Project Εργασία 3 Παρατηρήσεις σχετικά με τον Ant colony.

Ο αλγόριθμος Ant colony σε σχέση με τους άλλους αλγορίθμους δεν μπορεί να μπει στο cutoff μετά τα 45 σημεία χωρίς να γίνουν αρκετές αλλαγές στο θέμα της ακρίβειας και της δομής του αλγορίθμου, αλλά κάναμε κάποιες παρατηρήσεις για την συμπεριφορά του και το πως μπορεί να "αλλαξει" χρονική πολυπλοκότητα για ακρίβεια αποτελεσμάτων. Επίσης να αναφερθεί ότι λόγω της "τυχαιοτητας του αλγορίθμου κάποια αποτελέσματα μπορεί να είναι διαφορετικά ανά επανάληψη, ειδικά όσον αφορά τα breaks και κάποια αποτελέσματα στον πίνακα είναι "outliers" για αυτό τον λόγο επίσης.

Αλλαγές όσον αφορά τον κώδικα:
Παρατηρήσαμε ότι αρκετό χρόνο παίρνει η κατασκευή των πολύγωνων ανά βήμα και επειδή πρέπει να φτιάξει κάθε πιθανό πολύγωνο για ένα σημείο και κάθε πολύγωνο για όλα τα σημεία. Έτσι σκεφτήκαμε 2 επιλογές, να μην γίνεται έλεγχος για όλες τις ακμές και να μην ελέγχονται όλα τα σημεία. Ειδικά:
-Ο αλγόριθμος πριν για ένα σημείο έφτιαχνε όλα τα δυνατά πολύγωνα που μπορούσε, δηλαδή προσπαθούσε να σπάσει κάθε ακμή του πολυγώνου για να βάλει το νέο σημείο. Αυτό παρατηρήσαμε ότι δεν δίνει και τόσο μεγάλες διάφορες στο θέμα ακρίβειας άρα το αλλάξαμε για να διαλέγει την πρώτη διαθέσιμη ακμή (δηλαδή την πρώτη ακμή που δεν έχει εσωτερικό σημείο κάποιο άλλο σημείο του σημειοσυνόλου). Αξίζει να αναφερθεί εδώ ότι ίσως μια ευρετικη εδώ να έδινε καλύτερα αποτελέσματα αλλά δεν μπορέσαμε να βρούμε έναν τρόπο υπολογισμού της που να είχε βελτίωση. Αντίθετα μπορέσαμε να βρούμε μια για το δεύτερο κομμάτι, το πλήθος των σημείων που ελέγχονται. Ο αλγόριθμος κανονικά έφτιαχνε για κάθε σημείο στο χώρο όλα τα πολύγωνα, αλλά αλλάζοντας στο έτσι ώστε να υπολογίζει σε έναν πίνακα τα εμβαδά και να βρίσκει το μέσο εμβαδόν των τριγωνων που μπήκαν και μετά από ένα δωθεντο σημείο (πχ την μέση 'η τα ¾ που δοκιμάστηκαν) να αρχίζει να βλέπει την σχέση του μέσου εμβαδού με το εμβαδόν του τριγωνου και με μια πιθανότητα να το διαλέγει και να σταματάει νωρίτερα. Είδαμε ότι όσο αυξάνεται το ποσοστό του συνόλου που βλέπουμε τόσο αυξάνεται ο χρόνος αλλά και η ακρίβεια. Αξίζει να σημειωθεί ότι αυτό είναι ένα greedy approach στον ant colony και αυτό σημαίνει ότι υπάρχει μεγάλη πιθανότητα να απομακρυνθεί από την ολική λύση και να "κολλησει" σε έναν υπόχωρο λύσεων.

Άρα μια περίληψη των παραπάνω είναι ότι για να μπορέσουμε να βελτιώσουμε τον χρόνο, έτσι ώστε να μπορέσουμε να δώσουμε τιμές στα L και K μεγαλύτερες έπρεπε να περιορίσουμε τον χώρο αναζήτησης του αλγορίθμου.

Τώρα όσον αφορά για εμάς την πιο σημαντική παράμετρο, το L. Το L γενικά είναι οι κύκλοι που θα κάνουν τα μυρμήγκια μέχρι να τελειώσει ο αλγόριθμος. Στα papers και στις διαλέξεις συνησταται L>=10 αλλά αυτό είναι ένα νούμερο που δεν μπορεί να φτάσει ποτέ το cutoff. Αν θέσουμε L=1 τότε έχουμε απλά ένα AI agent algorithm, δηλαδή "αφηνουμε" K agents να ψάξουν για λύση και διαλέγουμε την καλύτερη που βρήκαμε. Με L>=2 έχουμε την φερομόνη να λειτουργεί, δηλαδή να "κατευθύνει" τα μυρμήγκια σε καλύτερες λύσεις. Όσο πιο μεγάλο είναι το σύνολο σημείων μας και όσο πιο μεγάλο είναι το L, τόσο πιο καλά λειτουργεί ο αλγόριθμος. Όμως για το cutoff αυτά δεν είναι εφικτά νούμερα. Θεωρούμε ότι είναι ο καλύτερος αλγόριθμος

εάν του δοθούν αρκετές επαναλήψεις, σε αρκετά μεγάλους χώρους και αρκετά iterations αλλά αυτό ξεφεύγει από τα χρονικά πλαίσια της εργασίας.

Τέλος έχουμε το K, που είναι το πλήθος των μυρμηγκιών. Στην εργασία 2 είχαμε όσα είναι το πλήθος των σημείων /5, εδώ έχουμε είτε /4 είτε /6. Η σωστή προσέγγιση του K είναι λογαριθμικά, δηλαδή για μικρά σημειοσύνολα να είναι αρκετά μεγάλο ποσοστό και όσο μεγαλώνουν να γίνονται όλο και λιγότερα αναλογικά με το μέγεθος της εισόδου γιατί είναι η πιο χρονοβόρα παράμετρος μετά το L αλλά δεν αυξάνει την ακρίβεια όπως το L.

Το elitism αξίζει να σημειωθεί ότι βοηθάει αρκετά την ακρίβεια χωρίς να επηρεάζει παρά πολύ τον χρόνο, όπως βλέπουμε από τις 2 τελευταίες στήλες στον πίνακα. Όμως όσο πιο μεγάλα είναι τα δεδομένα και όσο περισσότερα μυρμήγκια τόσο πιο χρονοβόρο θα γίνεται.Επίσης οσο πιο μικρό το σημειοσύνολο τοσο πιο καλό είναι το elitism,αλλά για μεγάλα σημειοσύνολα είναι καλύτερο να μην γίνεται αν θελουμε ακρίβεια.

Υλοποίηση σε θέμα κωδικά:
Στην συνάρτηση GenerateX βάλαμε κάποια breaks μέσα σε ifs για να μπορέσουμε να κάνουμε το greedy approach. Το k υπολογίζεται με 1 διαίρεση αλλά σε περίπτωση μεγαλύτερων σημειοσυνόλων (500+) θέλουμε να υπολογίζεται λογαριθμικά (log (n) αντί για n/4).Τα breaks γίνονται “on and off” μέσω μιας μεταβλήτης enable_breaks που δίνεται από την main σαν preprocess.

247 γραμμή :

```
int
i=0
;
                for(auto
       v3=space.edges_begin();v3!=space.edges_end();++v3,i++)
                if(v3[0]==s[0]){
                  pos=i+1;
                  break;
                }
```

Που αντι να δει όλα τα edges σταματάει στο πρώτο που είναι αποδεκτό.

291 : 350 γραμμή:

```
if(k==list.size()/1.5){
for (int s=0;s<=k;s++)
{


sum+=sizecounter[s];
}


for (int s=0;s<=k;s++)
{
 flag11=0;
int var=rand()%2;
if(minmax==1){
double prob
=((sum-sizecounter[s])/sizecounter[s])/100;


 if(var<abs(prob)){
   flag11=1;
   break;
 }
}


if(minmax==0){
double prob
=((sizecounter[s])/sum-sizecounter[s])/100;


 if(var<abs(prob)){
```

```
   flag11=1;


   break;
 }
}


}


if(flag11=1)
break;


}
else if(k>list.size()/1.5)
{
   int var=rand()%2;


if(minmax==1){
double prob
=((sum-sizecounter[k])/sizecounter[k])/100;


 if(var<abs(prob)){


   break;
 }
}


if(minmax==0){
double prob
=((sizecounter[k])/sum-sizecounter[k])/100;
```

```
 if(var<abs(prob)){


   break;
 }
}


}
```

Που φτιάχνουμε έναν πίνακα με τα εμβαδά για κάθε πολύγωνο, υπολογίζουμε το μέσο εμβαδόν, βλέπουμε αν κάποιο από αυτά με πιθανότητα είναι “καλο” και μετά ελέγχουμε όλα τα υπόλοιπα με βάση το μέσο εμβαδόν.

Εδώ βρίσκονται κάποιες από τις δοκιμές που κάναμε.

	Max ,L=3 ,k=n/6	Min,L =3,k= n/6	Min,L =3,k= n/6, breaks enabl ed at 2	Min,L= 3,k=n/ 6, breaks enable d at 1.2	Min,L= 4,k=n/ 6, breaks enable d at 2	Min,L= 4,k=n/ 4, breaks enable d at 2	Max,L =4,k=n /4, breaks enable d at 2	Max,L =4,k=n /4, 2, elitism =1	Max,L =4,k=n /4, 2, elitism =1 with breaks enable d at 2
euro-nig ht-10	0 secs, 0.69 score	0 secs, 0.49 score	0 secs, 0.49 score	1 sec, 0.35 score	0 secs, 0.35 score	0 secs, 0.35 score	0 secs, 0.51 score	0 secs, 0.78 score	0 secs, 0.77 score
euro-nig ht-15	0 secs, 0.62 score	1 sec, 0.5 score	0 secs, 0.51 score	0 secs, 0.50 score	0 secs, 0.41 score	0 secs, 0.38 score	0 secs, 0.52 score	0 secs, 0.73 score	0 secs, 0.69 score
euro-nig ht-20	0 secs, 0.79 score	0 secs, 0.5 score	0 secs, 0.5 score	0 secs, 0.41 score	0 secs, 0.42 score	1 secs, 0.41 score	1 sec, 0.6 score	3 secs, 0.81 score	0 secs, 0.76 score
euro-nig ht-25	1 sec, 0.64 score	2 secs, 0.42	1 sec, 0.45 secs	1 secs, 0.3 score	0 secs, 0.2 score	1 secs, 0.3 score	1 sec, 0.42 score	7 secs, 0.71 score	1 secs, 0.57 score

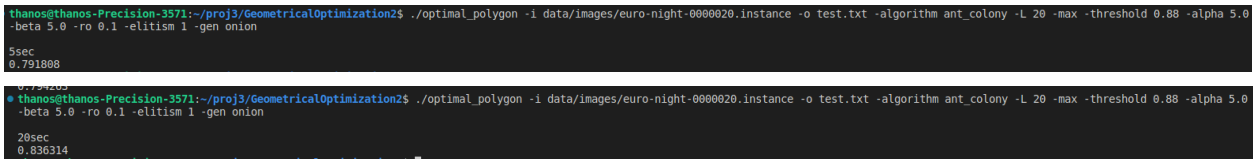
		score							
euro-night-30	3 secs, 0.72 score	3 secs, 0.53 score	3 secs, 0.54 score	3 secs, 0.52 score	2 secs, 0.52 score	3 secs, 0.51 score	3 secs, 0.63 score	24 secs, (0 real score), 0.76 score	3 secs, 0.77 score
euro-night-35	7 secs, 0.73 score	6 secs, 0.39 score	2 secs, 0.49 score	5 secs, 0.38 score	3 secs, 0.33 score	6 secs, 0.33 score	7 secs, 0.7 score	-	7 secs, 0.76 score
euro-night-40	13 secs, 0.74 score	10 secs, 0.4 score	5 secs, 0.44 score	9 secs, 0.4 score	8 secs, 0.4 score	18 secs, 0.4 score	17 secs, 0.75 score	-	18 secs, 0.80 score
euro-night-45	28 secs, (0 real score), 0.69 score	27 secs, (1 real score), 0.41 score	10 secs, 0.4 score	20 secs, 0.5 score	18 secs, 0.54 score	40 secs, (1 real score), 0.43 score	37 secs, (0 real score), 0.68 score	-	34 secs, (0 real score), 0.72 score

Τέλος όσον αφορά τα a,b,r έγιναν αυτές οι δοκιμές στα σημειοσύνολα των 10,15 και 20.
Έχουμε L=4,χωρίς breaks με K=n/4.

Max	Default values from above	a=2,b=3,r=0.03	a=2,b=5,r=0.03	a=1,b=3,r=0.1	a=2,b=3,r=0.09	a=3,b=3,r=0.05	a=5,b=3,r=0.05	a=0.5, b=3,r=0.05	a=5,b=5,r=0.1
euro-night-10	0.77 score	0.77 score	0.77 score	0.77 score	0.77 score	0.77 score	0.77 score	0.77 score	0.77 score
euro-night-15	0.69 score	0.66 score	0.66 score	0.65 score*	0.65 score	0.7 score	0.67 score	0.61 score	0.65 score
euro-night-20	0.76 score	0.75 score *	0.7 score*	0.75 score	0.73 score*	0.76 score	0.73 score*	0.77 score	0.78 score

*Είχε τεράστια διακύμανση.

Για L=20 είχαμε για το euro-night-20 score 0.79 αλλά 5 secs runtime με breaks,
0.85 αλλά με 20 secs runtime χωρίς τα breaks.Εδώ φαίνεται καλά η δύναμη ενός μεγάλου L.



Παρατηρούμε ότι δεν έχει μεγάλες διαφορές με βάση τα a,b,r για τα max.
Για τα min:

Min	Default values from above	a=2,b=3,r=0.03	a=2,b=5,r=0.03	a=1,b=3,r=0.1	a=2,b=3,r=0.09	a=3,b=3,r=0.05	a=5,b=3,r=0.05	a=0.5, b=3,r=0.05	a=5,b=5,r=0.1
euro-night-10	0.49 score	0.49 score	0.49 score	0.48 score	0.49 score	0.49 score	0.49 score	0.49 score	0.49 score

Σημείωση:
Κάλο είναι να αναφερθεί ότι έχει αρκετά μεγάλη διακύμανση σαν αλγόριθμος όσον αφορά και τον χρόνο και την ακρίβεια του, άρα τα πινακακια είναι μια εικόνα που προσεγγίζει αυτό που περιμέναμε. Επειδή έγινε τροποποίηση του αλγορίθμου με greedy approach αυτό δίνει ένα “random factor” στα αποτελέσματα.

Σύνοψη:
Είδαμε ότι σαν αλγόριθμος ο ant δεν μπορεί εύκολα να πιάνει τα cutoffs άρα έπρεπε να βρεθεί μια ευρετικη λύση. Αρκετές δοκιμάστηκαν αλλά οι 2 καλύτερες ήταν να σταματάει την αναζήτηση ακμής που θα βάλει εκεί το νέο σημείο (ίσως μπορεί να γίνεται και αυτό ευρετικα, δηλαδή αντί να σταματάει στην πρώτη να σταματάει με μια πιθανότητα αναλόγως με τι εμβαδόν μπήκε) και να σταματάει με πιθανότητα να δοκιμάζει σημεία στο χώρο για να περιοριστεί λίγο ο χώρος αναζήτησης. Έπειτα με χαμηλά L και λίγα ants μπορεί να φτάσει το cutoff απλά όσο πιο κάτω πάνε αυτές οι μεταβλητές τόσο χάνει το νόημα του ο αλγόριθμος και καταλήγει σαν ένας απλός agent (να ψάχνει k λύσεις και μετά να διαλέγει την καλύτερη), αντί να χρησιμοποιεί τις φερομόνες για να περιορίσει έτσι τον χώρο αναζήτησης. Μας φάνηκε περίεργο ότι δεν έγιναν μεγάλες διάφορες με αλλαγές στα a, b, r. Ίσως φταίει ότι χρειάζονται αρκετά μεγάλα σημειοσύνολα για να αρχίσουν να φαίνονται οι διάφορες. Γενικά δεν προτείνουμε τον ant σε περιπτώσεις περιορισμού χρόνου αλλά θεωρούμε ότι δοθέν αρκετό χρόνο είναι ο καλύτερος αλγόριθμος εύρεσης λύσης όσο πιο κοντά στην βέλτιστη γίνεται. Επίσης να σημειωθεί ότι επειδή προσπαθήσαμε να μειώσουμε την χωρική πολυπλοκότητα του (δεν κρατάμε σε 1 δομή τα πάντα αλλά σε πολλές μικρότερες δομές που περιέχουν ακριβώς την πληροφορία που θέλουμε) ίσως αυτό να επηρέασε την χρονική πολυπλοκότητα του.