκει στην κλάση NP αρκεί να βρούμε έναν αλγόριθμο, ο οποίος να επαληθεύει μία δεδομένη λύση σε πολυωνυμικό χρόνο. Έστω γράφημα $G = (V, E)$ και k να είναι ένα **threshold** για την απόσταση που μπορεί να διανύσει ο πωλητής στην περίπτωση της βέλτιστης διαδρομής. Αν μία προτεινόμενη λύση είναι ένας κύκλος h, τότε ο πολυωνυμικός αλγόριθμος που

επαληθεύει μία λύση είναι ο παρακάτω.

Algorithm 1: Επαλήθευση λύσης

Input: Γράφημα $G = (V, E)$, αριθμός k και κύκλος h

Result: Απαντά ΝΑΙ στην περίπτωση που το h επαληθεύει λύση, αλλιώς απαντά ΟΧΙ

```
if h είναι Hamiltonian κύκλος του G then
    sum = Υπολόγισε το άθροισμα των βαρών του h ;
    if sum ≤ k then
        Τύπωσε ΝΑΙ ;
    end
    else
        Τύπωσε ΟΧΙ ;
    end
end
else
    Τύπωσε ΟΧΙ ;
end
```

Συνεπώς, το TSP είναι πρόβλημα που ανήκει στην κλάση NP.

Ένα σύνολο των προβλημάτων που ανήκουν στην κλάση NP καλούνται NP-complete και συγκροτούν ένα σύνολο προβλήματων NP. Πιο συγκεκριμένα, ένα πρόβλημα καλείται NP-complete αν ανήκει στην κλάση NP και κάθε άλλο πρόβλημα που ανήκει στην κλάση NP ανάγεται σε αυτό. Στο σημείο αυτό θα αποδείξουμε ότι το TSP είναι NP-complete.

Λήμμα **4.0.2: TSP**

Το πρόβλημα του περιοδεύοντος πωλητή είναι NP-complete.

Απόδειξη:

Για να αποδείξουμε ότι το TSP είναι NP-complete πρέπει να αποδείξουμε ότι ανήκει στην κλάση NP και έπειτα να κάνουμε αναγωγή από ένα ήδη γνωστό NP-complete πρόβλημα.

Το πρώτο βήμα είναι εύκολο. Όπως αποδείχθηκε στο Λήμμα 4.0.1 το TSP ανήκει στην κλάση

NP.

Για το δεύτερο βήμα επικαλούμαστε το πρόβλημα ύπαρξης κύκλου Hamilton σε ένα γράφημα, το οποίο είναι NP-complete. Θα ανάγουμε το πρόβλημα αυτό στο πρόβλημα του περιοδεύοντος πωλητή.

Έστω ότι έχουμε ένα στιγμιότυπο του προβλήματος εύρεσης κύκλου Hamilton και το γράφημα $G = (V, E)$. Θα κατασκευάσουμε ένα καινούριο γράφημα $G' = (V', E')$ όπου $V' = V$ και E' θα περιλαμβάνει όλες τις δυνατές ακμές στο γράφημα. Κάθε ακμή στο G' θα έχει βάρος 1 αν είναι ακμή που υπήρχε και στο G, αλλιώς θα έχει βάρος 2. Η κατασκευή ενός γραφήματος G' γίνεται σαρώνοντας το G και προσθέτοντας τις ακμές που λείπουν με κατάλληλα βάρη. Συνεπώς, η διαδικασία αυτή έχει πολυωνυμική πολυπλοκότητα.

Ως threshold της απόστασης του μονοπατιού που θα διανύσει ο πωλητής θέτουμε το $|V|$.

Αν το G έχει Hamiltonian κύκλο, τότε και το G' θα έχει, αφού αποτελείται από τις ίδιες κορυφές και $E \subseteq E'$. Ο κύκλος θα έχει άθροισμα βαρών ακμών ίσο με $|V|$. Αν το G' έχει Hamilton κύκλο με άθροισμα βαρών ακμών ίσο με $|V|$, τότε ο κύκλος αυτός απαρτίζεται μόνο από τις κορυφές που έχουν βάρος 1, δηλαδή τις κορυφές που υπάρχουν και στο γράφημα G. Συνεπώς, και το G έχει κύκλο Hamilton.

Συνεπώς, το TSP είναι NP-complete, γιατί είναι τόσο δύσκολο όσο το πρόβλημα ύπαρξης Hamilton κύκλου.

Στο σημείο αυτό κρίνεται σκόπιμο να αναφέρουμε ότι ο αποδοτικότερος τρόπος να μεταβεί κάποιος από ένα σημείο x σε ένα σημείο y είναι μεταβαίνοντας απευθείας στο y χωρίς ενδιάμεσους σταθμούς. Συνεπώς, η εξάλειψη των ενδιάμεσων σταθμών αποκλείεται να αυξήσει το κόστος της διαδρομής. Αυτό είναι μία ιδιότητα που αποτυπώνεται στην τριγωνική ανισότητα.

$$w(x, y) \leq w(x, z) + w(z, y)$$

όπου $w(x, y)$ το κόστος μετάβασης από το σημείο x στο σημείο y.

Αν οι πόλεις που πρόκειται να επισκεφτεί ο πλανόδιος πωλητής είναι σημεία στο επίπεδο και η απόσταση μετράται με την Ευκλείδια απόσταση, τότε η τριγωνική ανισότητα ικανοποιείται και υπό αυτές τις συνθήκες το TSP είναι NP-complete, με άλλα λόγια δεν περιμένουμε αλγόριθμο που να επιλύει το πρόβλημα σε πολυωνυμικό χρόνο , αλλά κάποιον προσεγγιστικό αλγόριθμο.

Ένας προσεγγιστικός αλγόριθμος υπολογίζει σχεδόν βέλτιστες λύσεις και συνήθως αυτές αρκούν για τα NP-complete προβλήματα. Ένας προσεγγιστικός αλγόριθμος επιτυγχάνει λόγο προσέγγισης $\rho(n)$, όπου n το μέγεθος της εισόδου του αλγορίθμου. Για τον $\rho(n)$) ισχύει ότι

$$\max\left(\frac{C}{C_{OPT}}, \frac{C_{OPT}}{C}\right) \leq \rho(n)$$

όπου C και C_{OPT} το κόστος της λύσης του προσεγγιστικού αλγορίθμου και της βέλτιστης λύσης αντίστοιχα. Ένας αλγόριθμος που επιτυγχάνει λόγο προσέγγισης $\rho(n)$ καλείται $\rho(n)$-προσεγγιστικός.

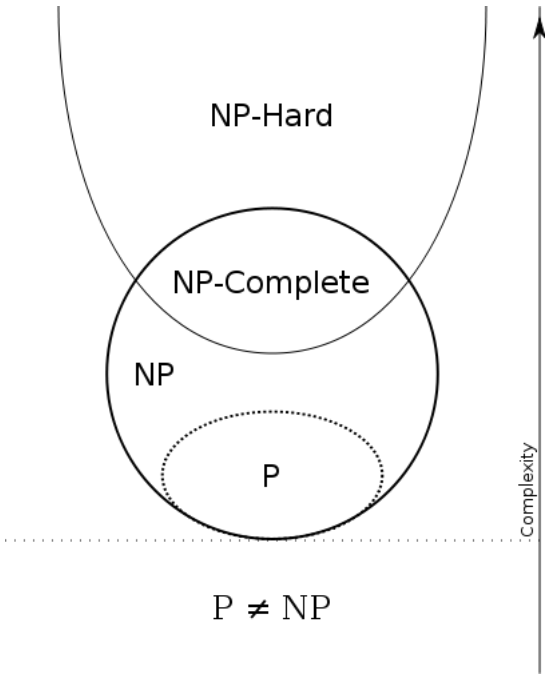

Figure 4.1: P κλάση και NP κλάση
πηγή: https://jcaip.github.io/Reduction-and-Intractibility/

Chapter 5

Μαθηματικό υπόβαθρο

Η μελέτη σε βάθος του προβλήματος του περιοδεύοντος πωλητή χρήζει απαραίτητη την γνώση μερικών βασικών μαθηματικών εννοιών. Στο κεφάλαιο αυτό, παρουσιάζουμε και αναλύυμε, στον βαθμό που κρίνεται απαραίτητο, όλες τις μαθηματικές γνώσεις που χρειάζονται για την κατανοήση του TSP.

5.1 Γραφήματα

Βασική έννοια για τη μελέτη του προβήματος του περιοδέυοντος πωλητή είναι τα γραφήματα. Τα γραφήματα είναι ένα πολύ σημαντικά εργαλείο στα χερια των θεωρητικών πληροφορικών. Προσφέρουν πολλές διευκολύνσεις κατά τη μελέτη προβλημάτων (όπως και στην περίπτωση μας), είναι σχετικά απλά στη μελέτη και την κατανόηση τους και μπορούν εύκολα να κωδικοποιηθούν και να θεμελιωθούν με αυστηρό, μαθηματικό τρόπο.

5.1.1 Βασική ορολογία

Τα βασικά συστατικά ενός γραφήματος είναι οι κορυφές και οι ακμές. Οι κορυφές ενώνονται με τη βοήθεια των ακμών και δημιουργούν ένα γράφημα.

Τα γραφήματα μπορούν να διακριθούν σε δύο βασικές κατηγορίες, τα κατευθυνόμενα γραφήματα και τη μη κατευθυνόμενα.

Ο αφηρημένος ορισμός στην περίπτωση των κατευθυνόμενων γραφημάτων περιλαμβάνει ένα διατεταγμένο ζεύγος (V, E), όπου V είναι το σύνολο και το E είναι μία διμελής σχέση. Το V αποτελεί το σύνολο των κορυφών και το E το σύνολο των ακμών. Το κατευθυνόμενο γράφημα συμβολίζεται με G. Ένα κατευθυνόμενο γράφημα μπορεί να αναπαρασταθεί γεωμετρικά ως ένα σύνολο από V σημεία, τα οποία ενώνονται με E βέλη. Ενα παραδειγμα γράφου φαίνεται στην εικόνα παρακάτω.

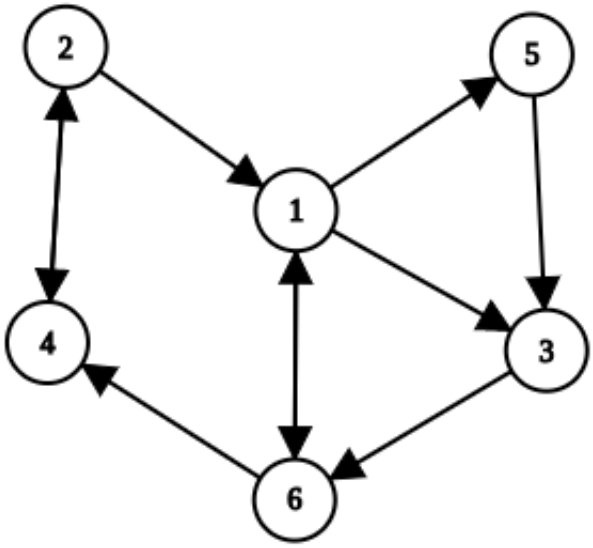

Figure 5.1: Παράδειγμα κατευθυνόμενου γράφου
κατασκευάστηκε με: https://csacademy.com/app/graph_editor/

Ο ορισμός για το μη κατευθυνόμενο γράφημα περιλαμβάνει ένα σύνολο V και ένα σύνολο πολυσυνόλων δύο στοιχείων E. Το V αποτελεί και σε αυτήν την περιπτωση το σύνολο των κορυφών και το E το σύνολο των ακμών. Μία ενδεικτική γεμετρική αναπαράσταση δίνεται στην αντίστοιχη

εικόνα παρακάτω. Και σε αυτήν την περίπτωση το V είναι ένα σύνολο σημείων, ωστόσο το E είναι ένα σύνολο γραμμών που δεν υποδικνύουν καμμία κατεύθυνση.

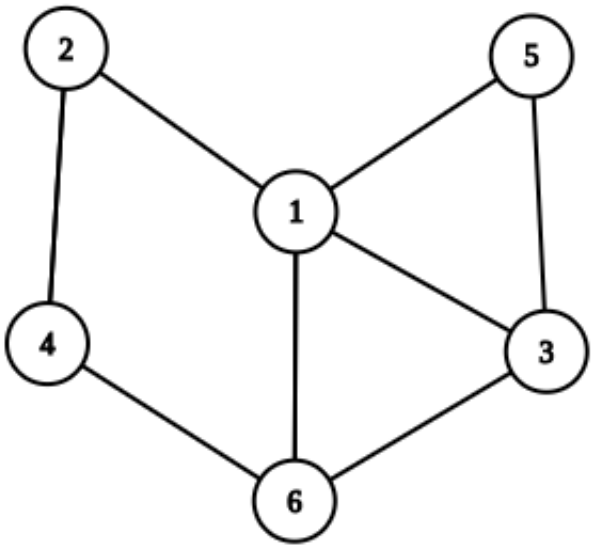

Figure 5.2: Παράδειγμα μη κατευθυνόμενου γράφου
κατασκευάστηκε με: https://csacademy.com/app/graph_editor/

5.1.2 Μονοπάτια

Σε κάθε κατευθυνόμενο γράφημα οι ακμές περιέχουν μία αρχική κορυφή και μία τερματική κορυφή. Για παράδειγμα η ακμή (1,3) της εικόνας 3.1 περιέχει δύο κορυφές. Η κορυφή 1 καλείται αρχική κορυφή και η κορυφή 3 καλείται τερματική κορυφή.

Στα κατευθυνόμενα γραφήματα, μία ακολουθία ακμών $(e_1, e_2, ..., e_k)$ όπου η τερματική κορυφή μίας ακμής e_j ταυτίζεται με την αρχική κορυφή της $e_{(j+1)}$ καλείται μονοπάτι.

Αν ένα μονοπάτι δεν περιέχει την ίδια ακμή δύο φορές ονομάζεται απλό μονοπάτι.

Στοιχειώδες μονοποάτι καλείται το μονοπάτι εκείνο όπου δεν περιέχει την ίδια κορυφή παρα-

πάνω από μία φορά.

Κύκλωμα καλείται το μονοπάτι $(e_1, e_2, ..., e_k)$, όπου η τερματική κορυφή της e_k ακμής συμπίπτει με την αρχική κορυφή της e_1 ακμής.

5.1.3 Χαμιλτόνειοι κύκλοι και μονοπατια

Ο Ιρλανδός φυσικός και μαθηματικός William Rowan Hamilton (4 Αυγούστου 1805 – 2 Σεπτεμβρίου 1865) μελέτησε εκτός των άλλων και τα γραφήματα. Συγκεκριμένα, δημιούργησε το μαθηματικό παιχνίδι "ο γύρος του κόσμου", του οποίου σκοπός ήταν η εύρεση ενός μονοπατιού από ακμές δωδεκαέδρου. Το μονοπάτι έπρεπε να περνά από κάθε κορυφή του δωδεκαέδρου ακριβώς μία φορά. Πιο συγκεκριμένα, ο ένας παίκτης κάρφωνε από μία βελόνα σε 5 διαδοχικές κορυφές και έπειτα ο άλλος παίκτης έπρεπε να συμπληρώσει το κύκλωμα έτσι ώστε να περιλαμβάνει όλες τις κορυφές. Μάλιστα, σε επιστολή προς τον φίλο του John T. Graves (17 Οκτωβρίου 1856) ο Hamilton τον πληροφορεί σχετικά με ένα παιχνίδι που βασίζεται στον εικοσιανό λογισμό (αλγεβρική δομή για τον υπολογισμών συμμετριών του εικοσαέδρου από τον ίδιο τον Hamilton) και την αναγνωρισιμότητα που έχει λάβει από μερικούς νεαρούς. Το παιχνίδι αυτό στάθηκε η αφορμή για την ανάπτυξη της θεωρίας γύρω από τα γραφήματα και προς τιμήν του Hamilton και του εν λόγω παιχνιδιού που επινόησε οι Χαμιλτονειανοί κύκλοι έλαβαν το όνομά του.

Figure 5.3: William Rowan Hamilton
πηγή: https://en.wikipedia.org/wiki/William_Rowan_Hamilton

Η εύρεση ενός μονοπατιού ή ενός κυκλώματος που περνά από κάθε κορυφή ενός δεδομένου

γραφήματος μόνο μία φορά φαντάζει αρχικά απλή υπόθεση, ωστόσο οι μέχρις στιγμής επιστημονικές προσεγγίσεις αποδεκνύουν το αντίθετο.

Μονοπάτι (ή κύκλωμα) **Hamilton** είναι ένα μονοπάτι (ή ένα κύκλωμα) που περνά από όλες τις κορυφές ενός γραφήματος ακριβώς μία φορά.

Ένα γράφημα που περιέχει κύκλο **Hamilton** καλείται χαμιλτόνειο, ενώ αν δεν περιέχει καλείται μη χαμιλτόνειο.

Δυστυχώς, μέχρι και τώρα δε γνωρίζουμε κάποια ικανή και αναγκάια συνθήκη για την ύπαρξη μονοπατιών και κυκλωμάτων **Hamilton**. Δοθέντος ενός γραφήματος G, η απόδειξη ύπαρξης ή μη ενός μονοπατιού ή κυκλώματος **Hamilton** είναι η κατασκευή του.

5.2 **Delaunay** Τριγωνοποίηση

Στη υπολογιστική γεωμετρία, η τριγωνοποίηση πολυγώνων είναι ένα βασικό ζήτημα μελέτης. Με το όρο τριγωνοποίηση εννοούμε την διαμέριση της περιοχής που ορίζεται από το πολύγωνο σε τρίγωνα των οποίων η ένωση παράγει την περιοχή του αρχικού πολυγώνου.

Ουσιαστικά, η τριγωνοποίηση ενός συνόλου σημείων στο επίπεδο είναι ένα σύνολο τριγώνων. Η ένωση των τριγώνων αυτών ισούται με το κυρτό περίβλημα των σημείων και η τομή δύο τριγώνων μπορεί να είναι είτε κενή, είτε να είναι ίση με την κοινή κορυφή των δύο τριγώνων ή με την κοινή τους ακμή.

Στην παρούσα υποενότητα θα αναφερθούμε στη **Delaunay** τριγωνοποίηση, καθώς θα μας φανεί χρήσιμη στη μελέτη του προβλήματος του περιοδεύοντος πωλητή, με τρόπο που περιγράφεται σε επόμενη ενότητα. Επισημαίνουμε τη βασική θεωρία γύρω από την εν λόγω τριγωνοποίηση και μερικούς θεμελιώδεις αλγορίθμους που την παράγουν.

5.2.1 Ιστορία

Η delaunay τριγωνοποίηση ήταν μία επινόηση του Ρώσου μαθηματικού Boris Nikolaevich Delone. Ο Delone γεννήθηκε στις 15 Μαρτίου του 1890 και πέθανε έπειτα από 90 χρόνια, στις 17 Ιουλίου του 1980. Ο Boris Delone ασχολήθηκε με την άλγεβρα και τη γεωμετρία και μία από τις σημαντικότερες ανακαλύψεις του ήταν η τριγωνοποίηση Delaunay το 1934.

Η τριγωνοποίηση έλαβε το όνομα Delaunay, προς τιμήν του δημιουργού της, ο οποίος καλούσε τον εαυτό του "Boris Nikolaeviq Delone". Καθώς την εποχή εκείνη οι δύο επικρατέστερες γλώσσες στους επιστημονικούς κόλπους ήταν τα γαλλικά και τα γερμανικά, επικράτησε η γαλλική εκδοχή του ονόματός του και έτσι η τριγωνοποίηση έλαβε τελικά το όνομα Delaunay.

5.2.2 Βασική θεωρία

Στόχος της τριγωνοποίησης Delaunay είναι με βάση ένα σύνολο σημείων στο επίπεδο να παράξει μία τριγωνοποίηση αυτών, η οποία να ικανοποιεί ορισμένες συνθήκες.

Για να μελετήσουμε την τριγωνοποίηση Delaunay πρέπει εξ αρχής να κάνουμε την παραδοχή ότι τα σημεία που πρόκειται να τριγωνοποιηθούν βρίσκονται σε γενική θέση. Γενική θέση στην παρούσα ενότητα εννοούμε ότι τέσσερα σημεία δε βρίσκονται πάνω στον ίδιο κύκλο.

Αρχικά, εισάγουμε την απαραίτητη έννοια του "διανύσματος γωνιών" (vector angle). Έστω ότι $P := \{p_1, p_2, ..., p_n\}$ είνα ένα σύνολο n σημείων στο επίπεδο. Επίσης, έστω ότι T είναι μια τριγωνοποίησή τους. Αν m είναι το σύνολο των τριγώνων που έχουν παραχθεί στη συγκεκριμένη τριγωνοποίηση, τότε με προφανή τρόπο εξάγεται το συμπέρασμα ότι το σύνολο των γωνιών που περιέχει η τριγωνοποίηση είναι $3m$. Τοποθετούμε τις γωνίες της τριγωνοποίησης σε ένα διάνυσμα

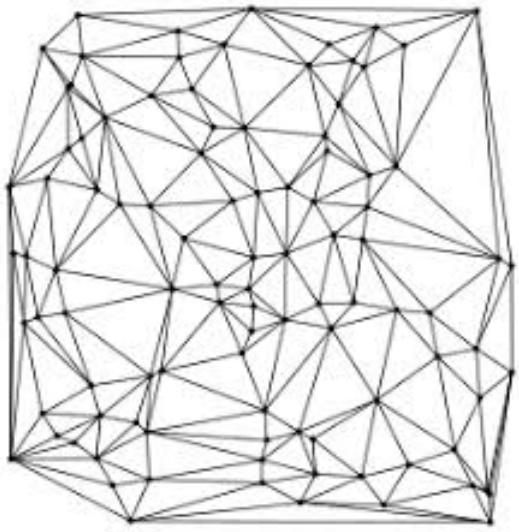

Figure 5.4: Τριγωνοποίηση Delaunay
πηγή: https://en.wikipedia.org/wiki/Delaunay_triangulation

Figure 5.5: Boris Nikolaevich Delone
πηγή: https://en.wikipedia.org/wiki/File:Delone_cropped_1924.jpg

σε αύξουσα σειρά. Το διάνυσμα αυτό καλείται **vector-angle** και ορίζεται ως εξής

$$A(T) := (a_1, a_2, ..., a_{3m}) \tag{5.1}$$

όπου

$$a_i \leq a_j, \forall i < j \tag{5.2}$$

Αν T' είναι μια διαφορετική τριγωνοποίηση από την T για το ίδιο σύνολο σημείων P και το αντίστοιχο διάνυσμα γωνιών είναι το $A(T') = (a'_1, a'_2, ..., a'_{3m})$, θα λέμε ότι το διάνυσμα γωνιών της T τριγωνοποίησης είναι μεγαλύτερο από το διάνυσμα γωνιών της T' τριγωνοποίησης και θα γράφουμε $A(T) > A(T')$ αν υπάρχει i, όπου $1 \leq i \leq 3m$ τέτοιο ώστε

$$a_j = a'_j, \forall j < i \text{ και } a_i > a'_i$$

Θα δούμε στη συνέχεια ότι στόχος μας εδώ είναι να καταφέρουμε να εντοπίσουμε το μεγαλύτερο διάνυσμα γωνιών.

Θα λέμε ότι μία τριγωνοποίηση είναι "**angle-optimal**" και θα γράφουμε $A(T) \geq A(T')$ αν $A(T) > A(T')$ για όλες τις δυνατές τριγωνοποιήσεις T'.

Στο σημείο αυτό πρέπει να αναφέρουμε ότι το πλήθος των τριγώνων που προκύπτουν από μία τριγωνοποίηση δεν είναι τυχαίο. Μπορεί να καθοριστεί με ακρίβεια, πράγμα που ήδη έχει διατυπωθεί σε αντίστοιχο θεώρημα.

Θεώρημα 5.2.1: Πλήθος ακμών και τριγώνων τριγωνοποίησης σημείων στο επίπεδο

Έστω P ένα σύνολο n σημείων στο επίπεδο. Έστω ότι τα σημεία δεν είναι όλα μεταξύ τους συνευθειακά και έστω με k να συμβολίζονται ότα τα σημεία που βρίσκονται στο όριο του convex hull του P. Οποιαδήποτε τριγωνοποίηση των σημείων του P παράγει $2n - 2 - k$ τρίγωνα και $3n - 3 - k$ ακμές.

Απόδειξη:

Έστω ότι τα P σημεία τριγωνοποιούνται και παράγονται m το πλήθος τρίγωνα. Συνεπώς, το επίπεδο έχει διαχωριστεί με $m + 1$ υποπεριοχές.

Κάθε τρίγωνο έχει 3 ακμές και το convex hull έχει k ακμές.

Κάθε ακμή ανήκει σε δύο υποπεριοχές του επιπέδου.

Συνεπώς, ο συνολικός αριθμός ακμών είναι $\frac{3m+k}{2}$.

Από τον τύπο του Euler, όπου $n_f = m + 1$ και $n_e = \frac{3m+k}{2}$, προκύπτει ότι

$$
\begin{aligned}
n - n_e + n_f = 2 \Leftrightarrow \\
n - \frac{3m + k}{2} + (m + 1) = 2 \Leftrightarrow \\
2n - (3m + k) + 2(m + 1) = 4 \Leftrightarrow \\
2n - 3m - k + 2m + 2 = 4 \Leftrightarrow \\
2n - m - k = 2 \Leftrightarrow \\
m = 2n - k - 2
\end{aligned}
$$

και αφού $m = 2n - 2 - k$, το n_e υπολογίζεται να είναι

$$n_e = \frac{3m + k}{2} \Leftrightarrow$$
$$n_e = \frac{3(2n - k - 2) + k}{2} \Leftrightarrow$$
$$n_e = \frac{6n - 3k - 6 + k}{2} \Leftrightarrow$$
$$n_e = \frac{6n - 6 - 2k}{2} \Leftrightarrow$$
$$n_e = 3n - 3 - k$$

□

Όπως ήδη αναφέραμε στόχος είναι να βρεθεί η τριγωνοποίηση που δίνει το "μεγαλύτερο" διάνυσμα γωνιών. Για να καταφέρουμε να φτάσουμε στο σημείο να παράξουμε μία τέτοια τριγωνοποίηση για ένα δεδομένο σύνολο σημείων πρέπει να αναφέρουμε μερικές ακόμα θεμελιώδεις έννοιες. Εισάγουμε την έννοια της "παράνομης ακμής" (illegal edge).

Έστω μία τριγωνοποίηση των P σημείων και έστω μία ακμή της τριγωνοποίησης που βρίσκεται εντός ενός κυρτού τετράπλευρου και είναι κοινή ακμή για δύο διαφορετικά τρίγωνα της τριγωνοποίησης. Η ακμή αυτή μπορέι να γίνει flip. Στην περίπτωση αυτή το μόνο που αλλάζει είναι το $A(T)$.

Ορισμός 5.2.1: Παράνομη ακμή

Έστω T μία τριγωνοποίηση των σημείων P. Καλούμε μία ακμή "παράνομη" εαν μπορούμε να αυξήσουμε τη μικρότερη γωνία του διανύσματος γωνιών κάνοντάς την flip.

Αν T μία τριγωνοποίηση που περιέχει μία παράνομη ακμή και T' η τριγωνοποίηση που έχει γίνει flip η παράνομη ακμή, τότε ισχύει ότι η T' είναι angle-optimal:

$$A(T') \geq A(T)$$

Ορισμός **5.2.2**

Μία τριγωνοποίηση T καλείται νόμιμη εαν δεν περιέχει καμία παράνομη ακμή.

Συνεπώς, κάθε angle-optimal τριγωνοποίηση είναι νόμιμη.

Με βάση αυτήν την τεχνική μπορούμε να οδηγηθούμε σε μία τριγωνοποίηση Delaunay.

Ορισμός **5.2.3:** Τριγωνοποίηση **Delaunay**

Για ένα σύνολο σημείων P στο επίπεδο, η τριγωνοποίηση Delaunay είναι η τριγωνοποίηση εκείνη που δεν περιέχει καμία παράνομη ακμή. Συμβολίζουμε την τριγωνοποίηση Delaunay των σημείων με $Del(P)$.

Μία εναλλακτική προσέγγιση των flips και των ελέγχων των διανυσμάτων γωνιών για την παραγωγή Delaunay τριγωνοποίησης είναι η προσέγγιση με τη βασικό εργαλείο τον κύκλο. Στο σημείο αυτό, επισημαίνουμε κάποιες βασικές ιδιότητες που αφορούν τον κύκλο και πηγάζουν από το θεώρημα του Θαλή.

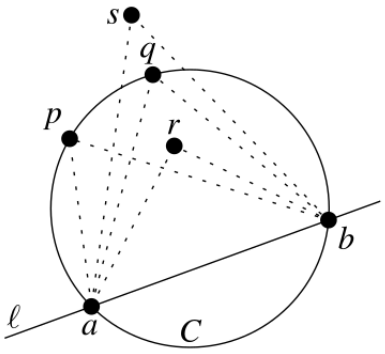

Figure 5.6: Κύκλος C με τόξο ab
πηγή: "Computational Geometry, Algorithms and Applications,Third Edition", σελ. 194

Θεώρημα 5.2.2: Γωνίες που βαίνουν σε χορδή

Έστω C να είναι ένας κύκλος και l μία ευθεία που τέμνει τον κύκλο στα σημεία a και b. Έστω p και q σημεία πάνω στην περίμετρο του κύκλου. Τότε η γωνία που βαίνει στο τόξο ab με κορυφή το σημείο p είναι ίση με την γωνία που βαίνει στο ίδιο τόξο και έχει κορυφή το σημείο q.

Θεώρημα 5.2.3: Γωνίες που βαίνουν σε χορδή

Έστω C να είναι ένας κύκλος και l μία ευθεία που τέμνει τον κύκλο στα σημεία a και b. Έστω r ένα σημείο εντός του κύκλου και s ένα σημείο εκτός. Τότε η γωνία που βαίνει στο τόξο ab με κορυφή το σημείο r είναι μεγαλύτερη από την γωνία που βαίνει στο ίδιο τόξο και έχει κορυφή το σημείο s.

Συνοψίζοντας

$$\angle arb > \angle apb = \angle aqb > \angle asb \tag{5.3}$$

Από όλα τα παραπάνω προκύπτει ότι

Λήμμα 5.2.1

Έστω C ένας κύκλος και p_i, p_j, p_k σημεία πάνω στον κύκλο και p_l σημείο εντός του κύκλου. Αν τα p_i, p_j, p_k, p_l σχηματίζουν ένα κυρτό πολύγωνο, τότε αυτό μπορεί να τριγωνοποιηθεί και να παραχθούν τα τρίγωνα $p_ip_jp_k$ και $p_ip_jp_l$ με κοινή ακμή την p_ip_j. Η ακμή p_ip_j είναι παράνομη.

Λήμμα 5.2.2

Έστω C ένας κύκλος και p_i, p_j, p_k σημεία πάνω στον κύκλο και p_l σημείο εκτός του κύκλου. Αν τα p_i, p_j, p_k, p_l σχηματίζουν ένα κυρτό πολύγωνο, τότε αυτό μπορεί να τριγωνοποιηθεί και να παραχθούν τα τρίγωνα $p_ip_jp_k$ και $p_ip_jp_l$ με κοινή ακμή την p_ip_j. Η ακμή p_ip_j είναι νόμιμη.

Φυσικά, να υπενθυμίσουμε ότι τα σημεία βρίσκονται σε γενική θέση, συνεπώς δεν παίρνουμε την περίπτωση τα p_i, p_j, p_k, p_l να είναι και τα 4 πάνω στον ίδιο κύκλο.

Στο παρακάτω θεώρημα συνοψίζεται η προσέγγιση της τριγωνοποίησης με εργαλείο τον κύκλο.

Θεώρημα 5.2.4: Η ιδιότητα των κενών κυκλων

Έστω P ένα σύνολο σημείων στο επίπεδο που βρίσκονται σε γενική θέση. Μία τριγωνοποίηση T είναι τριγωνοποίηση Delaunay αν και μόνο αν ο κυκλος που σχηματίζεται από κάθε τριάδα σημείων που αποτελούν τρίγωνο της τριγωνοποίησης δεν περιέχει κανένα άλλο σημείο του P στο εσωτερικό του.

Απόδειξη:

$\Rightarrow$

Αν κανένα σημείο του συνόλου σημείων P δεν βρίσκεται εσωτερικά του εκάστοτε κύκλου που σχηματίζεται από τις τρεις κορυφές ενός τριγώνου της τριγωνοποίησης, τότε όλες οι ακμές είναι νόμιμες. Συνεπώς, η τριγωνοποίηση είναι νόμιμη.

$\Leftarrow$ "Αν μια τριγωνοποίηση είναι Delaunay, τότε κανένα σημείο του P δεν βρίσκεται εντός του εκάστοτε κύκλου τριών κορυφών τριγώνου της τιγωνοποίησης"

Θα αποδείξουμε αυτήν την κατεύθυνση με απαγωγή σε άτοπο.

Έστω ότι η τριγωνοποίηση T είναι τριγωνοποίηση Delaunay και έστω ότι υπάρχει κύκλος που περιέχει σημεία του P εκτός των A, B, C που τον ορίζουν και είναι οι κορυφές ενός τριγώνου της τριγωνοποίησης.

Έστω ότι από όλα αυτά τα σημεία επιλέγεται εκείνο που απέχει την μικότερη απόσταση απο την ακμή του τριγώνου, έστω D.

Επειδή η τριγωνοποίηση είναι Delaunay, το τρίγωνο BCD δε μπορεί να ανήκει στην τριγωνοποίηση.

Έστω E ένα σημείο εκτός του κύκλου και BCE τρίγωνο. Το D βρίσκεται εντός του κύκλου που ορίζεται από τα σημεία B, C, E και εκτός του τριγώνου BCE. .....

Συνοψίζουμε τα κριτήρια για την Delaunay τριγωνοποίηση σε ένα θεώρημα.

Θεώρημα **5.2.5: Delaunay** Τριγωνοποίηση

Σε μία τριγωνοποίηση Delaunay T των σημείων P:

1. Τρία σημεία κατασκευάζουν τρίγωνο αν και μόνο αν ο περιγεγραμμένος ανοιχτός κύκλος του δεν περιέχει κανένα άλλο σημείο του P.

2. Δύο σημεία κατασκευάζουν ακμή αν και μόνο αν υπάρχει κύκλος με τα σημεία αυτά στην περιφέρειά του που δεν περιέχει κανένα άλλο σημείο του P

Για δεδομένη τριγωνοποίηση, θα αποφασίζουμε αν αυτή είναι Delaunay ανν ο περιγεγραμμένος κύκλος κάθε τριγώνου δεν περιέχει κανένα άλλο σημείο.

5.2.3 Κατασκευή **Delaunay** Τριγωνοποίησης

Αφού είδαμε τη θεωρητική βάση της τριγωνοποίησης Delaunay, μπορούμε πλέον να αναφερθούμε σε τεχνικές και αλγόριθμους που κατασκευάζουν μία τριγωνοποίηση Delaunay.

Αυξητκός αλγόριθμος

Ο αλγόριθμος αυτός δε γνωρίζει εκ των προτέρων όλα τα σημεία που πρόκειται να τριγωνοποιηθούν. Για τη ακρίβεια, λαμβάνει ένα σύνολο σημείων, σε κάθε βήμα του εξετάζει ενα απο τα σημεία αυτά και παράγει την τρέχουσα Delaunay τριγωνοποίηση σαν να μην υπάρχουν άλλα σημεία προς εξέταση, μέχρι που εξετάζει όλο το σύνολο σημείων και παράγει της τελική Delaunay τριγωνοποίηση. Η επιλογή σημείου σε κάθε βήμα γίνεται με τυχαίο τρόπο.

Ο αλγόριθμος έχει $O(nlogn)$ μέση πολυπλοκότητα, ενώ στη χείριστη περίπτωση η πολυπλοκότητά του είναι $O(n^2)$.

Ο αλγόριθμος επιλέγει 3 σημεία στο επίπεδο των οποίων το αντίστοιχο τρίγωνο περιλαμβάνει όλα τα σημεία προς τριγωνοποίηση. Η επιλογή των τριών αυτών σημείων δεν είναι τυχαία και πρέπει να πληρεί ορισμένα κριτήρια. Θα αναφερθούμε σε αυτά στη συνέχεια, αφού πρώτα παρουσιάσουμε τον πυρήνα του αλγορίθμου.

Ο αλγόριθμος αυξητικά παράξει σε κάθε του βήμα την τριγωνοποίηση. Για κάθε σημείο που ελέγχει εντόπίζει αν βρίσκεται εντός τριγώνου της ήδη κατασκευασμένης τριγωνοποίησης ή πάνω σε κάποια ακμής της τριγωνοποίησης. Στην περίπτωση που βρίσκεται μέσα σε τρίγωνο κατασκευάζονται τρείς νέες ακμές, οι οποίες εκτείνονται από το υπό εξέταση σημείο προς την κάθε κορυφή του τριγώνου που το περιέχει. Στην περίπτωση που το σημείο βρίσκεται πάνω σε ακμή της τριγωνοποίησης είναι εύκολο κανείς να συνειδητοποιήσει ότι η ακμή αυτή είναι κοινή ακμή για δύο τριγωνα. Συνεπώς, δημιουργούνται δύο νέες ακμές από το σημείο που εξετάζεται προς τη μία κορυφη του ενός και του άλλου τριγώνου που δεν ανήκουν στο ευθύγραμμο τμήμα. Για την καλύτερη κατανόηση των δύο περιπτώσεων παρέχεται η αντίστοιχη εικόνα. Αφού δημιουργηθούν οι νέες ακμές με κατάλληλο τρόπο μένει να ελέγξουμε ότι η τριγωνοποίηση που έχει παραχθεί είναι Delaunay. Η προσθήκη μιας νέας ακμής μπορεί να κάνει παράνομες τις ήδη υπάρχουσες ακμές της τριγωνοποίησης. Αυτο αντιμετωπίζεται με κατάλληλα flips κάθε φορά. Ο αλγόριθμος παρουσιάζεται συνοπτικά παρακάτω.

Ο αλγόριθμος σαφώς τερματίζει, καθώς το πλήθος των ακμών της τριγωνοποίησης είναι πεπερασμένο. Επίσης, ο αλγόριθμος είναι ορθός διότι εξετάζει κάθε ακμή ως προς την "νομιμότητά" της στο βήμα όπου καλεί τον αλγόριθμο "Νομιμοποίηση ακμής". Συνεπώς, ποτέ δε θα προκύψει παράνομη ακμή χωρίς να ελεγχθεί και να μετατραπεί σε νόμιμη. όπως ήδη έχουμε αναφέρει, μια τριγωνοποίηση είναι Delaunay αν είναι νόμιμη.

Algorithm 2: Αυξητικός αλγόριθμος τριγωνοποίησης Delaunay

Input: n σημεία του επιπέδου

Result: Τριγωνοποίηση Delaunay για n σημεία του επιπέδου

Δημιούργησε κατάλληλα σημεία p_{-1}, p_{-2}, p_{-3} στο επίπεδο έτσι ώστε το τρίγωνο που σχηματίζουν να περέχει τα n σημεία εισόδου. ;

for ένα σημείο του επιπέδου που δεν έχει ενταχθέι στην τριγωνοποίηση **do**

    Βρες σε ποιο τρίγωνο ή ακμή ανήκει ;

    if σημείο ανήκει σε τρίγωνο **then**

        Πρόσθεσε 3 νέες ακμές προς τις κορυφές του τριγώνου ;

        Κάλεσε τον αλγόριθμο 2 για τις 3 πλευρές του τριγώνου ;

    end

    else

        Πρόσθεσε 2 νέες ακμές προς τις κορυφές των τριγώνων που έχουν κοινή πλευρά την πλευρά που ανήκει το υπο εξέταση σημείο ;

        Κάλεσε τον αλγόριθμο 2 για τις 4 πλευρές των 2 τριγώνων ;

    end

end

Algorithm 3: Νομιμοποίηση ακμής

Input: Τριγωνοποίηση σημείων, ακμή της τριγωνοποίησης $p_i p_j$, σημείο p_r

Result: Τριγωνοποίηση που είναι νόμιμη

Βρές τρίγωνο (p_i, p_j, p_l) ;

if $p_i p_j$ παράνομη **then**

    Διαγραφή $p_i p_j$;

    Δημιουργία $p_r p_l$;

    $p_i p_l$ = νόμιμη ως προς την p_r ;

    $p_j p_l$ = νόμιμη ως προς την p_r ;

end

Στο σημείο αυτό μπορούμε να εξηγήσουμε τον τρόπο που επιλέγουμε τα τρία αρχικά σημεία που δημιουργούν τρίγωνο που περιβάλει όλα τα προς τριγωνοποίηση σημεία. Ουσιαστικά, τα

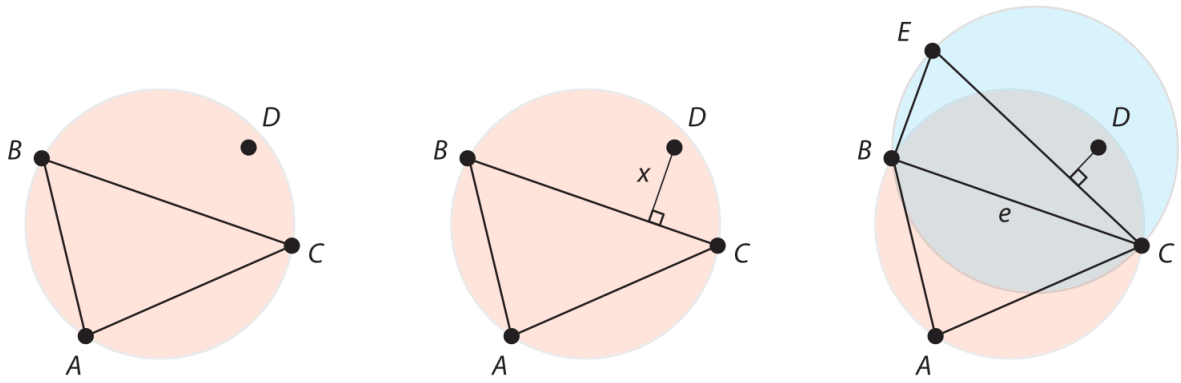

Figure 5.7: Απόδειξη του θεωρήματος 3.2.4
πηγή: "Discrete and Computational Geometry", σελ.: 86

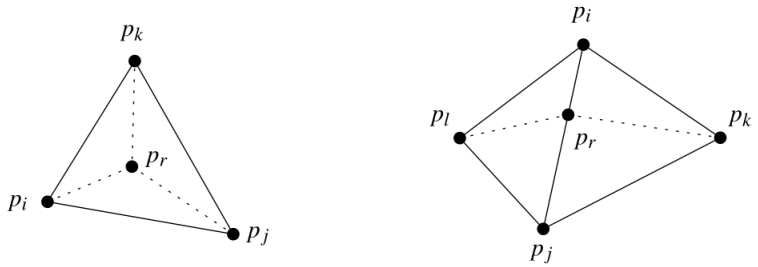

Figure 5.8: Οι δύο περιπτώσεις που εξετάζει ο αυξητικός αλγόριθμος κατά την εισαγωγή ενός νέου σημείου p_r. Αριστερά φαίνεται η πρώτη περίπτωση, όπου το νεό σημείο εντοπίζεται εντός τριγώνου της τριγωνοποίησης, ενώ δεξία φαίνεται η δεύτερη περίπτωση όπου το νέο σημείο βρίσκεται πάνω σε ακμή της τριγωνοποίησης
πηγή: "Computational Geometry, Algorithms and Applications, Third Edition", σελ.: 200

σημεία αυτά πρέπει να λάβουν θέσεις στο επίπεδο, που να μην επηρεάζουν την έκβαση της τριγωνοποίσης μετέπειτα. Αυτό επιτυγχάνεται μόνο αν τοποθετηθούν αρκετά μακρυά από τα σημεία που πρόκειται να τριγωνοποιηθούν. Για να συμβεί αυτό πρέπει καθένα από τα σημεία p_{-1}, p_{-2}, p_{-3} να μην βρίσκονται εντός των κύκλων που σχηματίζονται από όλες τις δυνατές τριάδες σημειων. Μόνο τότε μπορούμε να είμαστε βέβαιοι ότι δεν έχουν επηρεάσει την τριγωνοποίηση Delaunay.

Τέλος, πρέπει να αποσαφηνίσουμε τον τρόπο με τον οποίον ο αλγόριθμος αντιλαμβάνεται πότε ένα σημείο είναι εντός τριγώνου ή πάνω σε κάποια ακμή. Αυτό επιτυγχάνεται με τη δημιουργία ενός κατάλληλου κατευθυνόμενου ακυκλικού γράφου. Ο γράφος αυτός κωδικοποιεί όλα τα τρίγωνα που δημιουγήθηκαν κατά την εκτέλεση του αλγορίθμου και την χρονική τους ιεραρχία. Οι κόμβοι είναι τρίγωνα και οι ακμές δηλώνουν τις σχέσεις μεταξύ δύο τριγώνων. Αν δύο κόμβοι συνδέονται με ακμή, δηλαδη με σχέση "πατέρα - παιδιού", τότε τα αντίστοιχα τρίγωνα έχουν μη κενή τομή. Είναι προφανές πως η ρίζα του γράφου είναι το τρίγωνο $p_{-1}p_{-2}p_{-3}$. Έχουμε ήδη αναφέρει ότι η προσθήκη ενός νέου σημείου μπορεί να δημιουργήσει τρία ή τέσσερα τρίγωνα στην τριγωνοποίηση. Σε όρους γράφου, αυτό σημαίνει τρία ή τέσσερα νέα φύλλα. Το flip μιας ακμής δημιουργεί μόνο δύο φύλλα.

Για τον εντοπισμό ενός σημείου, διατρέχεται ο γράφος ξεκινόντας από τη ρίζα. Ελέγχονται τα τρία παιδία της ρίζας και εντοπίζουμε σε ποιο από αυτά τα τρία τρίγωνα (που περιέχονται στους κόμβους) ανήκει το προς εξέταση σημείο. Μεταβαίνουμε στο κατάλληλο παιδί και συνεχίζουμε την ίδια ακριβώς διαδικασία μέχρι να φτάσουμε σε φύλλο του γράφου στο οποίο και εντοπίζεται το σημείο που εξετάζουμε.

Chapter 6

Γραφοθεωρητική Προσέγγιση του Προβλήματος του πλανόδιου πωλητή

Η προσέγγιση του TSP με βάση τη θεωρία γραφημάτων συνδέεται στενά με τα τους Χαμιλτόνειους κύκλους. Επί της ουσίας το πρόβλημα του περιοδεύονοτς πωλητή είναι μία επέκταση του προβλήματος εύρεσης κυκλώματος Hamilton.

Όπως ήδη έχουμε αναφέρει στην εισαγωγή, στόχος του προβλήματος είναι ο πωλητής να επισκεφτεί ένα σύνολο πόλεων, ακριβώς μία φορά την κάθε μία, ελαχιστοποιόντας τις αποστάσεις που πρέπει να διανύσει. Αν αναπαραστήσουμε το σύνολο των πόλεων προς επίσκεψη με ένα σύνολο κορυφών V και το σύνολο όλων των πιθανών διαδρομών με ενα σύνολο E, τότε μπορούμε να μεταφέρουμε την εικόνα του χάρτη που μελετά ο πωλητής σε γραφική αναπαράσταση. Ο πωλητής εκκινεί και τερματίζει το ταξίδι του στην ίδια πόλη και επισκέπτεται όλες τις υπόλοιπες ακριβώς μία φορά, διανύοντας το έλαχιστον συνολικό μήκος.

Θεωρούμε το παραπάνω γράφημα του χαρτη των πόλεων και των αντίστοιχων διαδρομών να είναι το $G = (V, E, w(i, j))$, όπου το V περιλμβάνει τις n πόλεις (κορυφές), το E τις διαδρομές μεταξύ δύο πόλεων (ακμές) και το w να είναι μία συνάρτηση

$$w : E \rightarrow \mathbb{R}^{+} \tag{6.1}$$

τέτοια ώστε να ισχύει

$$w(i,k) \leq w(i,j) + w(j,k) \tag{6.2}$$

Ουσιαστικά το $w(i,j)$ είναι το μήκος της διαδρομής από την πόλη i στην πόλη j ή με όρους γραφημάτων το βάρος της ακμής (i,j). Στην περίπτωση κυκλώματος, το μήκος του ορίζεται να είναι το άθροισμα των μηκών των αντίστοιχν ακμών.

Το TSP αναζητά ένα κύκλωμα Hamilton με ελάχιστο μήκος. Δυστυχώς, μέχρι και σήμερα δε γνωρίζουμε κανέναν αλγόριθμο που να επιλύει το πρόβλημα του περιοδεύοντος πωλητή για μερικές εκατοντάδες πόλεων σε εύλογα χρονικά πλαίσια.

6.1 Η μέθοδος του πλησιέστερου γείτονα

Μία πολύ απλή μέθοδος για την εύρεση Χαμιλτονειανού κύκλου σε ένα γράφημα $G = (V, E, w(i, j))$ είναι η μέθοδος του πλησιέστερου γείτονα.

Algorithm 4: Μέθοδος πλησιέστερου γείτονα

Result: Κύκλωμα Hamilton για το πρόβλημα του περιοδεύοντος πωλητή

Επέλεξε αυθαίρετα μία κορυφή v ;

Ανέθεσε την v στην x ;

for *x* **do**

Επέλεξε μία κορυφή v που δεν βρίσκεται στο μονοπάτι και απέχει τη μικρότερη απόσταση από την τρέχουσα x ;

Πρόσθεσε την ακμή (x,v) στο μονοπάτι ;

Ανανέωσε την x με την v ;

end

Σχημάτισε κύκλωμα συνδέοντας την αρχική κορυφή με την τελική κορυφή του μονοπατιού ;

Chapter 7

Γεωμετρική Προσέγγιση του Προβλήματος του πλανόδιου πωλητή

7.1 Προσέγγιση με βάση την τριγωνοποίηση **Delaunay**

Ένας από τους πλέον διαδεδομένους γεωμετρικούς τρόπους προσέγγισης του προβλήματος του περιοδεύοντος πωλητή είναι η προσέγγιση με βάση την τριγωνοποίηση Delaunay. Έχουμε αναφερθεί εκτενώς στην τριγωνοποίηση Delaunay στο αντίστοιχο κεφάλαιο που αφορά το απαραίτητο μαθηματικό υπόβαθρο για την κατανόηση της παρούσας μελέτης.

Η τριγωνοποίηση Delaunay έχει μερικά χαρακτηριστικά τα οποία μπορούν να φανούν ιδιαιτέρως σημαντικά στο πρόβλημα του πλανόδιου πωλητή.

- Μεγιστοποιεί τη μικρότερη γωνία των τριγώνων της τριγωνοποίησης.
- Μπορεί να υπολογιστεί σε χρόνο $O(n \log(n))$.
- Για κάθε τρίγωνο της τριγωνοποίησης υπάρχει κύκλος που περιέχει τις τρεις κορυφές του τριγώνου και μόνο αυτές.
- Είναι μοναδική αν δεν υπάρχουν τέσσερα σημεία που βρίσκονται στον ίδιο κύκλο.

- Περιέχει proximity graphs όπως το MST, το nearest neighbor graph, relative neighbor graph, Gabriel graph.

Η Delaunay τριγωνοποίηση μπορεί να συμβάλλει στην εύρεση μιας κατάλληλης διαδρομής για τον πλανόδιο πωλητή. Ωστόσο, δεν είναι απαραίτητο ότι θα καταφέρει να εντοπίσει την βέλτιστη διαδρομή, δηλαδή τη διαδρομή εκείνη που ελαχιστοποιεί το συνολικό κόστος, με άλλα λόγια τη συνολική απόσταση που χρειάζεται να διανύσει ο πωλητής. Υπάρχουν περιπτώσεις, όπου οι αλγόριθμοι επίλυσης του TSP με τη βοήθεια της Delaunay τριγωνοποίησης δεν παράγουν το βέλτιστο μονοπάτι. Το βέλτιστο μονοπάτι δεν είναι πάντοτε ένα υποσύνολο του Delaunay γράφου. Ένα τέτοιο παράδειγμα φαίνεται στην εικόνα 5.1. Ωστόσο, η Delaunay τριγωνοποίηση μπορεί να δώσει μία αρκετά καλή προσέγγιση της λύσης του προβλήματος.

Στην υποενότητα αυτή παρουσιάζουμε έναν αλγόριθμο εύρεσης μονοπατιού για το πρόβλημα του πλανόδιου πωλητή με βάση την Delaunay τριγωνοποίηση όπως παρουσιάζεται στο [10].

Πρωτού ξεκινήσει η διαδικασία κατασκευής της διαδρομής, οι πόλεις, που πλέον λογίζονται ως σημεία στο επίπεδο, πρέπει να έχουν περάσει από τη διαδικασία της τριγωνοποίησης Delaunay. Έπειτα, από την τριγωνοποίηση των σημείων ο αλγόριθμος έχει να διαχειριστεί έναν γράφο, όπου οι πόλεις είναι οι κόμβοι του γράφου και οι διαδρομές (οι ακμές που έχουν παραχθεί από την τριγωνοποίηση) είναι οι ακμές του γράφου. Ο αλγόριθμος ξεκινά με μία διαδρομή T_i που περιέχει κάποιες από τις πόλεις σταθμούς και i το πλήθος ακμές που υποδεικνυουν την διαδρομή. Επιχειρεί με επαναλαμβανόμενα βήματα να εντάξει και τις υπόλοιπες.

Στο σημείο αυτό αναφέρουμε τον απαραίτητο συμβολισμό για την περαιτέρω ανάλυση του αλγορίθμου.

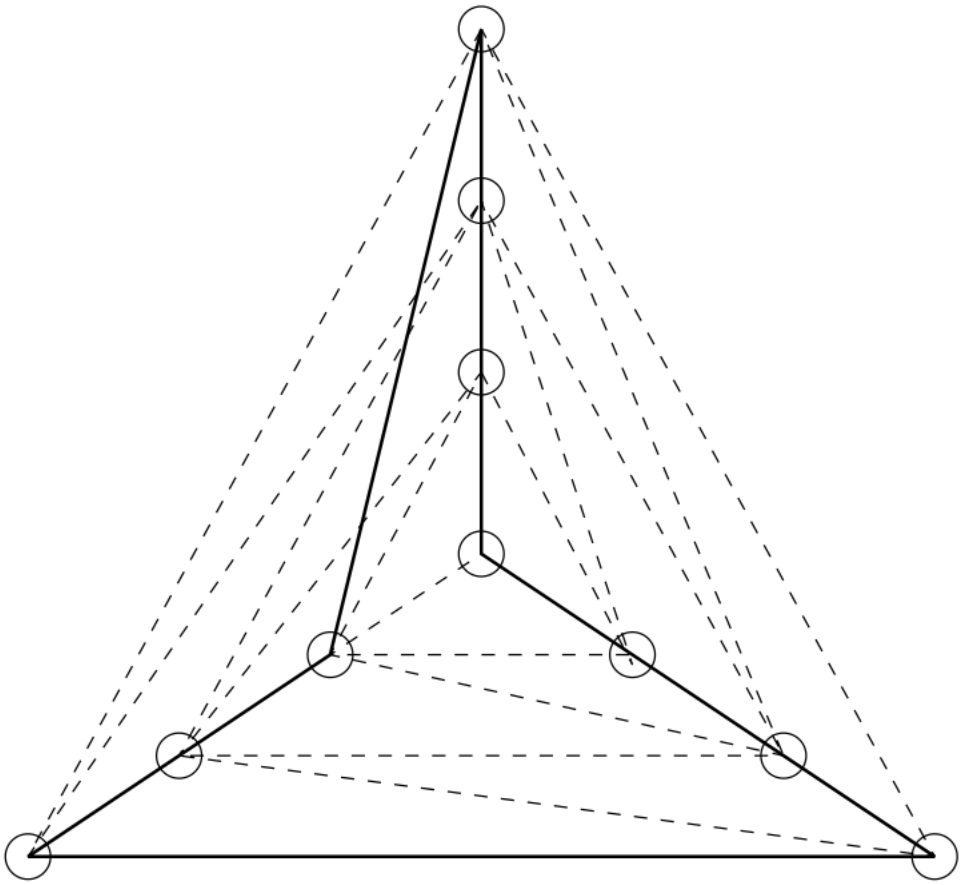

Figure 7.1: Η καλύτερη διαδρομή δεν είναι στο υποσύνολο του Delaunay γράφου
πηγή: ”An O(n log n) Heuristic for the Euclidean Traveling Salesman Problem” [10]

- E_{mn} είναι η ακμή που συνδέει τις πόλεις C_m, C_n
- $C_L(E_{mn})$ είναι η πόλη που βρίσκεται στο αριστερό τρίγωνο της E_{mn}
- $C_R(E_{mn})$ είναι η πόλη που βρίσκεται στο δεξί τρίγωνο της E_{mn}

Είναι προφανές πως κάθε ακμή του γράφου βρίσκεται ανάμεσα σε δύο τρίγωνα, εκτός αν πρόκειται για ακμή που βρίσκεται στο convex hull των σημείων.

Σε κάθε βήμα του αλγορίθμου μία από τις ακμές του T_i διαγράφεται και στη θέση της εισάγονται δυο κανούριες ακμές. Έστω πως η ακμή που διαγράφεται είναι η E_{mn}, που συνδέει τις πόλεις C_m και C_n. Οι δύο νέες ακμές που προσθέτονται στό T_{i+1} είναι οι E_{mL} και E_{Ln}, που συνδέουν την πόλη C_m με την πόλη $C_L(E_{mn})$ και την πόλη $C_L(E_{nm})$ με την πόλη C_n. Ως E_{nm} λογίζεται η ακμή E_{mn}, αλλά με αντίθετη φορά. Ευκολά μπορεί, λοιπόν κανείς να συμπεράνει ότι $C_L(E_{mn}) = C_L(E_{nm})$ Φυσικά, το βήμα αυτό λαμβάνει χώρα μόνο εαν η $C_L(E_{mn})$ δεν ανήκει στις πόλεις που ήδη περιέχονται στη διαδρομή του πλανόδου πωλητή στο βήμα T_i. Δίνεται ο αλγόριθμος Remove Edge, που υλοποιεί τη διαγραφή ακμής και την εισαγωγή δύο νέων ακμών παρακάτω. Θα μπορούσαμε αντίστοιχα να αναφερθούμε και στην πόλη $C_R(E_{mn})$, ωστόσο κατί τέτοιο είναι πλήρως αντίστοιχο με την αναφορά στην πόλη που βρίσκεται στα αριστερά, οπότε ο αλγόριθμος περιορίζεται μόνο στις πόλεις αυτές. Αφού, λοιπόν, ο αλγόριθμος επεκτείνεται προς τις πόλεις που βρίσκονται στα αριστερά, μία εύλογη επιλογή για αρχική διαδρομή θα ήταν το convex hull των σημείων με CCW σειρά.

Algorithm 5: Remove Edge

Result: Αντικατάσταση ακμής του μονοπατιού με 2 νέεο ακμές

Διέγραψε το top του heap E ;

e_1 = ακμή αριστερού τριγώνου ;

e_2 = ακμή αριστερού τριγώνου ;

v = κορυφή μεταξύ των e_1, e_2 ;

Πρόσθεσε το v στο tour ;

Πρόσθεσε τα e_1, e_2 στο E ;

Το ερώτημα τώρα είναι πως καθορίζεται η επιλογή της E_{mn}. Μία ακμή επιλέγεται να διαγραφεί, έναντι των άλλων, σύμφωνα με την ποιότητά της. Η ποιότητα της ακμής E_{mn} συμβολίζεται με Q_{mn} (από το quality). Ο υπολογισμός του μεγέθους Q_{mn} δεν έγκειται σε μία καθορισμένη διαδικασία. Μερικές προτεινόμενες προσεγγίσεις δίνονται παρακάτω. Θεωρούμε $C_l = C_L(E_{nm})$ και L_i το μήκος της ακμής i.

- Farthest insertion : $Q_{mn} = -min(L_{ml}, L_{ln})$
- Nearest insertion : $Q_{mn} = min(L_{ml}, L_{ln})$
- Cheapest insertion : $Q_{mn} = -L_{ml} + L_{ln} - L_{mn}$

Σε μία κατάλληλη δομή δεδομένων φυλάσσονται τα qualities των ακμών, ώστε να μπορεί να γίνει η επιλογή της κατάλληλης ακμής προς διαγραφή. Μία τέτοια δομή είναι το heap.

Μετά την εκτέλεση των παραπάνω βημάτων είναι πολύ πιθανό να υπάρχουν πόλεις οι οποίες να μην περιλαμβάνονται ακόμα στο μονοπάτι του πλανόδιου πωλητή. Σε αυτήν την περίπτωση πρέπει να υπάρξει ειδική μέριμνα. Έστω ότι η πόλη που δεν ανήκει στο μονοπάτι μετα την εκτέλεση του αλγορίθμου είναι η C_x. Τότε σίγουρα η C_x θα είναι γειτονική με κάποια άλλη πόλη, έστω τη C_m. Επιλέγεται για διαγραφή η ακμή E_{mn} και προστίθονται οι ακμές E_{mx} και E_{xn}. Αν υπάρχουν

πολλές επιλογές για την επιλογή ακμής προς διαγραφή, τότε επιλέγεται εκείνη που ελαχιστοποιεί την αύξηση του τελικού μονοπατιού ($L_{mx} + L_{xn} - L_{mn}$). Να σημειωθεί εδω ότι η ακμή E_{xn} δεν είναι απαραίτητο να είναι ακμή της τριγωνοποίησης.

Ο υπορουτίνα που υλοποιεί την παραπάνω διαδικασία ονομάζεται Tour Finish.

Algorithm 6: Tour Finish

Result: Ενσωμάτωση των πόλεων που δεν περιλαμβάνονται ακόμα στο μονοπάτι

Τοποθέτησε όλες τις πόλεις που δεν έχουν ακόμα ενσωματωθεί στο μονοπάτι σε ένα heap ;

for πόλη *c* **do**

    Αναζήτησε τις ακμές που μπορούν να χρησιμοποιηθούν για την εισαγωγή της πόλης ;

    if δεν υπάρχει κατάλληλη ακμή **then**

        Τοποθέτησε την πόλη c στο heap ;

        continue ;

    end

    Διαγραφή ακμής ;

    Πρόσθεση 2 ακμών για την εισαγωγή της c ;

end

Δύο βασικές βελτιώσεις στον αλγόριθμο που αναφέραμε είναι το clustering και το linear clustering.

Το cluestring αναφέρεται στα υπομονοπάτια που δημιουργούνται ξεκινόντας από το βασικό μονοπάτι (convex hull). Αν κατά τη διαδικασία αφαίρεση ακμής και εισαγωγής δύο νέων ακμών ο αλγόριθμος αποφασίσει πως αυτή του η ενέργεια είναι πιο δαπανηρή σε σχέση με τη δημιουργία μίας νέας υποδιαδρομής, τότε αλλάζει απόφαση και αντί να εκτελέσει το βασικό βήμα που περιγράψαμε παραπάνω δημιουργεί ένα ένα υπομονοπάτι, το οποίο καλείται cluster. Συνεπώς, ένα cluster είναι μία υποδιαδρομή, δηλαδή ένα σύνολο πόλεων που ενώνονται μεταξύ τους με ακμές. Το μικρότερο cluster που μπορεί να προκύψει αποτελείται από τρείς πόλεις, οι οποίες εννώνονται

και δημιουργούν τρίγωνο.

Ο αλγόριθμος από την πρώτη στιγμή έχει γνώση των τριγώνων που έχουν δημιουργηθεί από την Delaunay τριγωνοποίηση και τα αποθηκέυει σε κατάλληλη δομή δεδομένων όπως θα δούμε παρακάτω. Κάθε τρίγωνο αξιολογείται για την ποιότητά του, όπως ακριβώς και οι ακμές από τον παρακάτω τύπο:

$$Q_t = \frac{min(L_a, L_b, L_c)}{mid(L_a, L_b, L_c)}$$

Μικρό Q_t σημαίνει ότι το τρίγωνο έχει μία πάρα πολύ μικρή γωνία και δύο πολύ μεγάλες σε μήκος ακμές.

Τα linear clusters είναι και αυτά clusters με τη διαφορά ότι δε δημιουργούν κύκλο, είναι απλά ένα μονοπάτι πάνω στον γράφο και η αρχική πόλη του μονοπατιού με την τελική δεν συμπίπτουν.

Παρακάτω παραθέτουμε τους αλγόριθμους που υλοποιούν τις διαδικασίες που σχετίζονται με τα clusters και τα linear clusters.

Algorithm 7: Remove Cluster Edge

Result: Διαγραφή ακμής που ανήκει σε cluster και προσθήκη δύο νέων ακμών

Διέγραψε το top του heap C ;

e_1 = ακμή παρακείμενου τριγώνου ;

e_2 = ακμή παρακείμενου τριγώνου ;

v = κορυφή μεταξύ των e_1, e_2 ;

Πρόσθεσε το v στο cluster ;

Πρόσθεσε τα e_1, e_2 στο C ;

Algorithm 8: Generate Cluster

Result: Δημιουργία τριγώνου - cluster

Διέγραψε το top του heap T ;

Δημιούργησε τρίγωνο ;

Πρόσθεσε τις ακμές στο C ;

Algorithm 9: Generate Linear Cluster

Result: Δημιουργία linear cluster

for πόλη που δεν ανήκει στο *tour* **do**

    Άνοιξε clusters με Delaunay ακμές ;

    Ενοποίησε την πόλη στο tour ;

end

Algorithm 10: Merge

Result: Ενοποίηση Cluster με Cluster ή CLuster με Tour

Διέγραψε το top του heap M ;

if *Cluster-Cluster* **then**

    Δίεγραψε 2 ακμές ;

    Προσθεσε 2 ακμές ;

    Προσθεσε τις ακμές στο C ;

end

else

    Δίεγραψε 2 ακμές ;

    Προσθεσε 2 ακμές ;

    Προσθεσε τις ακμές στο E ;

end

Όπως είναι φανερό γίνεται εκτεταμένη χρήση της δομής heap. Πιο συγκεκριμένα χρησιμοποιείται ένα heap για τις ακμές του μονοπατιού και ένα heap E που θα φιλοξενήσει ακμές ταξινομημένες σύμφωνα με την ποιότητά τους (Q). Το E αρχικοποιείται με τις ακμές που convex hull των σημείων. Επίσης, χρησιμοποιείται ένα heap C, που περιέχει όλα τα clusters, ένα heap T, που περιέχει όλα τα τρίγωνα, ταξινομημένα σύμφωνα με την ποιότητα τους και ένα heap M, που περιέχει όλες τις πόλεις που προορίζονται για cluster merging. Παραθέτουμε τον βασικό αλγόριθμο που καλεί τις παραπάνω υπορουτίνες και υλοποιεί την ιδέα που αναπτύχθηκε στην αρχή της παρούσας ενότητας για την εύρεση μιάς καλής προσέγγισης του βέλτιστου μονοπατιού για τον πλανόδο πωλητή.

Η επιλογή της ενέργειας που θα εφαρμόσει σε κάθε βήμα ο αλγόριθμος γίνεται σύμφωνα με το κόστος που αυτή κατέχει (όπως ήδη έχουμε αναφέρει ενδεικτικά στην παράγραφο που αφορά τα clusters). Το κόστος αυτό υπολογίζεται από αντίστοιχες υπορουτίνες.

Algorithm 11: TSP with Dealunay Triangulation

Result: Δημιουργία μονοπατιού

Υπολογισμός convex hull ; Υπολογισμός Delaunay τριγωνοποίησης ;

Υπολογισμός quality ακμών ; Υπολογισμός quality τριγώνων ;

initial tour = convex hull ;

E = ακμές του convex hull ; C = NULL ; T = τρίγωνα ; M = NULL ;

while *True* **do**

    edge cost = Edge Cost(E) ; cluster cost = Cluster Cost(C) ;

    triangle cost = Triangle Cost(T) ; merge cost = Merge(M) ;

    if *egde cost > threshold and cluster cost > threshold* **then**

        break ;

    end

    if *edge cost is best* **then**

        Remove Edge(E) ;

        continue ;

    end

    if *cluster cost is best* **then**

        Remove Cluster Edge(C) ;

        continue ;

    end

    if *triangle cost is best* **then**

        Generate Cluster(T) ;

        continue ;

    end

    if *merge cost is best* **then**

        Merge(M) ;

        continue ;

    end

end

Generate Linear Cluster() ;

Tour Finish() ;

Algorithm 12: Edge Cost

Result: Υπολογισμός κόστους διαγραφής ακμής μονοπατιού

e = top του heap E ;

e_1 = ακμή παρακείμενου τριγώνου ;

e_2 = ακμή παρακείμενου τριγώνου ;

v = κορυφή μεταξύ των e_1, e_2 ;

if v ανήκει στο μονοπάτι **then**

    Κάλεσε ξανά τη διαδικασία ;

end

if v ανήκει σε *cluster* **then**

    Υπολόγισε το κόστος για merge με το cluster ;

    Πρόσθεσε την e στο M ;

    Κάλεσε ξανά τη διαδικασία ;

end

Επέστρεψε το κόστος της e ;

Algorithm 13: Cluster Cost

Result: Υπολογισμός κόστους διαγραφής ακμής cluster

e = top του heap C ;

e_1 = ακμή παρακείμενου τριγώνου ;

e_2 = ακμή παρακείμενου τριγώνου ;

v = κορυφή μεταξύ των e_1, e_2 ;

if v ανήκει στο *cluster* **then**

    if v ανήκει στο ίδιο *cluster* **then**

        Κάλεσε ξανά τη διαδικασία ;

    end

end

Υπολόγισε κόστος **merge** με άλλο cluster ;

Πρόσθεσε το e στο M ;

Κάλεσε ξανά τη διαδικασία ;

if v ανήκει στο μονοπάτι **then**

    Υπολόγισε το κόστος για **merge** με το μονοπάτι ;

    Πρόσθεσε την e στο M ;

    Κάλεσε ξανά τη διαδικασία ;

end

Επέστρεψε το κόστος της e ;

Algorithm 14: Triangle Cost

Result: Υπολογισμός κόστους δημιουργίας τριγώνου

t = top του heap T ;

if υπάρχει πόλη στο τρίγωνο που ανήκει σε *cluster* ή στο μονοπάτι **then**

    Κάλεσε ξανά τη διαδικασία ;

end

Επέστρεψε το κόστος της e ;

7.2 Time Window TSP

7.2.1 Deadline TSP

Το Deadline TSP είναι μία παραλλαγή του κλασσικού TSP. Στην περίπτωση αυτή έχουμε n σημεία που πρέπει να επισκεφτεί ο πωλητής, έχοντας ως αφετηρία το σημείο r. Κάθε σημείο v έχει ένα χρονικό deadline $D(v)$. Στόχος του προβλήματος είναι η εύρεση ενός μονοπατίου που ξεκινά από το σημείο r και διέρχεται από όσα το δυνατόν περισσότερα σημεία προτού λήξει το χρονικό deadline τους. Συνεπώς, η έλευση του πωλητή από ένα σημείο είναι έγκυρη αν γίνει πριν το εκάστοτε deadline. Το μονοπάτι πρέπει να περιέχει όσο γίνεται περισσότερα σημεία. Αν υποθέσουμε ότι κάθε σημείο επίσκεψης έχει μία τιμή η οποία προστίθεται στην αξία του μονοπατιού, τότε στόχος του Deadline TSP προβλήματος είναι να βρεί ένα μονοπάτι που μεγιστοποιεί το άθροισμα των τιμών αυτών.

7.2.2 Time Window TSP

Το Time Window TSP ή αλλιώς Vehicle Routing Problem with Time Windows είναι μία περίπτωση του Deadline TSP που ήδη αναφέραμε. Στην περίπτωση του Time Window TSP κάθε σημείο επίσκεψης v, εκτός από το deadline $D(v)$, διαθέτει καί έναν χρόνο ελευθέρωσης (release time) $R(v)$. Συνεπώς, κάθε σημείο επίσκεψης χαρακτηρίζεται από ένα time window $[R(v), D(v)]$. Ο πωλητής μπορεί να επισκεφτεί ένα σήμειο μόνο εντός του χρονικού πλαισίου $[R(v), D(v)]$.

Θεωρούμε το Deadline TSP ως μία γενίκευση του Time Window TSP, όπου το $R(v) = 0$ για όλους τα σημεία επίσκεψης v.

Θα μελετήσουμε το Time Window TSP και τη γενίκευση του ως προς την αποτελεσ-

ματικότητά τους. Για να συμβεί αυτό είναι απαραίτητο να εισάγουμε μία βασική ορολογία όπως προκύπτει από το [12].

Θεωρούμε γράφο $G = (V, E)$ με βάρη στις ακμές του. V είναι το σύνολο των σημείων επίσκεψης και E το σύνολο των διαδρομών ανάμεσα στα σημεία επίσκεψης. Οι ακμές διαθέτουν βάρη που αντιστοιχούν στο κόστος της διαδρομής από τον έναν κόμβο στον άλλον. Θεωρούμε επίσης r να είναι ο κόμβος εκκίνησης του πλανόδιου πωλητή. Έστω συνάρτηση $\Pi : V \to \mathbb{Z}^+$ να είναι μία συνάρτηση επιβράβευσης. Ο πωλητής λαμβάνει επιβράβευση κάθε φορά που επισκέπτεται ένα σημείο εντός σωστού χρονικού πλαισίου. Έστω $D : V \to \mathbb{Z}^+$ συνάρτηση που δίνει τα **deadlines** των κόμβων και $R : V \to \mathbb{Z}^+$ σύνάρτηση που δίνει το **release time**. Χωρίς βλάβη της γενικότητας θεωρούμε για κάθε κόμβο ότι $D(v) > R(v)$. Τέλος, θεωρούμε συνάρτηση $l : E \to \mathbb{Z}^+$ που δίνει το μήκος μίας ακμής. Στο εξής, θα γράφουμε $l(u, v)$ και θα εννοούμε τη μικρότερη διαδρομή ανάμεσα στους κόμβους u και v στον γράφο G. Ο χρόνος της διαδρομής από το u στο v μέσα σε ένα μονοπάτι P θα συμβολίζεται με $t_P(u, v)$. Αν $u = r$, τότε $t_P(r, v) = t_P(v)$. Το t εκφράζει τον χρόνο ή εναλλακτικά την απόσταση ενός μονοπατιου. Ακόμα, συμβολίζουμε με **O** το βέλτιστο μονοπάτι.

Στόχος εδώ είναι να βρεθεί μονοπάτι που ξεκινά από τον κόμβο r και μεγιστοποιεί τη συνολική επιβράβευση. Να σημειωθεί πως ο πωλητής μπορεί να επισκεφτεί έναν κόμβο περισσότερες από μία φορές, ωστόσο μόνο μία φορά μπορεί να αποκομίσει από αυτόν επιβράβευση.

Το σκεπτικό που θα ακολουθήσουμε πηγάζει από το [12], εισάγει αρχικά μία $O(\log n)$ προσέγγιση για το Deadline TSP και έπειτα πάνω σε αυτήν βασίζει την $O(\log^2 n)$ προσέγγιση για το Time Window TSP.

Σε ό,τι αφορά το Deadline TSP η βασική ιδέα είναι να διαχωριστούν τα σημεία σε υποσύνολα V_i τέτοια ώστε

$$V_i = \{v \in V : D(v) \in (d_i, d_{i+1}]\}$$

όπου d_i συμβολίζει το i-οστό deadline. Στόχος εδώ είναι να βρεθεί μονοπάτι το οποίο επισκέπτεται πρώτα όλους τους κόμβους στο V_i σύνολο και έπειτα στο V_j, όπου $j > i$ και αποκομίζει $\Omega(\frac{1}{\log n})$ από την βέλτιστη επιβράβευση.

Για την κωδικοποίηση του προβλήματος κρίνεται απαραίτητο να τοποθετήσουμε τα σημεία επίσκεψης σε ένα σύστημα δύο αξόνων, όπου ο οριζόντιος άξονας αναφέρεται στα deadlines και ο κάθετος στόν χρόνο. Συνεπώς, κάθε σήμειο έχει τη μορφη

$$p_i = (D(v_i), t_O(v_i))$$

Εισάγουμε τώρα έναν σημαντικό ορισμό.

Ορισμός 7.2.1: Minimal κόμβος

Minimal κόμβος καλείται κάθε κόμβος v_i για τον οποίον είτε ισχύει ότι $t_O(v_i) \geq t_O(v_j)$, είτε $D(v_i) \leq D(v_j)$ για κάθε $v_j \neq v_i$.

Το σύνολο των minimal κόμβων οργανώνεται σε ένα σύνολο $M = \{v_0, ..., v_m\}$ σε αύξουσα σειρά σύμφωνα με το $D(v)$.

Κάθε τριάδα minimal κόμβων δημιουργούν ένα τετράγωνο στο σύστημα αξόνων που προαναφέραμε. Πιο συγκεκριμένα, έστω τα minimal σημεία $v_h, v_j, v_k \in M$ όπου $h \leq j \leq k$. Τα τρία αυτά σημεία δημιουργούν ένα τετράγωνο $R(h, j, k)$ που ικανοποιεί για v κόμβο τις εξής συνθήκες:

$$D(v_j) \leq D(v) \leq D(v_k)$$

$$t_O(v_h) \leq t_O(v) \leq t_O(v_j)$$

Από όλα τα σημεία στο $R(h, j, k)$ ο πλανώδιος πωλητής τελευταίο επισκέπτεται το v_j, συνεπώς όλα τα υπόλοιπα σημεία του R πρέπει να τα έχει επισκεφτεί μέχρι τη χρονική στιγμή $D(v_j)$. Αυτό μας επιτρέπει να χρησιμοποιήσουμε point-to-point orienteering. Στο point-to-point orienteering είναι καθορισμένο το σημείο εκκίνησης και το σημείο τερματισμού και ο πωλητής πρέπει να διανύσει μία διαδρομή ξεκινόντας από την εκκίνηση και καταλήγοντας στον τερματισμό χωρίς να διανύσει απόσταση μεγαλύτερη από μία προκαθορισμένη τιμή D. Όλο αυτό πρέπει να το επιτύχει συλλέγοντας τη μεγαλύτερη δυνατή επιβράβευση.

Εισάγουμε τώρα έναν ακόμη ορισμό.

Ορισμός 7.2.2: Ξένα τετράγωνα

Δύο τετράγωνα $R_1 = R(h_1, j_1, k_1)$ και $R_2 = R(h_2, j_2, k_2)$ είναι ξένα αν $h_2 \geq j_1$ και $j_2 \geq k_1$. Εναλλακτικά, δύο τετράγωνα είναι ξένα αν στην γραφική τους απεικόνιση οι άξονες του χρόνου και του deadline είναι ξένοι.

Συμβολίζουμε με C μία συλλογή ξένων τετραγώνων, που είναι ανά δύο ξένα. Η επιλογή μελέτης ξένων τετραγώνων δεν είναι τυχαία. Με αυτόν τον τρόπο δεν υπερβαίνουμε το χρονικό όριο εκτέλεσης της διαδρομής, ούτε υπολογίζουμε τους κόμβους παραπάνω από μία φορά.

Σύμφωνα με το [12] μία συλλογή C_i δίνεται από τον τύπο

$$C_i = R(n_{i,j-1}, n_{i,j}, n_{i,j+1})_{-1 \leq j \leq 2^{\log m - i}} \tag{7.1}$$

όπου

$$n_{i,j} = j2^i + 2^{i-1} : 0 \leq j \leq 2^{\log m - i} - 1$$

$$n_{i,-1} = 0$$

$$n_{i,2^{\log m - i}} = m$$

και θέτουμε σύνολο S τέτοιο ώστε

$$S_i = \left\{ n_{i,j} : 0 \leq j \leq 2^{\log m - i} - 1 \right\} \bigcup \{0, m\} \tag{7.2}$$

Μερικές συλλογές ενδεικτικά για $i = 1, 2, 3$ είναι οι εξής

$$C_1 = \left\{ R((j-1)2 + 1, j2 + 1, (j+1)2 + 1) : 0 \leq j \leq \frac{m}{2} - 1 \right\}$$
$$C_1 = \left\{ R(2j - 1, 2j + 1, 2j + 3) : 0 \leq j \leq \frac{m}{2} - 1 \right\}$$

$$C_2 = \left\{ R((j-1)4 + 2, j4 + 2, (j+1)4 + 2) : 0 \leq j \leq \frac{m}{4} - 1 \right\}$$
$$C_2 = \left\{ R(4j - 2, 4j + 2, 4j + 6) : 0 \leq j \leq \frac{m}{4} - 1 \right\}$$

$$C_3 = \left\{ R((j-1)8 + 4, j8 + 4, (j+1)8 + 4) : 0 \leq j \leq \frac{m}{8} - 1 \right\}$$
$$C_3 = \left\{ R(8j - 4, 8j + 4, 8j + 12) : 0 \leq j \leq \frac{m}{8} - 1 \right\}$$

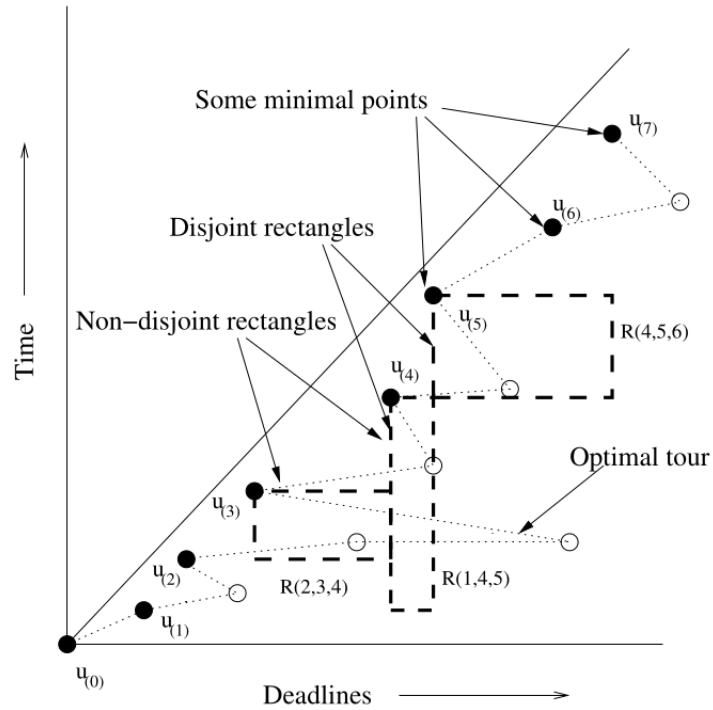

Figure 7.2: Γραφική αναπαράσταση minimal κόμβων σε σύστημα συντεταγμένων Deadline-Time
πηγή: Approximation Algorithms for Deadline-TSP and Vehicle Routing with Time-Windows [12]

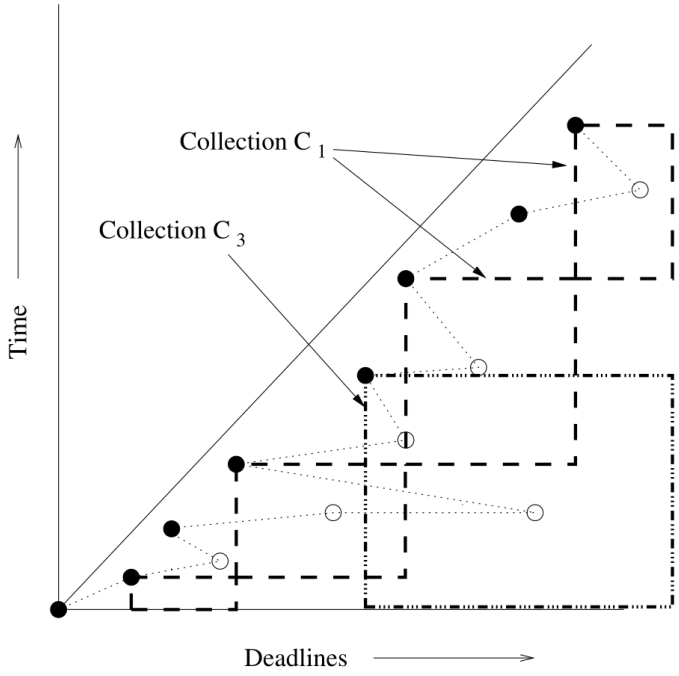

Figure 7.3: Γραφική αναπαράσταση minimal κόμβων σε σύστημα συντεταγμένων Deadline-Time και ξένα τετράγωνα
πηγή: Approximation Algorithms for Deadline-TSP and Vehicle Routing with Time-Windows [12]

Το ερώτημα τώρα είναι πόσο μεγάλο μπορεί να γίνει το i. Για να βρούμε το επιτρεπτό εύρος τιμών για το i πρέπει να ανατρέξουμε στο επιτρεπτό εύρος τιμών για το j. όπως υποδεικνύει η σχέση (7.1)

$$-1 \leq j \leq 2^{\log m - i}$$

Κοιτάζουμε το άνω φράγμα για το j

$$2^{\log m - i} = 2^{\log m} \cdot 2^{-i} = \frac{m}{2^i}$$

αν υποθέσουμε ότι το $\log$ έχει βάση το 2. Για προφανείς λόγους το άνω φράγμα του j πρέπει να είναι μεγαλύτερο της μονάδας, οποτε

$$\frac{m}{2^i} \geq 1 \Leftrightarrow$$
$$m \geq 2^i \Leftrightarrow$$
$$i \leq \log m$$

Υποθέτουμε, λοιπόν, ότι έχουμε μια οικογένεια συλλογών από ξένα τετράγωνα $F = C_1, \ldots, C_{\log m}$.

Chapter 8

Results

Chapter 9

Discussion and Future work

Chapter 10

Acknowledgement

Chapter 11

References

[1] Exact and Approximation Algorithms for Time-Window TSP, Jie Gao, Su Jia, Joseph S. B. Mitchell, CG:YRF, Boston, MA, USA, June 14-18, 2016

[2] An Optimal Lower Bound for the Hilbert-type, Planar Universal Traveling Salesman Problem, Patrick Eades, Julián Mestre, CG:YRF, Brisbane, Australia, July 4-7, 2017

[3] The Geometric Maximum Traveling Salesman Problem, David S. Johnson, Arie Tamir, Article in Journal of the ACM · May 2002

[4] Εισαγωγή στους αλγορίθμους, Δεύτερη έκδοση, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Πανεπιστημιακές εκδόσεις Κρήτης, 2011, ISBN: 978-960-524-473-6

[5] Τεχνητή Νοημοσύνη, Μία σύγχρονη προσέγγιση, Δεύτερη Αμερικανική έκδοση, Stuart Russel, Peter Norvig, σελ.: 101, Κλειδάριθμος 2005, ISBN: 960-209-873-2

[6] Στοιχεία διακριτών μαθηματικών, C. L. Liu, σελ.: 171-172, 178-179, 190-201, Πανεπιστημιακές εκδόσεις Κρήτης 2014, ISBN: 978-960-524-072-1

[7] Discrete and Computational Geometry, Satyan L. Devadoss, Joseph O'Rourke, σελ.: 81-86, Princeton University Press, 2011, ISBN: 978-0-691-14553-2

[8] Computational Geometry, Algorithms and Applications, Third Edition, Mark de Berg, Otfried Cheong, Marc van Kreveld, Mark Overmars, σελ.: 193-204, Springer, 2008, ISBN: 978-3-540-77973-5

[9] Υπολογιστική Γεωμετρία: Μια σύγχρονη αλγοριθμική προσέγγιση, Γιάννης Ζ. Εμίρης, σελ.: 199-208, Κλειδάριθμος, 2008, ISBN: 978-960-461-141-6

[10] An O (n log n) Heuristic for the Euclidean Traveling Salesman Problem, Evgeny Yanenko, Eckart Schuhmacher, Ulrich Spörlein, Kurt Tutschku, April 25, 2005

[11] Applications of the TSP, http://www.math.uwaterloo.ca/tsp/apps/index.html

[12] Approximation Algorithms for Deadline-TSP and Vehicle Routing with Time-Windows, N. Bansal, A. Blum, S. Chawla, and A. Meyerson, In Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing, STOC '04, pages 166–174, 2004

[13] Good triangulations yield good tours, Adam N. Letchford, Nicholas A. Pearson, Department of Management Science, Lancaster University, Lancaster LA1 4YW, UK, 4 May 2006

[14] Σχεδίαση και Ανάλυση Αλγορίθμων, Κωνσταντίνος Τσίχλας, Ιωάννης Μανωλόπουλος, Αναστάσιος Γούναρης, σελ.: 317-320, http://repfiles.kallipos.gr/html_books/4410/contents.html ,2015 ISBN: 978-960-603-465-7