ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ

ΑΛΓΟΡΙΘΜΟΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ

02/06/2023

ΟΜΑΔΑ 22

Νικος Κρασσας 5449

Οδυσσεας Καραγιαννης 5420

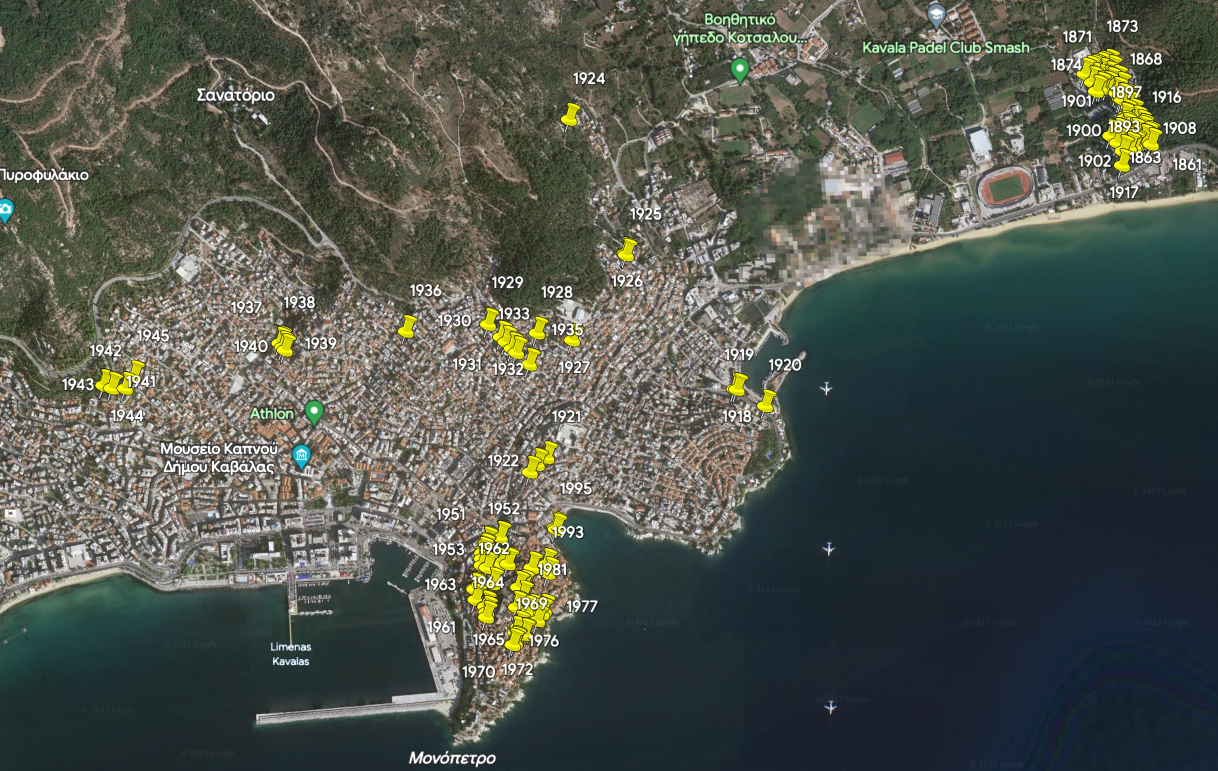
**ΕΙΣΑΓΩΓΗ**

Στα πλαισια της αξιολόγησης μας για το μαθημα ΑΛΓΟΡΗΘΜΟΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ μας ανατέθηκε ένα προβλημα , με σκοπο η επιλυση του να λειτουργήσει ως δείκτης της κατανόησης μας για την διδαχθείσα υλη του εξαμήνου

**ΤΟ ΠΡΟΒΛΗΜΑ**

Εχοντας ενα αρχειο kml το οποίο περιγράφει 136 ζευγαρια

συντεταγμένων στον χάρτη (κάθε σημειο εντιπροσωπευει και έναν καδο), καλουμαστε να αξιοποιησουμε τον καταλλλο αλγορυθμο , ο οποιος θα βρησκει την βελτιστη διαδρομη ετσι ώστε το επορριματοφορο να επισκεφθει όλα τα σημεια πανω στον χαρτη , ακριβως μια φορα , καλυπτοντας ταυτοχρονα την μικροτερη δυνατη αποσταση. Για λογους ευκολιας θα υπολογισουμε την κάθε αποσταση ως τιμη στην μοναδα των χιλιομετρων

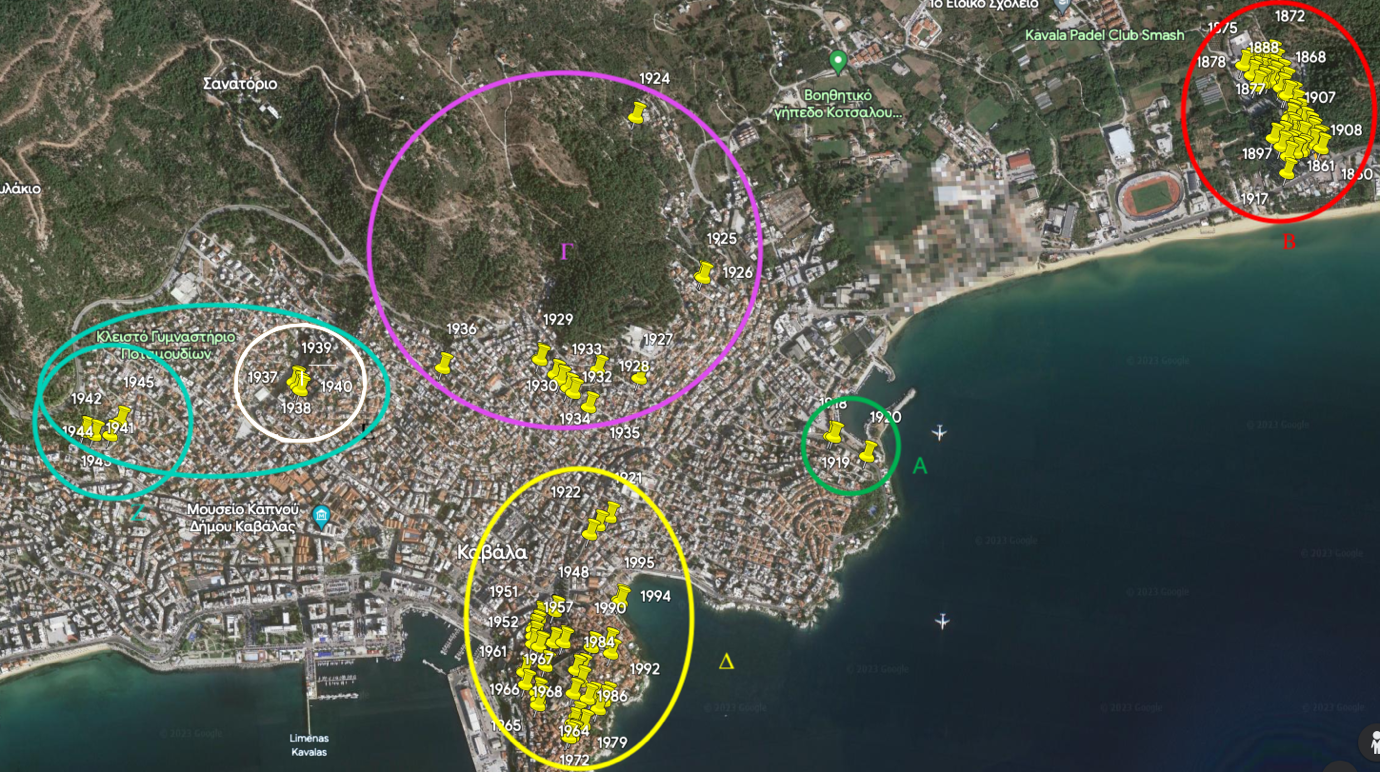


Εικόνα 1: σημεια στον χαρτη

**ΕΠΙΛΥΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ**

1.ΟΜΑΔΟΠΟΙΗΣΗ ΓΕΙΤΟΝΙΚΩΝ ΣΗΜΕΙΩΝ

Προκειμενου να προσεγγυσουμε την βελτιστη διαδρομη οσο καλυτερα μπορουμε αλλα ταυτοχρονα να μειωσουμε τον χρονο ευρεσης αυτης , θα χωρισουμε τα σημεια που φαινονται να είναι πολύ κοντα μεταξυ τους σε ομαδες (clusters) και θα δωσουμε ένα γραμμα στην κάθε μια και ένα ξεχωριστό χρωμα για την ευκολη αναγνωρηση τους αργοτερα



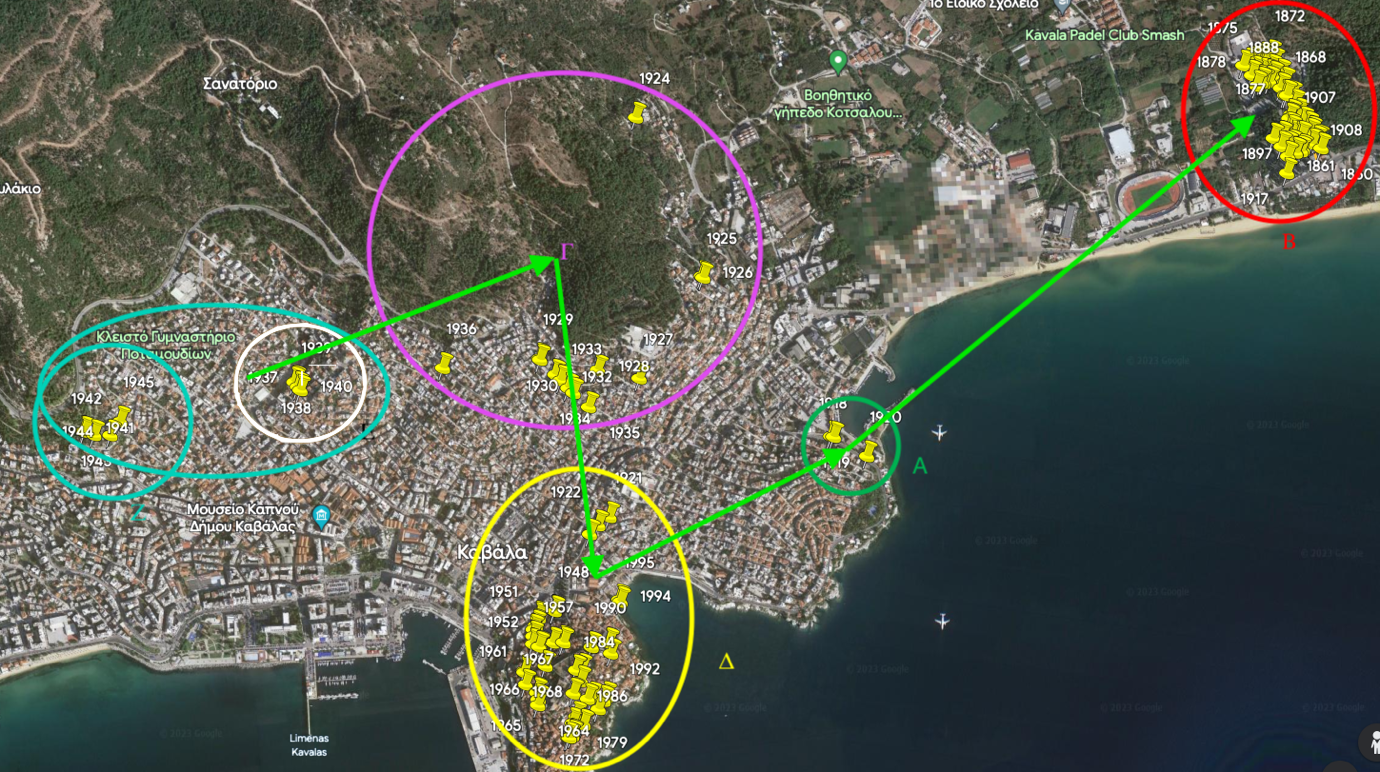
Εικόνα 2: ομαδοποιηση γειτονικων σημειων

ΑΠΟΤΕΛΕΣΜΑ

Με αυτόν τον τροπο επιτυγχανουμε την υποδιαιρεση του προβλματος σε μικροτερα προβληματα για πιο ευκολη διαχειρηση των πολλων δεδομενων και ταχυτερη εφαρμογη του αλγοριθμου . αργοτερα συνδιαζοντας τα αποτελεσματα που θα λαβουμε από κάθε ομαδα , μπορουμε να εξαγουμε ένα τελικο συμπερασμα για το προβλημα συνολικα

2.ΕΠΙΛΟΓΗ ΣΗΜΕΙΩΝ ΕΙΣΟΔΟΥ-ΕΞΟΔΟΥ

Εχοντας πλεον τις 5 ομαδες σημειων Α,Β,Γ,Δ,Ζ μπορουμε να εφαρμοσουμε τον αλγοριθμο σε κάθε ομαδα ξεχωριστα. Για να μπορουμε όμως να συνδεσουμε τα αποτελεσματα μετα ώστε να εχουμε μια βελτιστη λυση για το συνολο του προβληματος , θα πρεπει πρωτα να βρουμε την ιδανικη σειρα με την οποια θα επισκεφθουμε τους κομβους. Θα υποθεσουμε στην περιπτωση μας ότι η βελτιστη σειρα είναι η Ζ,Γ,Δ,Α,Β όπως φαινεται στην εικονα παρακατω

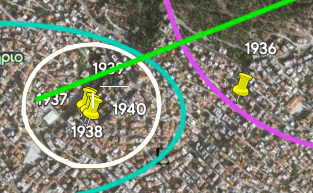


Εικόνα 3: Γενικη βελτιστη διαδρομη επισκεψης ομαδων

Επειτα , θα θεσουμε ένα σημειο εισοδου και ένα σημειο εξοδου για την κάθε ομαδα. με αυτό τον τροπο μπορουμε να δειξουμε στον αλγοριθμο από ποιο σημειο θελουμε να ξεκιναει την ευρεση του και σε ποιο σηημειο θελουμε να σταματαει , εμεσα δηλαδη θα συνδεσουμε τους κομβους μεταξυ τους. Η διαδικασια είναι ιδια για ολους τους κομβους οποτε θα δωσουμε ένα παραδειγμα της πρωτης συνδεσης για να κατανοησουμε το σκεπτικο.

ΠΗΓΑΙΝΟΝΤΑΣ ΑΠΟ ΤΟΝ Ζ ΣΤΟΝ Γ

Ο κομβος Ζ είναι και η αρχη του ααλγοριθμου μας οποτε μπορουμε να ορισουμε οποιο σημειο εκκινησης θελουμε . στην πραξη όμως το σημειο που μας εξυπηρετεί καλυτερα είναι το 1941 αφου μπορουμε ευκολα να φανταστουμε την βελτιστη νοητη γραμμη για την βελτιστη διαδρομη μεσα στον Ζ . αυτό που μας μενει τωρα είναι να βρουμε σε ποιο σημειο θα τερματιζει ο αλγοριθμος



Εικόνα 4: κοντινα σημεια Ζ και Γ

Παρατειρουμε ότι το κοντινοτερο σημειο του κομβου Γ στον Ζ είναι το 1936. Συνεπως θα πουμε στον αλγοριθμο ότι θελουμε το σημειο τερματισμου να είναι αυτό. Ετσι , όταν τρρεξουμε ττον αλγοριθμο για την ομαδα Ζ θα βρουμε την βελτιστη διαδρομη που ξεκκιναει από το σημειο 1941 και τερματιζει στην αρχη του επομεενου κομβου , του Γ , στο σημειο 1936

Χρησημοποιοντας την ιδια μεθοδο θα ορισουμε για κκαθε ομαδα την επιθυμητη βελτιστηη εισοδο και εξοδο , τις οποιες μπορουμε ευκολα να διακρινουμε βαση την αρχικη γενικη σειρα επισκεψης που θελουμε να ακολουθηησουμε . ο παρακατω πινακας δειχνει εισοδους-εξοδους και συνδεσεις μεταξυ κομβων.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Ζ->Γ | Γ->Δ | Δ->Α | Α->Β |
| Είσοδος | 1941 | 1936 | 1921 | 1920 |
| Έξοδος | 1936 | 1921 | 1920 | 1917 |

2.ΕΠΙΛΟΓΗ ΚΑΤΑΛΛΗΛΟΥ ΑΛΓΟΡΙΘΜΟΥ

Αφου εχουμε κανει όλα τα παραπανω βηματα , τωρα πρεπει να βρουμε τον αλγοριθμο που θα αξιοποισουμε για να προσεγγησουμε την βελττιστηη δυναττη λυση του προβληματος .

Επιλεξαμε τον Ant Colony Organization (ACO) καθως είναι ένας πολύ γνωστος γνωστος αλγοριθμος για την ευρεση της βελτιστης διαδρομης σεε προβληηματα TSP (Traveling Salesman Problem)

Ο ACO (Ant Colony Optimization) είναι ένας βασισμένος στη φυσική μεταφορέας βαθμωτός αλγόριθμος που εμπνέεται από τη συμπεριφορά μυρμηγκιών κατά την αναζήτηση τροφής. Αναπτύχθηκε από τον Μάρκο Ντόριγκο (Marco Dorigo) τη δεκαετία του 1990.

Ο αλγόριθμος ACO χρησιμοποιεί έναν αριθμό εικονικών μυρμηγκιών που κινούνται πάνω σε ένα γράφο, ακολουθώντας κάποιους κανόνες κίνησης και επικοινωνίας μεταξύ τους. Οι μυρμήγκια αφήνουν φερομόνες κατά μήκος των μονοπατιών που διανύουν, και αυτές οι φερομόνες χρησιμοποιούνται από τα μυρμήγκια για να επιλέξουν την επόμενη κίνησή τους.

Τα μυρμήγκια τείνουν να ακολουθούν μονοπάτια με υψηλότερη συγκέντρωση φερομόνης, και καθώς τα μυρμήγκια ανακαλύπτουν καλύτερες λύσεις, αυξάνεται η συγκέντρωση της φερομόνης σε αυτά τα μονοπάτια. Στη διάρκεια του χρόνου, ο αλγόριθμος συγκλίνει σε ένα καλό μονοπάτι που οδηγεί σε μια βέλτιστη λύση.

Ο ACO εφαρμόζεται ευρέως σε προβλήματα βελτιστοποίησης, όπως το πρόβλημα του περιπλανώμενου πωλητή (Travelling Salesman Problem), όπου ο πωλητής πρέπει να επισκεφτεί μια σειρά από πόλεις με το συντομότερο δυνατόν μονοπάτι.

ΛΟΓΟΙ ΕΠΙΛΟΓΗΣ ΑΛΓΟΡΙΘΜΟΥ ACO

Ο αλγόριθμος ACO είναι κατάλληλος για το πρόβλημα του περιπλανώμενου πωλητή (TSP) για διάφορους λόγους:

1. Αποκέντρωση και παραλληλισμός: Ο ACO λειτουργεί με ένα σύνολο εικονικών μυρμηγκιών που ανεξάρτητα εξερευνούν το πρόβλημα. Αυτό επιτρέπει την παράλληλη αναζήτηση πολλαπλών λύσεων και μπορεί να οδηγήσει σε γρήγορη σύγκλιση προς μια βέλτιστη λύση.

2. Υποκειμενική πληροφορία: Τα μυρμήγκια αφήνουν φερομόνες κατά μήκος των μονοπατιών που διανύουν. Η συγκέντρωση των φερομόνων καταγράφει την πληροφορία για τις προηγούμενες επιτυχημένες λύσεις. Αυτή η υποκειμενική πληροφορία παρέχει κατευθυντήριο σήμα για την εύρεση βέλτιστων λύσεων καθώς ο χρόνος προχωράει.

3. Επιλογή μονοπατιού με βάση τη φερομόνη: Τα μυρμήγκια έχουν την τάση να ακολουθούν μονοπάτια με υψηλότερη συγκέντρωση φερομόνης. Καθώς ο αλγόριθμος εξερευνά τον χώρο λύσεων, η συγκέντρωση των φερομόνων αυξάνεται σε καλύτερες λύσεις, προτείνοντας οδηγίες για την επιλογή των μονοπατιών που οδηγούν σε βέλτιστες λύσεις.

Αυτοί οι παράγοντες καθιστούν τον αλγόριθμο ACO αποτελεσματικό για το πρόβλημα TSP, όπου χρειάζεται να βρεθεί το συντομότερο μονοπάτι που επισκέπτεται κάθε πόλη μόνο μία φορά. Η δυνατότητα παράλληλης αναζήτησης και η χρήση φερομόνων παρέχουν έναν καλό τρόπο για την εξερεύνηση του διασταύρωσης μεταξύ πολλαπλών πιθανών λύσεων και την σύγκλιση προς τη βέλτιστη λύση.

ΕΞΗΓΗΣΗ ΑΛΓΟΡΙΘΜΟΥ ΣΕ PYTHON

Οριζουμε ένα ‘αντικειμενο’ (class) ACO , το οποιο αντιπροσωπευει κάθε μυρμιγκι και περιεχει τις παρακατω μεταβλητες :

* Coordinates : παιρνει ως ορισμα τον πινακα με τις συνττεταγμενες κάθε ομαδας
* Num\_cities : το ποσα σημεια εχει η ομαδα που χρησημοποιουμε
* Distances : υπολογιζει ττην αποσταση μεταξυ δυο συντεταγμενων καλωντας μια συναρτηση (def distance) που θα δουμε παρακατω
* Pheromones : ποσοτητα φερομονης που θα αφηνει πισω του ένα μυρμήγκια
* Alpha & beta : θα δουμε αναλυτικά για τις δυο αυτές μεταβλητες παρακατω
* Evaporation\_rate : με τι ρυθμο θα εξατμίζεται φερομονη από τα σημεια αφου καποιο μυρμιγκι φυγει
* Num\_ants : ο αριθμος των μυρμιγκιων που θα χρησημοποιησουμε
* Num\_iterations : ποσες επαναληψεις θα γινουν. περισσοτερα iterations σημαινουν μεγαλυτερη πιθανοτηητα να βρεθει η καλυτερη δυνατηη λυση. Το κάθε iteration βασιζει τις προσπαθιες του συμφωνα με τα δεδομενα των προηγουμενων του.

Επιπλεον, η ACO εχει και τις ακολουθες συναρτησεις:

* -‘find\_optimal\_path`: Αυτή η μέθοδος βρίσκει την βέλτιστη διαδρομή μεταξύ μιας αρχικής πόλης και μιας τελικής πόλης χρησιμοποιώντας τον αλγόριθμο ACO. Εκτελεί έναν καθορισμένο αριθμό επαναλήψεων και ενημερώνει την βέλτιστη διαδρομή και απόσταση αν βρεθεί μια μικρότερη διαδρομή.
* - `construct\_paths`: Αυτή η μέθοδος κατασκευάζει πολλαπλές διαδρομές χρησιμοποιώντας τον αλγόριθμο ACO. Επιλέγει επαναληπτικά την επόμενη πόλη που θα επισκεφθεί βάσει των επιπέδων φερομόνης και των αποστάσεων, εξασφαλίζοντας ότι κάθε πόλη επισκέπτεται μόνο μία φορά.
* - `construct\_path`: Αυτή η μέθοδος κατασκευάζει μια μεμονωμένη διαδρομή επιλέγοντας την επόμενη πόλη που θα επισκεφθεί βάσει των επιπέδων φερομόνης και των αποστάσεων των μη επισκεφθέντων πόλεων.
* - `calculate\_path\_distance`: Αυτή η μέθοδος υπολογίζει την συνολική απόσταση μιας δοθείσας διαδρομής, προσθέτοντας τις αποστάσεις μεταξύ διαδοχικών πόλεων στη διαδρομή.
* - `update\_pheromones`: Αυτή η μέθοδος ενημερώνει τα επίπεδα φερομόνης βάσει των διαδρομών που διανύθηκαν και των αντίστοιχων αποστάσεων τους. Υπολογίζει τη νέα τιμή φερομόνης προσθέτοντας το αντίστροφο της απόστασης της διαδρομής.
* - `plot\_optimal\_path`: Αυτή η μέθοδος δημιουργεί ένα γράφημα που απεικονίζει τη βέλτιστη διαδρομή που βρέθηκε.

Υπάρχει ένα παράδειγμα χρήσης όπου δημιουργείται ένα αντικείμενο της κλάσης `ACO`, καθορίζονται η αρχική και τελική πόλη, και στη συνέχεια καλούνται οι μέθοδοι `find\_optimal\_path` και `plot\_optimal\_path` για να βρεθεί η βέλτιστη διαδρομή και να απεικονιστεί σε ένα γράφημα.

ΟΙ ΠΑΡΑΜΕΤΡΟΙ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ ΚΑΙ Η ΣΗΜΑΣΙΑ ΤΟΥΣ

Στον αλγόριθμο ACO, οι παράμετροι `alpha` και `beta` επηρεάζουν την ισορροπία μεταξύ των φαινομένων φερομόνης και των αποστάσεων όταν τα μυρμήγκια επιλέγουν την επόμενη πόλη που θα επισκεφθούν κατά τη διάρκεια της κατασκευής των μονοπατιών.

- Η παράμετρος `alpha` ελέγχει τη σημασία των φαινομένων φερομόνης. Υψηλές τιμές του `alpha` ενθαρρύνουν την εξερεύνηση του χώρου αναζήτησης με βάση τις πληροφορίες της φερομόνης, προτιμώντας πιθανώς μονοπάτια με υψηλότερα επίπεδα φερομόνης.

- Η παράμετρος `beta` ελέγχει τη σημασία των αποστάσεων μεταξύ των πόλεων. Υψηλές τιμές του `beta` δίνουν προτεραιότητα σε μικρότερα μονοπάτια βάσει των αποστάσεων, προτιμώντας πιθανώς πιο άμεσες διαδρομές.

Αλλάζοντας τις τιμές του `alpha` και `beta`, μπορουμε να ελέγξουμεε την ισορροπία μεταξύ της εξερεύνησης (αναζήτηση νέων μονοπατιών) και της εκμετάλλευσης (ενίσχυση σύντομων μονοπατιών).

Αλλαγές στον ρυθμό εξάτμισης της φερομόνης (`evaporation\_rate`) επηρεάζουν τον ρυθμό με τον οποίο μειώνονται τα επίπεδα φερομόνης με την πάροδο του χρόνου. Υψηλότερες τιμές του ρυθμού εξάτμισης οδηγούν σε ταχύτερη μείωση των φερομόνης, μειώνοντας την επίδρασή τους στις μελλοντικές αποφάσεις των μυρμηγκιών. Χαμηλότερες τιμές, από την άλλη πλευρά, οδηγούν σε πιο αργή μείωση, επιτρέποντας στα φαινόμενα φερομόνης να διατηρούνται για μεγαλύτερο διάστημα.

Ο αριθμός των επαναλήψεων (`num\_iterations`) καθορίζει πόσες φορές ο αλγόριθμος θα κατασκευάσει μονοπάτια και θα ενημερώσει τα επίπεδα φερομόνης. Η αύξηση του αριθμού των επαναλήψεων επιτρέπει περισσότερη εξερεύνηση του χώρου αναζήτησης, με την πιθανότητα βελτίωσης των λύσεων. Ωστόσο, αυξάνει επίσης τον χρόνο υπολογισμού.

Ο αριθμός των μυρμηγκιών (`num\_ants`) καθορίζει πόσα μυρμήγκια θα συμμετάσχουν στην κατασκευή μονοπατιών κατά κάθε επανάληψη. Αυξάνοντας τον αριθμό των μυρμηγκιών, αυξάνεται η πιθανότητα εξερεύνησης περισσότερων μονοπατιών και ενδέχεται να βελτιωθούν οι επιλύσεις. Ωστόσο, αυξάνει επίσης τον χρόνο υπολογισμού και την απαιτούμενη μνήμη.

Αλλαγές σε αυτές τις παραμέτρους μπορούν να επηρεάσουν την απόδοση του αλγορίθμου και την εύρεση βέλτιστων λύσεων. Οι καλύτερες τιμές για αυτές τις παραμέτρους εξαρτώνται από τη φύση του προβλήματος και πρέπει να προσαρμοστούν πειραματικά.

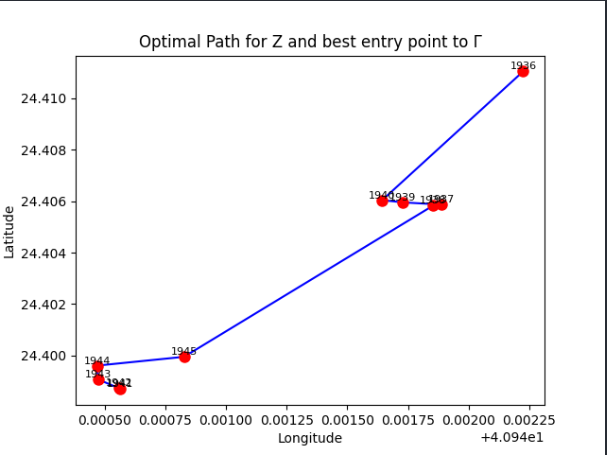
Στην περιπτωση μας , βρηκαμε μετα από αρκετες πειραματικες εκτελεσεις του αλγοριθμου, ότι οι παρακατω τιμες μας εδωσαν τα καλυτερα αποτελέσματα :

* Alpha = 2
* Beta = 5
* Evaporation\_rate = 0.5
* Num\_ants = 100
* Num\_iterations = 300

Οι τιμες για τις Num\_iterations και Num\_ants μειωθήκαν σε ένα βαθμο όταν ο αλγοριθμος επεξεργαζοτταν κομβους με μικροτερο πληθος σημειων.

ΑΠΟΤΕΛΕΣΜΑΤΑ ΑΛΓΟΡΙΘΜΟΥ

Ελαχιστη διαδρομη από τον Ζ στον Γ (αρχη: 1941 , τελος : 1936)



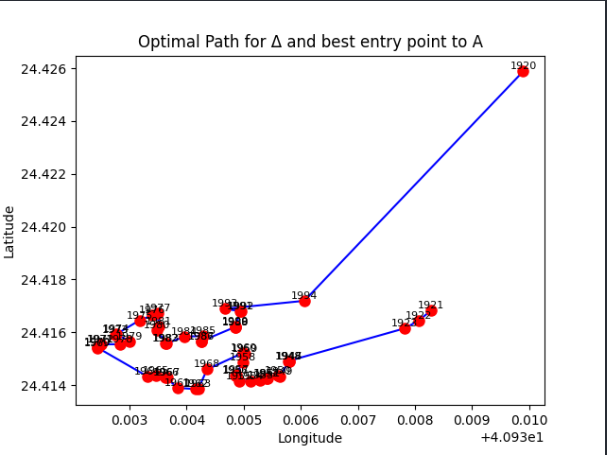
Απόσταση : 1.40 χλμ

Ελαχιστη διαδρομη από τον Γ στον Δ (αρχη: 1936 , τελος : 1921)



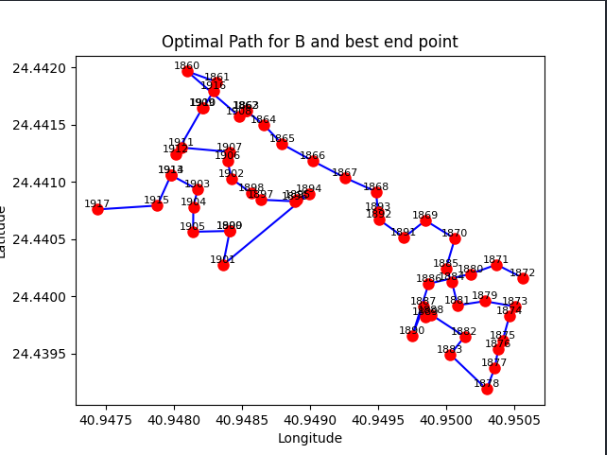
Απόσταση : 2,7 χλμ

Ελαχιστη διαδρομη από τον Δ στον Α (αρχη: 1921 , τελος : 1920)



Απόσταση : 2,9 χλμ

Ελάχιστη διαδρομη από τον Α στον Β (αρχη: 1920 , τελος : 1917)



Απόσταση : 1,29 χλμ

Τελική διαδρομή : 1,40 + 2,7 + 2,9 + 1,29 = 8,29 χλμ

Αποθετήριο github : https://github.com/Nick7255/ACO\_implementation\_omada\_22.git