ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Εύρεση βέλτιστης διαδρομής σε δίκτυο πόλεων με πολλαπλά μέσα μεταφοράς, με Μοντέλο Μικτού Ακέραιου Προγραμματισμού»

ΚΑΜΠΙΤΣΗΣ ΔΗΜΗΤΡΙΟΣ

Επιβλέπων : Δρ. ΣΑΧΑΡΙΔΗΣ ΓΕΩΡΓΙΟΣ

ΒΟΛΟΣ 2014

"Κάθε
αποχωρισμός είναι ένας μικρός θάνατος"
George Eliot, 1819-1880

Εγκρίθηκε από τα Μέλη της Τριμελούς Εξεταστικής Επιτροπής:

Πρώτος Εξεταστής (Επιβλέπων)	Δρ. Σαχαριδης Γεώργιος Λέκτορας, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας
Δεύτερος Εξεταστής	Δρ. Λυμπερόπουλος Γεώργιος Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας
Τρίτος Εξεταστής	Δρ. Κοζανίδης Γεώργιος Επίκουρος Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα της μεταπτυχιακής μου εργασίας, Λέκτορα κ. Σαχαρίδη Γεώργιο για την πολύτιμη βοήθεια και καθοδήγησή του κατά τη διάρκεια της δουλειάς μου. Επίσης είμαι ευγνώμων στα υπόλοιπα μέλη της εξεταστικής επιτροπής Καθηγητές κ. Λυμπερόπουλο Γεώργιο και κ Κοζανίδη Γεώργιο για την προσεκτική ανάγνωση της εργασίας μου και για τις πολύτιμες υποδείξεις τους. Οφείλω ένα μεγάλο ευχαριστήσω στους γονείς μου για την κατανόηση τους, ιδιαίτερα κατά τη διάρκεια των τελευταίων μηνών της προσπάθειας μου, και την υποστήριξη τους όλα αυτά τα χρόνια στους οποίους και αφιερώνω την εργασία αυτή.

Εύρεση βέλτιστης διαδρομής σε δίκτυο πόλεων με πολλαπλά μέσα μεταφοράς, με Μοντέλο Μικτού Ακέραιου Προγραμματισμού

ΚΑΜΠΙΤΣΗΣ ΔΗΜΗΤΡΙΟΣ

Πανεπιστήμιο Θεσσαλίας, Τμήμα Μηχανολόγων Μηχανικών, 2014

Επιβλέπων Λέκτορας: Δρ. Γεώργιος Σαχαρίδης, Λέκτορας σε Μεθόδους Επιχειρησιακής Έρευνας στη Βιομηχανική Διοίκηση

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται το πρόβλημα δρομολόγησης χρησιμοποιώντας πολλαπλά μέσα μεταφοράς με πραγματικούς χρόνους άφιξης-αναχώρησης. Συγκεκριμένα προσεγγίζεται το πρόβλημα ενός ταξιδιώτη, ο οποίος επιθυμεί να μεταβεί από μια πόλη της Ελλάδας σε μια άλλη, έχοντας τη δυνατότητα να μετακινηθεί με τέσσερα (4) μέσα μεαφοράς. Αρχικά πραγματοποιείται μια σύντομη αναφορά στη θεωρία γράφων και δικτύων καθότι η προσέγγιση του προβλήματος βασίζεται κυρίως σε αυτήν. Ακόμη στο θεωρητικό τμήμα της εργασίας παρουσιάζονται τα σημαντικότερα προβλήματα δρομολόγησης που έχουν απασχολήσει και συνεχίζουν έντονα να απασχολούν την επιστημονική κοινότητα. Εν συνεχεία, στο πειραματικό τμήμα της εργασίας, περιγράφεται η μοντελοποίηση του προαναφερθέντος προβλήματος με πρόγραμμα μικτού ακεραίου γραμμικού προγραμματισμού. Τα τέσσερα (4) διαφορετικά - εναλλακτικά μαθηματικά μοντέλα (M1, M1.1, M2, M2.1) που σχεδιάστηκαν με σκοπό την εύρεση της βέλτιστης λύσης παρουσιάζονται, αναλύονται και υλοποιούνται σε γλώσσα C ++ με τρία (3) ρεαλιστικά παραδείγματα δρομολόγησης. Στο τελευταίο κομμάτι της διπλωματικής, πραγματοποιείται μια σύγκριση και των τεσσάρων μοντέλων λαμβάνοντας υπόψη κυρίως τον υπολογιστικό χρόνο και την λύση που αποδίδει το καθένα από αυτά.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΑΚΡΩΝΥΜΙΩΝ

PoI : Point of Interest

WTP : Waiting Time at Point of Interest

ToA : Time of Arrival

O : Origin

D : Destination

TSP : Travel Salesman Problem

VRP : Vehicle Routing Problem

SPP : Shortest Path Problem

ΚΕΦΑΛΑΙΟ ΠΡΩΤΟ:
ΕΙΣΑΓΩΓΗ

1.1 Εισαγωγικά

Στη σύγχρονη ζωή, ο σχεδιασμός διαδρομής αποκτά όλο και μεγαλύτερη σημασία. Όσο τα δίκτυα μεταφοράς γίνονται όλο και πιο σύνθετα και η κινητικότητα στην κοινωνία μας πιο σημαντική, τόσο η ζήτηση για αποτελεσματικές μεθόδους στο σχεδιασμό της διαδρομής αυξάνεται ακόμη περισσότερο. Πολλές εταιρίες έχουν αναπτύξει συστήματα πλοήγησης για τα αυτοκίνητα, που έχουν ήδη καθιερωθεί ως στοιχεία του βασικού εξοπλισμού τους, τα οποία βοηθούν τον ταξιδιώτη στο σχεδιασμό διαδρομών. Παράλληλα οι περισσότερες σιδηροδρομικές καθώς και αεροπορικές εταιρίες προσφέρουν κάποιο είδος εφαρμογών σχεδιασμού διαδρομής μέσω του Διαδικτύου.

Ωστόσο, όλα αυτά τα συστήματα έχουν ένα κοινό περιορισμό: επιτρέπουν μόνο το σχεδιασμό διαδρομών με τη χρήση συγκεκριμένων μέσων αγνοώντας την ύπαρξη άλλων. Για παράδειγμα, τα sat-nav συστήματα δείχνουν μόνο το δρόμο στο οδικό δίκτυο. Αν θα θέλαμε να χρησιμοποιήσουμε τα μέσα μαζικής μεταφοράς, είμαστε αναγκασμένοι να σχεδιάσουμε μόνοι μας διαδρομή από και προς το πλησιέστερο σταθμό ή το αεροδρόμιο, καθώς οι εφαρμογές σχεδιασμού διαδρομών των μέσων μαζικής μεταφοράς συνήθως δεν περιλαμβάνουν το οδικό δίκτυο. Ακόμα χειρότερα, όταν σχεδιάζουμε μια διαδρομή αεροπορικώς θα πρέπει αρχικά να επιλέξουμε μια πτήση, εν συνεχεία το συγκεκριμένο δρομολόγιο με τρένο που φτάνει στο αεροδρόμιο την κατάλληλη ώρα και τέλος την κατάλληλη ώρα που θ' αναχωρήσουμε από το σπίτι μας ώστε να προλάβουμε το τρένο στο σταθμό.

Η δρομολόγηση πολλαπλών μέσων αναπτύχθηκε ώστε να παρέχει μια ποικιλία από τρόπους μετακίνησης στους ταξιδιώτες με απότερο σκοπό να αυξηθεί η ικανοποίηση που λαμβάνουν από την ανωτέρω διαδικασία. Ωστόσο η ύπαρξη διαφορετικών προτιμήσεων σε ότι αφορά τόσο τα μέσα μετακίνησης όσο και τον αντικειμενικό στόχο που έχει η χρήση του κάθε μέσου, προκαλεί πολλαπλά προβλήματα. Η διαδρομή με τη χρήση πολλαπλών μέσων ορίζεται ως η διαδρομή στην οποία χρησιμοποιούνται δύο ή περισσότερα μέσα μετακίνησης ώστε να επιτευχτεί η μετακίνηση από την αφετηρία στον προορισμό (*Van Nes 2002*). Η ελεύθερη εναλλαγή από ένα μέσο σε ένα άλλο είναι ένα ουσιώδες χαρακτηριστικό.

Σκοπός ύπαρξης της δρομολόγησης πολλαπλών μέσων είναι να παρέχει την δυνατότητα στον εκάστοτε ταξιδιώτη, με τη χρήση διαφορετικών μέσων (ιδιωτικών ή δημόσιων), να μετακινηθεί από μία αφετηρία σε κάποιον προορισμό με τον βέλτιστο γι’ αυτόν τρόπο. Ο όρος βέλτιστος ή βέλτιστη διαδρομή αφορά το κριτήριο που θέτει κάθε φορά ο εκάστοτε ταξιδιώτης (κόστος,χρόνο). Επιπλέων απαίτηση του χρήστη μπορεί να είναι ένα υποσύνολο ενδιάμεσων σημείων ενδιαφέροντος (PoI), τα

οποία ο ταξιδιώτης επιθυμεί να επισκεφτεί πριν φτάσει στον προορισμό του. Τέτοια σημεία μπορεί να είναι αξιοθέατα, σταθμοί ανεφοδιασμού, αγορές και γενικότερα οτιδήποτε επιθυμεί να επισκεφτεί ο ταξιδιώτης.

Στο πλαίσιο αυτό η παρούσα διπλωματική εργασία παρουσιάζει την μοντελοποίηση με μικτό - ακέραιο προγραμματισμό ενός προβλήματος δρομολόγησης με τέσσερα διαφορετικά μέσα μεταφοράς (Τρένο - ΚΤΕΛ - Ι.Χ.-Πλοίο) στην επικράτεια της Ελλάδας. Στόχος είναι μέσω της επίλυσης του με τη βοήθεια λογισμικού C++, η εύρεση της βέλτιστης (οικονομικότερης) διαδρομής για τον ταξιδιώτη σ' ένα δίκτυο πόλεων με συγκεκριμένες ώρες αναχώρησης και άφιξης.

1.2 **Επισκόπηση Διπλωματικής Εργασίας**

Η παρούσα διπλωματική αποτελείται από έξι (6) κεφάλαια, συγκεκριμένα :

- Κεφάλαιο 1^{ο} : Στο κεφάλαιο αυτό γίνεται μια εισαγωγή περί δρομολόγησης και δρομολόγησης πολλαπλών μέσων και παρουσιάζεται η επισκόπηση της διπλωματικής.

- Κεφάλαιο 2^{ο} : Στο κεφάλαιο αυτό αναφέρονται τα βασικότερα χαρακτηριστικά της θεωρίας γράφων και δικτύων, αναπτύσσονται οι βασικότεροι ορισμοί και περιγράφονται οι τρόποι αναπαράστασης τους.

- Κεφάλαιο 3^{ο} : Το τρίτο κεφάλαιο αφορά τη δρομολόγηση. Δίνεται ένας ορισμός για το *Routing* και αναλύονται τα σημαντικότερα προβλήματα δρομολόγησης που έχουν απασχολήσει τη διεθνή επιστημονική κοινότητα και βρίσκουν εφαρμογή οπουδήποτε εμπλέκονται δίκτυα μεταφορών.

- Κεφάλαιο 4^{ο} : Στο τέταρτο κεφάλαιο και αφού έχει τελειώσει η σύντομη θεωρητική ανασκόπηση, περιγράφεται το πρόβλημα αναζήτησης της βέλτιστης διαδρομής με τη χρήση πολλαπλών μέσων μεταφοράς σε ένα δίκτυο πόλεων και παρουσιάζονται τα τέσσερα μαθηματικά μοντέλα μικτού ακέραιου γραμμικού προγραμματισμού που σχεδιάστηκαν

- Κεφάλαιο 5^{ο} : Στο κεφάλαιο αυτό παρουσιάζονται και σχολιάζονται τα πειραματικά αποτελέσματα απο την υλοποίηση των μοντέλων που παρουσιάστηκαν στο προηγούμενο κεφάλαιο πάνω σε τρία ρεαλιστικά παραδείγματα. Σχολιάζονται οι διαδρομές που προκύπτουν και οι υπολογιστικοί χρόνοι που απαιτούνται και μέσω της συγκριτικής ανάλυσης επιλέγεται το αποτελεσματικότερο μοντέλο.

- Κεφάλαοιο 6^{ο} : Στο τελευταίο κεφάλαιο αναφέρονται τα συμπεράσματα της παρούσας διπλωματικής και καταγράφονται οι τυχόν βελτιώσεις που επιδέχονται οι μοντελοποιήσεις.

- Παράρτημα : Στο παράρτημα παρατίθεται το μαθηματικό μοντέλο M1 για 30 κόμβους και οι πίνακες κόστους και χρόνου μετακίνησης απο κόμβο σε κόμβο.

ΚΕΦΑΛΑΙΟ ΔΕΥΤΕΡΟ:
ΘΕΩΡΙΑ ΔΙΚΤΥΩΝ

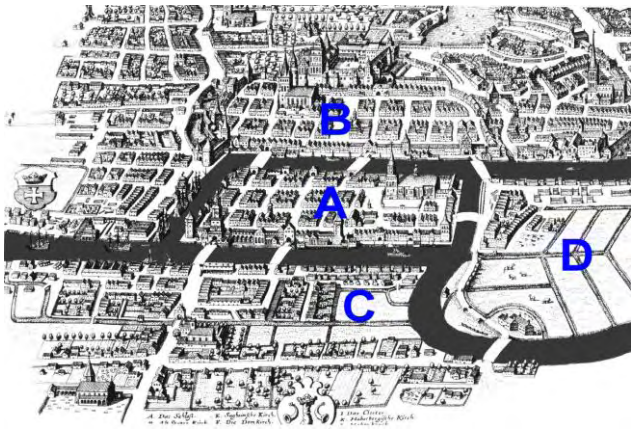
2.1 Εισαγωγή

Στο συγκεκριμένο κεφάλαιο θα γίνει μια σύντομη αναφορά στη βασική θεωρία γράφων και δικτύων δεδομένου ότι η προσομοίωση του προβλήματος δρομολόγησης που εξετάζουμε βασίζεται πάνω σ' αυτήν. Αρχικά θα γίνει μια σύντομη ιστορική αναδρομή, εν συνεχεία θα παρουσιαστούν τα βασικά χαρακτηριστικά ενός γράφου και ενός δικτύου.

2.2 Ιστορική Αναδρομή

Για την ιστορία της θεωρίας γράφων[19] θεωρείται σημαντική η μελέτη του *Leonhard Euler* για τις Επτά Γέφυρες του *Konigsberg* το 1736. Η συγκεκριμένη πόλη ήταν χτισμένη στον ποταμό Πρέγκελ, στην Πρωσία, και περιλάμβανε δύο μεγάλα νησιά, A και D, που συνδέονταν μεταξύ τους αλλά και με την ηπειρωτική χώρα B και C με επτά γέφυρες όπως φαίνεται στην εικόνα 2.1. . Το πρόβλημα ήταν κατά πόσον είναι δυνατόν να υπάρξει μια διαδρομή σύμφωνα με την οποία θα διασχίζετε κάθε γέφυρα ακριβώς μια φορά και θα επιστρέφουμε ξανά στο σημείο εκκίνησης. Η συγκεκριμένη δημοσίευση, όπως και εκείνη που γράφτηκε από τον Γάλλο χημικό *Alexandre - Théophile Vandermonde* στο μαθηματικό πρόβλημα του Ίππου στη σκακιέρα που κατευθύνεται με την ανάλυση θέσης που εισήγαγε ο *Leibniz*. Ο τύπος του *Euler*, σχετικά με τον αριθμό των ακμών, των κορυφών και των εδρών ενός κυρτού πολυέδρου μελετήθηκε από τον *Augustin Louis Cauchy* και τον *Simon Antoine Jean L'Huillier* και είναι αρχή της τοπολογίας.

Σχεδόν εκατό χρόνια μετά τη μελέτη του *Euler* και την εισαγωγή της τοπολογίας από

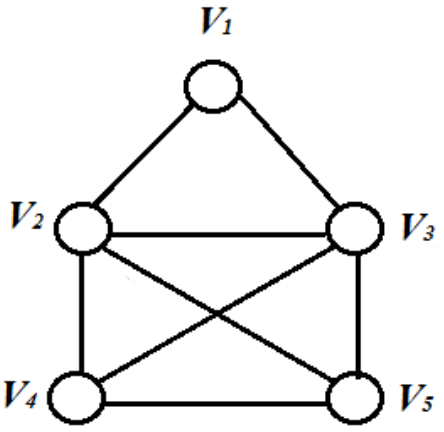


Εικόνα 2.1: Το πρόβλημα του Konigsberg

τον *Johann Benedict Listing*, ο *Arthur Cayley* μελέτησε μια ιδιαίτερη κατηγορία γράφων, τα δέντρα. Η μελέτη αυτών των ιδιαίτερων γράφων είχε πολλές εφαρμογές στη θεωρητική χημεία. Οι τεχνικές που αναπτύχθηκαν είχαν να κάνουν κυρίως με την απαρίθμηση γράφων που παρουσίαζαν κάποιες ιδιαίτερες ιδιότητες. Η απαριθμητική θεωρία γράφων ήταν ένα από τα συμπεράσματα του *Cayley* και δημοσιεύτηκε από τον *George Pólya* μεταξύ των ετών 1935 και 1937, ενώ η γενίκευση των συμπερασμάτων εκδόθηκε από τον *Nicolaas Govert de Bruijn* το 1959. Ο *Cayley* συνέδεσε τα συμπεράσματά του για τα δέντρα με τις σύγχρονες μελέτες για τη χημική σύνθεση. Η σύνθεση των μαθηματικών και των χημικών εννοιών είναι το αρχικό τμήμα της στερεότυπης (*standard*) ορολογίας της θεωρίας γράφων. Συγκεκριμένα ο όρος γράφος παρουσιάστηκε για πρώτη φορά σε μια δημοσίευση το 1878 και το πρώτο έντυπο εγχειρίδιο για Θεωρία Γράφων εκδόθηκε το 1936 από τον *Denes Konig*. Σημαντικός παράγοντας στην εξέλιξη της θεωρίας των γράφων αποτέλεσε η χρήση τεχνικών σύγχρονης άλγεβρας με το πρώτο παράδειγμα να προέρχεται από τον φυσικό *Gustav Kirchhoff* και την έρευνα που πραγματοποίησε για τον υπολογισμό της τάσης και του ρεύματος στα ηλεκτρικά κυκλώματα.

2.3 **Γράφος**

Ένας γράφος[1,17,18] συμβολίζεται ως G = {V,E}, και αποτελείται από ένα πεπερασμένο μη κενό σύνολο V = { v_1,v_2, ... , v_n } των κορυφών (*vertices*) ή κόμβων και ομοίως από ένα πεπερασμένο μη κενό σύνολο E = { e_1, e_2, ... , e_n } που αποτελούν τις ακμές (*edges*) ή γραμμές του γράφου. Κάθε ακμή ορίζεται ως ένα μη διατεταγμένο ζεύγος δύο κόμβων $E \subseteq V \times V$ και ορίζουμε ακμή από το $u \in V$ στο $v \in V$ αν και μόνο αν $(u,v) \in E$ όπως φαίνεται στον γράφο της εικόνας 2.2 και θα μπορούσαμε να πούμε ότι σε περίπτωση που οι κόμβοι χρησιμοποιούνται για την παράσταση κάποιων δεδομένων τότε και οι ακμές χρησιμοποιούνται για να απεικονίσουν κάποια σχέση που συνδέει μεταξύ τους τα δεδομένα.



$$V = \{v_1, v_2, v_3, v_4, v_5\}$$

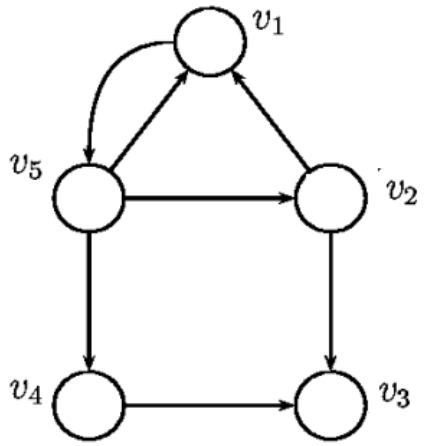
$$E = \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_5), (v_2, v_4), (v_3, v_4), (v_3, v_5), (v_5, v_4)\}$$

Εικόνα 2.3 : Γράφος

2.3.1 Ορισμοί

Οι γράφοι μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες :

- **Μη κατευθυνόμενος** (*Undirected graph)* : Είναι ο γράφος, οι ακμές του οποίου δεν είναι προσανατολισμένες και ισχύει ότι $(V_1,V_2) = (V_2,V_1)$
- **Κατευθυνόμενος** (*Directed graph)* : Οι ακμές του είναι προσανατολισμένες προς μια κατεύθυνση, δηλαδή τα ζεύγη (V_1,V_2) και (V_2,V_1) δεν είναι ισοδύναμα. Στις κατευθυνόμενες πλέον ακμές (V_1,V_2) η κορυφή V_1 ονομάζεται πηγή (*Origin)* και η V_2 προορισμός (*Destination*).
- **Μεικτός** (*Mixed Graph*) : Εάν ορισμένοι κλάδοι μόνο έχουν κατεύθυνση και κάποιοι όχι.



Εικόνα 2.4 : Κατευθυνόμενος γράφος

Οι κορυφές V_1 και V_2 λέγονται **γειτονικές** (*adjacent*) ή ότι γειτνιάζουν εάν (V_1,V_2) είναι ακμή ή πιο απλά όταν συνδέονται με ένα κλάδο και το αντίστοιχο και για δύο κλάδους οι οποίοι συνδέονται με έναν κόμβο.

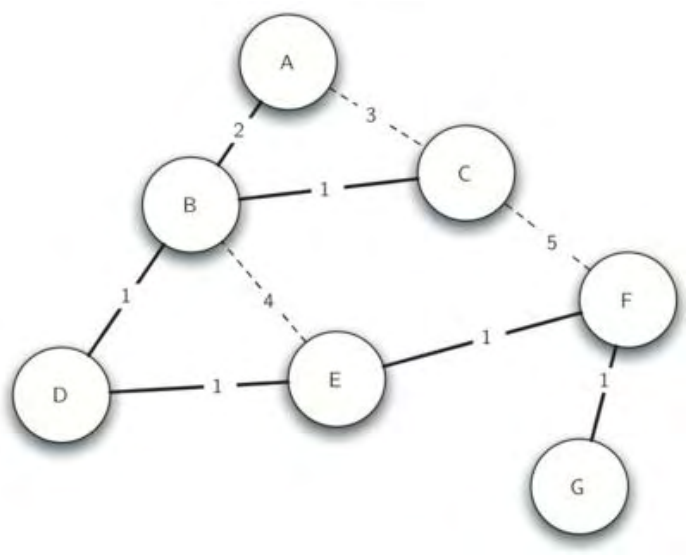
Ο **βαθμός** ενός κόμβου σε ένα μη προσανατολισμένο δίκτυο είναι ο αριθμός των ακμών που συνδέονται με αυτόν. Σε ένα προσανατολισμένο δίκτυο, ορίζεται αντίστοιχα ο βαθμός για τον αριθμό των κλάδων που καταλήγουν σε αυτό τον κόμβο (*indegree*) και ο βαθμός για τον αριθμό των κλάδων που απομακρύνονται από αυτό τον κόμβο (*outdegree*). **Τάξη** ενός γράφου ορίζεται ως το πλήθος των κόμβων του και συμβολίζεται με |V|. **Μέγεθος** ενός γράφου ορίζεται ως το πλήθος των ακμών και συμβολίζεται |E|.

Μονοπάτι ή διαδρομή (*path*) ενός γράφου μήκους n, όπου **μήκος** ενός μονοπατιού είναι ο αριθμός ακμών που περιέχει, είναι μια ακολουθία κόμβων $v_0, v_1, \dots, v_n$, όπου για κάθε i , $0 \leq i < n$, (v_i , v_{i+1}) είναι ακμή του γράφου δηλαδή μια αλληλουχία γειτονικών κλάδων και κόμβων. Σε ένα προσανατολισμένο γράφο τα μονοπάτια έχουν διεύθυνση. **Απλό** (*Simple*) ονομάζεται το μονοπάτι

αυτό που κάθε κλάδος εμφανίζεται το πολύ μια φορά και όλες οι κορυφές είναι διαφορετικές μεταξύ τους εκτός από την πρώτη και την τελευταία οι οποίες μπορεί να είναι οι ίδιες. **Κύκλος** (*Cycle*) είναι ένα μονοπάτι του οποίου ο αρχικός και ο τελικός κόμβος συμπίπτουν και αντίστοιχα απλός όταν οι κορυφές του είναι διαφορετικές.

Συνεκτικός (*Connected*) ονομάζεται ένας μη κατευθυνόμενος γράφος αν για κάθε ζεύγος κορυφών, υπάρχει μονοπάτι που τις συνδέει. Αντίστοιχα ένας κατευθυνόμενος γράφος είναι συνεκτικό αν το αντίστοιχο μη κατευθυνόμενο είναι συνεκτικό. Ο κατευθυνόμενος γράφος που ικανοποιεί την ιδιότητα αυτή ονομάζεται **ισχυρά συνεκτικός** (*strongly connected*) και αν ο μη κατευθυνόμενος στον οποίο αντιστοιχεί είναι συνεκτικός, τότε ονομάζεται **ασθενά συνεκτικός** (*weakly connected*).

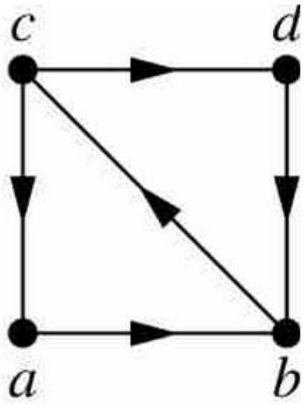
Στις περιπτώσεις που αντιστοιχίζουμε μια τιμή σε κάθε ακμή ενός γράφου, τότε ο γράφος αυτός ονομάζεται **γράφος με βάρη** και η τιμή αυτή ονομάζεται **βάρος**. Η τιμή αυτή ανάλογα με το εκάστοτε πρόβλημα μπορεί να είναι χρόνος, απόσταση, κόστος ή και οτιδήποτε άλλο μπορεί να αντιπροσωπεύσει. Αντίστοιχα όταν κάθε κορυφή χαρακτηρίζεται από μια μοναδική ετικέτα τότε έχουμε **γράφο με ετικέτες.**



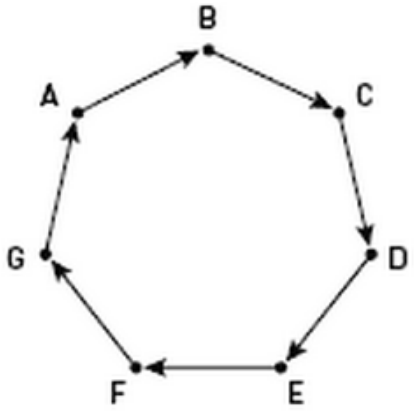
Εικόνα 2.5 : Γράφος με βάρη και ετικέτες

Κύκλος Euler ονομάζεται ένας κύκλος που περνά ακριβώς μια φορά από κάθε πλευρά ενός γράφου χωρίς απαραίτητα να περνά ακριβώς μια φορά και από κάθε κορυφή. Ο γράφος που έχει τέτοιο κύκλο ονομάζεται **γράφος Euler** και προκύπτει εύκολα ότι οι γράφοι αυτοί έχουν άρτιο βαθμό κορυφών.

Ένας κύκλος που περνά ακριβώς μια φορά από κάθε κορυφή ενός γράφου χωρίς ωστόσο απαραίτητα να διασχίζει και όλες τις πλευρές ονομάζεται **κύκλος Hamilton** και αντίστοιχα **γράφος Hamilton**.



Εικόνα 2.7 : Κύκλος Euler



Εικόνα 2.6 : Κύκλος Hamilton

2.3.2 Αναπαράσταση Γράφων

2.3.2.1 Πίνακες Γειτνίασης (Adjacency Matrix)

Κάθε γράφος G = {V,E} με n κορυφές. Τότε ο γράφος μπορεί να παρασταθεί με τη βοήθεια ενός n × n πίνακα A(G), όπου :

$$\mathrm{A}(G)=[a_{i,j}]\,,\quad a_{i,j}=\begin{cases}1,\ \alpha\nu\ (v_i,v_j)\in E\\ 0,\ \delta\iota\alpha\varphi o\rho\varepsilon\tau\iota\kappa\acute{\alpha}\end{cases}$$

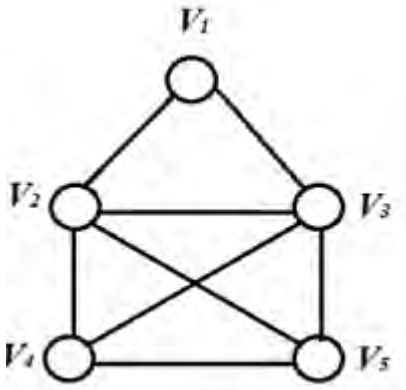
Ο πίνακας ονομάζεται πίνακας γειτνίασης και συμμετρικός. Στην περίπτωση που ο γράφος είναι με βάρη και έστω ότι το βάρος της κάθε ακμής είναι κάποια τιμή t , τότε για την αναπαράσταση του γράφου ο πίνακας γειτνίασης έχει την παρακάτω μορφή.

$$\mathrm{A}(G)=[a_{i,j}],\quad a_{i,j}=\begin{cases}\beta\acute{\alpha}\rho o\varsigma\,(i,j),\ \alpha\nu\ (v_i,v_j)\in E\\ \infty,\ \delta\iota\alpha\varphi o\rho\varepsilon\tau\iota\kappa\acute{\alpha}\end{cases}$$

2.3.2.2 Πίνακας Πρόσπτωσης

Έστω ένας γράφος G = {V,E} με n κορυφές και m ακμές. Τότε ο γράφος αυτός μπορεί να παρασταθεί με τη βοήθεια ενός n × m πίνακα B(G), που ονομάζεται πίνακας πρόσπτωσης και ισχύει

$$B(G)=[b_{i,j}],\quad b_{i,j}=\begin{cases}1,\ \alpha\nu\ e_j\ \pi\rho o\sigma\pi\acute{\iota}\pi\tau\varepsilon\iota\, v_i)\\ 0,\ \delta\iota\alpha\varphi o\rho\varepsilon\tau\iota\kappa\acute{\alpha}\end{cases}$$



	V1	V2	V3	V4	V5
V1	0	1	1	0	0
V2	1	0	1	1	1
V3	1	1	0	1	1
V4	0	1	1	0	1
V5	0	1	1	1	0

Εικόνα 2.8 : Αναπαράσταση γράφου με πίνακα γειτνίασης

:

2.3.2.3 Λίστες γειτνίασης (Adjacency Lists)

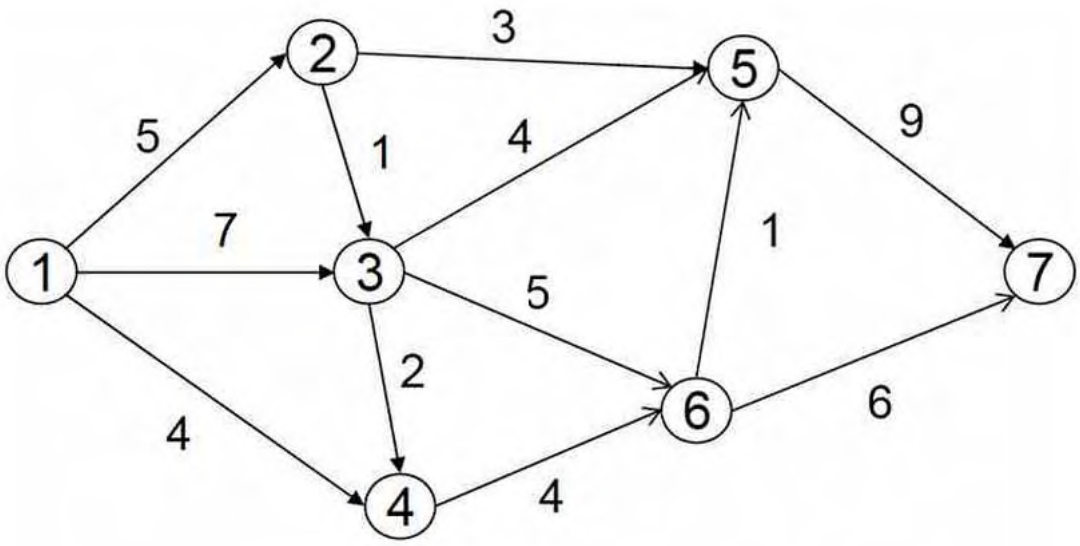
Ένας γράφος G = {V,E} είναι δυνατόν να αναπαρασταθεί ως ένας μονοδιάστατος πίνακας A. Για κάθε κορυφή v_i, A[v_i] είναι ένας δείκτης σε μια συνδεδεμένη λίστα στην οποία αποθηκεύονται οι κορυφές που γειτνιάζουν με τη v_i. Οι λίστες που απαιτούνται είναι όσες και ο αριθμός των κόμβων. Ομοίως στη περίπτωση του γράφου με βάρη στη λίστα γειτνίασης εισάγουμε το βάρος της κάθε ακμής.

2.4 **Δίκτυα**

Η έννοια του δικτύου είναι άμεσα συνδεμένη με αυτή του γράφου. Ένα δίκτυο[1,17,18] G = {V,E} μπορεί να οριστεί ως ένας κατευθυνόμενος γράφος με τις ακμές του να υποδηλώνουν κάποια σχέση μεταξύ των δυο κορυφών.

Ο όρος δίκτυο συνήθως χρησιμοποιείται για να περιγράψει μια δομή που μπορεί να είναι είτε φυσική (π.χ. δρόμοι και διασταυρώσεις, τηλεφωνικές γραμμές κλπ) είτε εννοιολογική (π.χ. γραμμές πληροφοριών, τηλεοπτικοί σταθμοί κλπ). Κάθε ένα από αυτά τα δίκτυα περιλαμβάνει δύο τύπους στοιχείων, ένα σύνολο σημείων και ένα σύνολο γραμμών που συνδέουν τα σημεία αυτά μεταξύ τους. Συνεπώς προκύπτει ο μαθηματικός ορισμός ενός δικτύου, ως ένα σύνολο κόμβων ή κορυφών και μια σειρά από συνδέσμους ή ακμές που συνδέει τους κόμβους αυτούς. Κάθε ακμή χαρακτηρίζεται από μια τιμή (βάρος) η οποία επηρεάζει τη συγκεκριμένη σύνδεση - διαδρομή. Η μονάδα μέτρησης της τιμής αυτής εξαρτάται καθαρά απο τη φύση του προβλήματος διότι μπορεί να αντιπροσωπεύει ηλεκτρική αντίσταση,χρόνο, κόστος ή οτιδήποτε άλλο σχετικό. Οι κόμβοι αντιπροσωπεύουν το σημείο σύνδεσης των ακμών και δε συνδέονται με το βάρος. Στους κόμβους των δικτύων της συγκεκριμένης εργασίας είναι δυνατόν να περάσεις ακολουθώντας ένα μονοπάτι - διαδρομή.. Όπως αναφέρθηκε και στην προηγούμενη ενότητα μονοπάτι είναι μια ακολουθία κατευθυνόμενων ακμών που οδηγεί από έναν κόμβο σε έναν άλλο. Το συνολικό βάρος - κόστος της

διαδρομής είναι το άθροισμα των επιμέρους κάθε ακμής του συγκεκριμένου μονοπατιού που ακολουθήθηκε. Αξίζει να σημειωθεί ότι ένα ζεύγος κόμβων είναι δυνατόν να συνδέεται συνήθως με περισσότερες από μια διαδρομές. Για παράδειγμα από το δίκτυο της εικόνας 2.9 η μετάβαση απο τον κόμβο 1 στον κόμβο 6 μπορεί να πραγματοποιηθεί με τις παρακάτω διαδρομές σύμφωνα με τα βάρη των ακμών : (5,1,5), (5,1,2,4), (7,5), (7,2,4), (4,4). Το ποιά διαδρομή ακριβώς θα επιλεγεί θα αναλυθεί σε επόμενο κεφάλαιο. Φυσικά είναι επόμενο ότι σε μεγάλα δίκτυα ο αριθμός των διαδρομών που μπορεί να προκύπτουν κάθε φορά μπορεί να είναι εξαιρετικά μεγάλος



Εικόνα 2.9 : Κατευθυνόμενο δίκτυο με βάρη

2.4.1 Ανάλυση και μέτρα δικτύων

Η ανάλυση ενός δικτύου αφορούν στην αξιολόγηση της δομής του απαιτεί τον ορισμός δυο μεγεθών :

- Συνδετικότητα(*Connectivity*), η οποία αναφέρεται στην πυκνότητα των συνδέσμων - ακμών σε ένα δίκτυο καθώς και την αμεσότητα των συνδέσμων.
- Προσβασιμότητα (*Accessibility*), η οποία αναφέρεται στην σχέση των επιμέρους κόμβων με το υπόλοιπο δίκτυο.

Η συνδετικότητα ουσιαστικά μετρά το βαθμό σύνδεσης μεταξύ των κόμβων και αποτελεί μια απο τις πιο σημαντικές ιδιότητες ενός δικτύου και αποτελεί βασικό χαρακτηριστικό της πολυπλοκότητας του. Η αξιολόγηση του καθορίζεται από τους παρακάτω δείκτες :

- Δείκτης Γάμμα (γ)

  Ο δείκτης αυτός είναι ο λόγος του αριθμού των ακμών του δικτύου (e) προς το μέγιστο δυνατό αριθμό ακμών e_{max} που υπάρχουν στο δίκτυο. Οι τιμές που μπορούν να προκύψουν είναι από 0 έως 1 και αναφέρονται σε ασύνδετα και πλήρως συνδεδεμένα δίκτυα.

$$\gamma = \frac{e}{e_{\max}} \qquad (2.1)$$

- Δείκτης Άλφα (α)

  Ο δείκτης άλφα συγκρίνει τον αριθμό των μονοπατιών με τον μέγιστο αριθμό όλων των δυνατών μονοπατιών

$$\alpha = \frac{u}{2v-5} \qquad (2.2)$$

  Όπου u είναι ο μέγιστος αριθμός ανεξάρτητων κύκλων σε ένα δίκτυο.

2.5 **Σύνοψη**

Η θεωρία γράφων και δικτύων αποτέλεσε τη βάση για την προσομοίωση των προβλημάτων δρομολόγησης. Για το λόγο αυτό στο κεφάλαιο αυτό παρουσιάστηκαν τα βασικά χαρακτηριστικά ενός γράφου. Δόθηκαν οι κυριότεροι ορισμοί, αναφέρθηκαν τα βασικότερα χαρακτηριστικά που διέπουν ένα γράφο και παρουσιάστηκαν οι τρόποι αναπαράστασης τους. Τέλος έγινε μια σύντομη αναφορά στα δίκτυα. Στο κεφάλαιο που ακολουθεί θα γίνει αναφορά στα κυριότερα προβλήματα δρομολόγησης που έχουν απασχολήσει και συνεχίζουν να απασχολούν τη διεθνή επιστημονική κοινότητα.

ΚΕΦΑΛΑΙΟ ΤΡΙΤΟ:
Δρομολόγηση

3.1 Εισαγωγή

Στο κεφάλαιο αυτό γίνεται μια εισαγωγή στην έννοια της δρομολόγησης. Στη πρώτη ενότητα δίνεται ένας ορισμός και ακολουθεί μια σύντομη αναφορά στα κυριότερα προβλήματα δρομολόγησης και σε ορισμένους βασικούς αλγόριθμους επίλυσής τους. Σκοπός του κεφαλαίου είναι να πραγματοποιηθεί μια επισκόπηση στα προβλήματα αυτά καθότι συνδέονται άμεσα με το πρόβλημα δρομολόγησης που πραγματεύεται η παρούσα διπλωματική εργασία.

3.2 Η έννοια της Δρομολόγησης (*Routing*)

Η έννοια της Δρομολόγησης σχετίζεται με την κατάστρωση και σχεδίαση της βέλτιστης διαδρομής πάνω σε ένα δίκτυο, του οποίου οι ακμές (δυνατές μεταβάσεις μεταξύ των κόμβων) χαρακτηρίζονται από κόστη. Τα κόστη αυτά μπορεί να είναι χρόνος μετακίνησης, αποστάσεις, οικονομική επιβάρυνση ή και συνδυασμοί των προηγουμένων, ανάλογα με τη φύση του εξεταζόμενου προβλήματος. Η δρομολόγηση γίνεται για διάφορα είδη δικτύων, συμπεριλαμβανομένου του τηλεφωνικού δικτύου (μεταγωγή κυκλώματος), ηλεκτρονικά δίκτυα δεδομένων (διαδίκτυο) και τα δίκτυα μεταφορών. Ο αντικειμενικός σκοπός του προβλήματος είναι ο εντοπισμός της βέλτιστης διαδρομής. Το πρόβλημα της δρομολόγησης αποτελεί ένα από τα πιο πολυσυζητημένα προβλήματα της επιστημονικής περιοχής της βελτιστοποίησης. Οι αλγόριθμοι και οι τεχνικές που χρησιμοποιούνται για την εύρεση ή προσέγγιση της βέλτιστης διαδρομής, απαντώνται τακτικά σε προβλήματα της συνδυαστικής βελτιστοποίησης *(combinatorial optimization*), με προεκτάσεις στις συγγενείς περιοχές της στοχαστικής βελτιστοποίησης (*stochastic optimization*) και βέλτιστου ελέγχου (*optimal control*).

Ένα πρόβλημα που μπορεί να τεθεί σε ένα δίκτυο είναι το ακόλουθο : Δοσμένου ενός αρχικού κόμβου, να οδηγηθούμε σε έναν άλλο κόμβο, επιλέγοντας εκείνο το μονοπάτι που φέρει το ελάχιστο αθροιστικό κόστος επί των ακμών του δικτύου, που οδηγούν διαδοχικά από τον αρχικό κόμβο στον κόμβο προορισμού. Η εύρεση αυτού του βέλτιστου μονοπατιού ορίζει ένα πρόβλημα δρομολόγησης και η ιδέα του προβλήματος μπορεί να αποτελεί την απλοποιημένη εκδοχή για μια ευρύτερη κλάση προβλημάτων δρομολόγησης. Το πρόβλημα αυτό είναι γνωστό σαν το Πρόβλημα του Συντομότερου Μονοπατιού (*Shortest Path Problem*). Αποτελεί σημείο αναφοράς για σχεδόν όλα τα προβλήματα δρομολόγησης και απαντάται σε πολλές περιοχές έρευνας όπως για παράδειγμα Η Θεωρητική Πληροφορική και Επιστήμη των Υπολογιστών, η Επιχειρησιακή Έρευνα, τα Διακριτά Μαθηματικά κ.α.

3.3 **Προβλήματα Δρομολόγησης**

Η εύρεση της βέλτιστης διαδρομής εξαρτάται κάθε φορά απτό είδος του προβλήματος, αν δηλαδή η διαδρομή που επιζητούμε είναι μονοπάτι ή κύκλος, αν τα κόστη είναι σταθερά ή μεταβαλλόμενα κ.α. Συνεπώς προκύπτει μια κατηγοριοποίηση των διαφόρων προβλημάτων δρομολόγησης, οπότε και εξετάζονται το καθένα διαφορετικά, τα οποία και αναφέρονται εν συντομία παρακάτω.

3.3.1 **Το Πρόβλημα του Πλανόδιου Πωλητή (TSP)**

Ένα από τα πιο διαδεδομένα προβλήματα δρομολόγησης είναι το πρόβλημα του πλανόδιου πωλητή[4,8] το οποίο ως εξής :

Έστω ένας πλανόδιος πωλητής ο οποίος απαιτείται να περάσει από ένα συγκεκριμένο αριθμό πόλεων. Ο πωλητής ξεκινάει από μια συγκεκριμένη πόλη (αφετηρία), με την απαίτηση να μεταβεί σε κάθε πόλη ακριβώς μια φορά και να επιστρέψει και πάλι στην αφετηρία του έχοντας πραγματοποιήσει την ελάχιστη διαδρομή. Όσον αφορά τον όρο ελάχιστη διαδρομή, εξαρτάται κάθε φορά απ το αντικείμενο προς βελτιστοποίηση και τη φύση του προβλήματος και μπορεί να είναι κόστος διαδρομής, η αποφυγή μιας δύσκολης διαδρομής, ο χρόνος μετάβασης και άλλα.

Το πρόβλημα αυτό διατυπώθηκε για πρώτη φορά το 1930 και έκτοτε αποτέλεσε βασικό αντικείμενο μελέτης για πολλά ακόμα προβλήματα βελτιστοποίησης. Στις επόμενες δεκαετίες 1950 και 60 το πρόβλημα έγινε αρκετά δημοφιλές στην επιστημονική κοινότητα καθώς διατυπώθηκε ως πρόβλημα γραμμικού προγραμματισμού. Αν και η διατύπωση του προβλήματος μοιάζει αρκετά

Εικόνα 3.1 : Εφαρμογή TSP στις 15 μεγαλύτερες πόλεις της Γερμανίας

απλή, ωστόσο το TSP αποτελεί ένα απ τα προβλήματα όπου δεν έχει βρεθεί κάποια γενική μέθοδος επίλυσης. Λύση στο TSP είναι δυνατόν να δώσουν ένας μεγάλος αριθμός ευρετικών αλγορίθμων ακόμα και σε περιπτώσεις που υπάρχει μεγάλος όγκος δεδομένων. Αξίζει να αναφερθεί ότι για πρώτη φορά μοντελοποιήθηκε και επιλύθηκε ένα πραγματικό πρόβλημα με τις 49 πρωτεύουσες πολιτειών των Η.Π.Α.

Το TSP περιγράφεται από έναν γράφο με βάρη, οι κορυφές του οποίου αναπαριστούν τις πόλεις και οι ακμές τις δυνατές μεταβάσεις. Τα βάρη επί των ακμών περιγράφουν τις αποστάσεις μεταξύ των πόλεων. Ουσιαστικά η διαδρομή σ ένα τέτοιο πρόβλημα συνιστά έναν κύκλο Hamilton ελάχιστου μήκους. Επιπλέον ο γράφος μπορεί να είναι είτε κατευθυνόμενος είτε όχι, με το πρόβλημα να είναι ακόμα πιο σύνθετο στη πρώτη περίπτωση. Όσον αφορά τη δυσκολία επίλυσης κατατάσσεται στην NP, με αυτό να σημαίνει ότι δεν υπάρχει κάποιος αλγόριθμος που να το επιλύει σε πολυωνυμικό χρόνο, δηλαδή σε χρόνο που εξαρτάται πολυωνυμικά από το πλήθος των πόλεων που υπάρχουν ως δεδομένα στο πρόβλημα. Συνεπώς στις περιπτώσεις που αυξάνονται τα δεδομένα είναι επόμενο ο χρόνος επίλυσης να αυξάνεται εκθετικά.

3.3.2 Το Πρόβλημα Δρομολόγησης Οχημάτων (VRP)

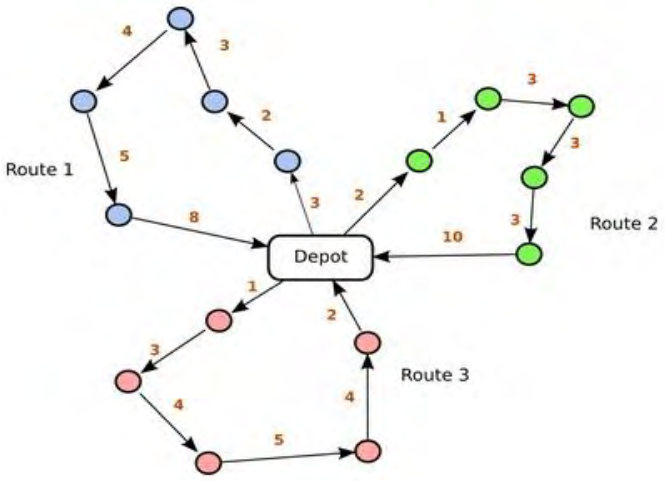
Το Πρόβλημα Δρομολόγησης Οχημάτων[13,14] (*Vehicle Routing Problem*) εντάσσεται στη κατηγορία των προβλημάτων Ακέραιου Προγραμματισμού (*Linear Programming*), στο οποία οι μεταβλητές λαμβάνουν επιτρεπτές τιμές από ένα σύνολο ακεραίων αριθμών.

Το πρόβλημα μπορεί να διατυπωθεί ως εξής : Απαιτείται να καθοριστεί το σύνολο των βέλτιστων διαδρομών για έναν στόλο οχημάτων, προκειμένου να εξυπηρετηθούν ορισμένοι πελάτες που βρίσκονται σε συγκεκριμένους κόμβους ενός δικτύου. Τα οχήματα αυτά είναι δυνατόν να αναχωρούν από μία ή και περισσότερες βάσεις - αποθήκες ανεφοδιασμού στις οποίες και πρέπει να επιστρέψουν.

Η έρευνα στη δρομολόγηση οχημάτων ξεκίνησε το 1959 με το πρόβλημα αποστολής φορτηγών από τους *Dantzig* και *Ramser*. Η διατύπωση του ήταν η εξής, έστω ένας στόλος φορτηγών μεταφοράς καυσίμων, ζητείται η βέλτιστη διαδρομή μεταξύ του σταθμού ανεφοδιασμού και των σταθμών εξυπηρέτησης. Η προσεγγιστικά βέλτιστη λύση προέκυψε μετά από μοντελοποίηση του προβλήματος με γραμμικό προγραμματισμό. Έκτοτε έχουν προταθεί αρκετοί αλγόριθμοι και μοντέλα για την ακριβή επίλυση διάφορων περιπτώσεων VRP. Σήμερα υπάρχουν αρκετά πακέτα λογισμικού τα οποία δίνουν λύση σε πραγματικά προβλήματα διαχείρισης στόλου οχημάτων.

Η βασική ιδέα είναι η διανομή αγαθών ανάμεσα σε ένα σύνολο αποθηκών και ένα σύνολο γεωγραφικά διασκορπισμένων πελατών, κάνοντας τη θεώρηση ότι οι θέσεις των πελατών αποτελούν τους κόμβους εξυπηρέτησης πάνω σ ένα δίκτυο στο οποίο σημειώνουμε και όλες τις δυνατές μεταβάσεις - ακμές που ενώνουν τους κόμβους αυτούς αλλά και με τους σταθμούς ανεφοδιασμού.

Το VRP αποτελεί μια επέκταση του TSP διότι στο πρώτο επιζητούμε να m το πλήθος βέλτιστες διαδρομές οχημάτων, στις οποίες κάθε όχημα εκκινεί από έναν κόμβο - αποθήκη, διατρέχει ένα υποσύνολο κόμβων πελατών σε καθορισμένη σειρά και επιστρέφει στον αρχικό κόμβο. Τα οχήματα είναι δυνατόν να έχουν διαφορετικές χωρητικότητες και πρέπει να επισκεφτούν τα σημεία εξυπηρέτησης ακριβώς μια φορά και όπως είναι επόμενο η συνολική ζήτηση φορτίου πάνω σε μια διαδρομή δε πρέπει να ξεπερνά τη χωρητικότητα του οχήματος. Ο αντικειμενικός σκοπός είναι η ελαχιστοποίηση του κόστους για κάθε μια απ τις m αυτές διαδρομές και επομένως και το συνολικό κόστος των παραδόσεων. Φυσικά σε ένα πραγματικό πρόβλημα μπορούν να προκύψουν πολλοί περιορισμοί οι οποίοι να μετατρέψουν το πρόβλημα σε ακόμα πιο πολύπλοκο όπως ορισμό του συνολικού χρόνου δρομολόγησης, να υπάρχουν συγκεκριμένα χρονοπαράθυρα μέσα στα οποία μπορεί να ξεκινήσει η εξυπηρέτηση κ.α.. Το πρόβλημα αυτό κατατάσσεται στην πολυπλοκότητα τύπου NP, δηλαδή όπως και στο πρόβλημα TSP δεν έχει βρεθεί αλγόριθμος που να το επιλύει σε πολυωνυμικό χρόνο.



Εικόνα 3.2 :Αναπαράσταση Προβλήματος Δρομολόγησης Οχημάτων

3.3.3 Το Πρόβλημα της Συντομότερης Διαδρομής (SPP)

Ένα από τα πιο βασικά προβλήματα στη θεωρία των δικτύων είναι το πρόβλημα της συντομότερης διαδρομής[1] (*Shortest Path Problem*). Το πρόβλημα αφορά την εύρεση της συντομότερης διαδρομής μεταξύ μιας προέλευσης (αρχικός κόμβος) και ενός προορισμού (τελικός κόμβος) διαμέσου ενός συνεκτικού δικτύου, όταν είναι γνωστές οι αποστάσεις των αντίστοιχων κλάδων του δικτύου. Ο ορισμός του προβλήματος καθορίζει ότι σε έναν κατευθυνόμενο γράφο με βάρη στις ακμές G = {V,E} και με n αριθμό κόμβων. Το μήκος του μονοπατιού είναι το μήκος όλων των ακμών του. Συντομότερο μονοπάτι χαρακτηρίζεται αυτό το οποίο έχει ίδιο κόμβο αφετηρίας και προορισμού αλλά με μικρότερο μήκος απο όλα τα υπόλοιπα. Αν και έχουν προταθεί διάφορες μέθοδοι επίλυσης (αλγόριθμοι), ο αλγόριθμος του *Dijkstra* που παρουσιάζεται παρακάτω είναι από

τις πιο γρήγορες και εύχρηστες μεθόδους. Η διαδικασία αυτή εξελίσσεται από την αρχή του δικτύου, προσδιορίζοντας μία αλληλουχία των κόμβων του δικτύου σε ανερχόμενη σειρά της συντομότερης τους απόστασης από την αρχή, λύνοντας έτσι το πρόβλημα όταν φτάσει στον τελικό κόμβο. Καθότι το πρόβλημα της εργασίας αυτής μπορεί να χαρακτηριστεί ώς πρόβλημα συντομότερης διαδρομής, παρακάτω θα γίνει μια σύντομη αναφορά στον βασικό αλγόριθμο επίλυσης του SPP.

3.3.3.1 Αλγόριθμος Dijkstra

Ο αλγόριθμος αυτός επινοήθηκε απο τον Ολλανδό επιστήμονα *Edsger Dijkstra* το 1959 και επιλύει το SPP σε έναν γράφο με μη αρνητικά βάρη. Παράγει το συντομότερο δέντρο όλων των μονοπατιών που οδηγούν απο τον κόμβο αφετηρίας στον κόμβο προορισμού και χρησιμοποιείται τακτικά σε προβλήματα δρομολόγησης. Όπως για παράδειγμα όταν οι κόμβοι ενός γράφου αναπαριστούν πόλεις και τα κόστη των δυνατών μεταβάσεων χιλιομετρικές αποστάσεις, ο αλγόριθμος αυτός μπορεί να χρησιμοποιηθεί για να βρεθεί η συντομότερη διαδρομή δοσμένης αφετηρίας και προορισμού.

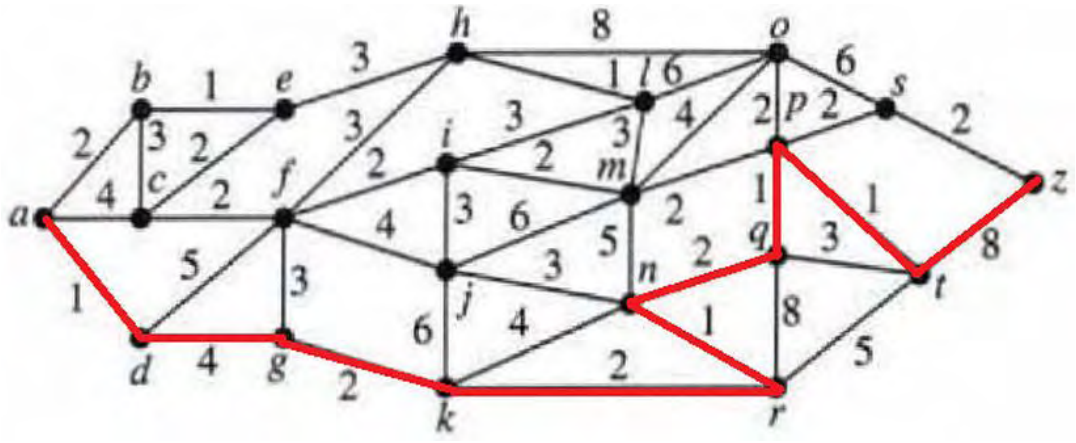
Συγκεκριμένα χωρίζει τους κόμβους του δικτύου σε λυμένους και άλυτους. Λυμένοι είναι οι κόμβοι των οποίων η ελάχιστη απόσταση από τον κόμβο προέλευσης έχει υπολογιστεί και άλυτοι αυτοί των οποίων δεν έχει. Στην αρχή, όλοι οι κόμβοι εκτός από τον κόμβο προέλευσης (του οποίου η ελάχιστη απόσταση είναι 0) είναι άλυτοι. Στη συνέχεια ο αλγόριθμος λειτουργεί επαναληπτικά, υπολογίζοντας σε κάθε επανάληψη τον επόμενο πλησιέστερο κόμβο στον αρχικό.

- *Σκοπός της n-οστής επανάληψης:* Βρες τον n-οστό πλησιέστερο κόμβο στον αρχικό. (Επανάλαβε για n = 1,2,... μέχρι ο n-οστός πλησιέστερος κόμβος να είναι ο κόμβος προορισμού.
- *Δεδομένο για τη n-οστή επανάληψη:* Είναι οι (n-1) πλησιέστεροι κόμβοι στον αρχικό (που λύθηκαν σε προηγούμενες επαναλήψεις), μαζί με τις συντομότερες διαδρομές τους και αποστάσεις από τον κόμβο προέλευσης. (Οι κόμβοι αυτοί καθώς και ο αρχικός είναι οι λυμένοι όπως είπαμε ενώ οι υπόλοιποι είναι οι άλυτοι).
- *Υποψήφιοι για n-οστοί πλησιέστεροι στον αρχικό:* Κάθε άλυτος κόμβος που είναι απευθείας συνδεδεμένος με έναν λυμένο κόμβο είναι υποψήφιος να είναι ο n-οστός πλησιέστερος κόμβος στην n-οστή επανάληψη. Η λυμένη κορυφή που είναι άμεσα συνδεμένη με ακμή με μια ή περισσότερες μη λυμένες κορυφές δίνει τη μη λυμένη κορυφή με τη συντομότερη συνδεμένη ακμή. (Οι ισοβαθμήσεις δίνουν επιπρόσθετους υποψήφιους.)
- *Υπολογισμός του n-οστού πλησιέστερου κόμβου:* Για κάθε λυμένο κόμβο που συνδέεται απευθείας με έναν ή περισσότερους άλυτους κόμβους υπολόγισε για κάθε ένα από αυτούς

τους άλυτους κόμβους το άθροισμα της ελάχιστης απόστασης του λυμένου κόμβου και της απόστασης μεταξύ των δύο αυτών κόμβων (λυμένου και άλυτου). Ο υποψήφιος κόμβος με τη μικρότερη συνολική απόσταση είναι ο n-οστός πλησιέστερος κόμβος (ισοβαθμήσεις δίνουν επιπρόσθετους λυμένους κόμβους), και η συντομότερη διαδρομή της είναι εκείνη που δίνει την απόσταση αυτή.

- Το πρόβλημα μέχρι τώρα έχει αντιμετωπιστεί ως το πρόβλημα ελαχιστοποίησης της απόστασης μεταξύ ενός κόμβου προέλευσης και ενός κόμβου προορισμού ενός δικτύου. Όμως αυτό δε σημαίνει ότι οι τιμές των κλάδων πρέπει οπωσδήποτε να είναι αποστάσεις. Για παράδειγμα, οι κλάδοι μπορεί να αντιστοιχούν σε δραστηριότητες κάποιου είδους, όπου η τιμή για κάθε κλάδο είναι το κόστος της δραστηριότητας. Σε αυτή την περίπτωση, το πρόβλημα είναι να βρεθεί η ακολουθία των δραστηριοτήτων που ελαχιστοποιεί το συνολικό κόστος. Ακόμη, η τιμή ενός κλάδου μπορεί να είναι ο χρόνος που χρειάζεται για την εκτέλεση της δραστηριότητας. Στην περίπτωση αυτή, το πρόβλημα είναι να βρεθεί η ακολουθία των δραστηριοτήτων που ελαχιστοποιεί το συνολικό χρόνο.

Η ανάλυση της πολυπλοκότητας του αλγόριθμου του Dijkstra μπορεί να σκιαγραφηθεί ως εξής. Σε κάθε επανάληψη, ο αλγόριθμος εκτελεί μία σειρά ενεργειών που είναι ανάλογη του αριθμού των άλυτων κόμβων, ο οποίος σε ένα δίκτυο με n κόμβους είναι αρχικά το πολύ n και μειώνεται σε κάθε επανάληψη. Επομένως, η πολυπλοκότητα του αλγόριθμου είναι O(n + n-1 + … + 1) = O(n2).



Εικόνα 3.3 : Συντομότερη Διαδρομή (Shortest Path)

3.4 **Σύνοψη**

Τα προβλήματα δρομολόγησης αποτελούν αντικείμενο συνεχούς έρευνας και μελέτης. Είναι απο τα πλέον ενδιαφέροντα προβλήματα βελτιστοποίησης ροής και βρίσκουν εφαρμογή οπουδήποτε εμπλέκονται δίκτυα μεταφορών όπως δίκτυα διανομής αγαθών και δίκτυα μεταφοράς προσωπικού. Το TSP, VRP και SPP αποτελούν τα κυριότερα απο αυτά με ένα μεγάλο πλήθος αλγορίθμων και μαθηματικών μοντέλων να έχουν αναπτυχθεί για την εύρεση της βέλτιστης λύσης. Ένα παρεμφερές πρόβλημα που αντιμετωπίζουμε στην καθημερινότητα μας αποτελεί η μετακίνηση με πολλαπλά μέσα μεταφοράς το οποίο αποτελεί και το πεδίο έρευνας της παρούσας διπλωματικής.

Στο επόμενο κεφάλαιο θα παρουσιαστεί το πρόβλημα δρομολόγησης πολλαπλών μέσων μεταξύ πόλεων που εξετάζει η παρούσα διπλωματική και θα αναλυθεί η μοντελοποίηση του.

Κεφάλαιο Τέταρτο:
Συντομότερη Διαδρομή σε Δίκτυο πόλεων της Ελλάδας με Μοντέλο Μικτού Ακέραιου Προγραμματισμού

4.1 Εισαγωγή

Στο προηγούμενο κεφάλαιο δόθηκε ένας ορισμός για την έννοια της δρομολόγησης και παρουσιάστηκαν εν συντομία τα κυριότερα προβλήματα δρομολόγησης μαζί με το SPP που αποτελεί και το πρόβλημα της παρούσας εργασίας.

Στο κεφάλαιο αυτό θα γίνει παρουσίαση των δύο διαφορετικών μαθηματικών μοντέλων μικτού ακέραιου γραμμικού προγραμματισμού που καταστρώθηκαν προκειμένου να λυθεί σε ένα δίκτυο πόλεων της Ελληνικής επικράτειας το πρόβλημα της συντομότερης διαδρομής με γνώμονα το συνολικό κόστος μετακίνησης.

4.2 Γενικά περί Γραμμικού Προγραμματισμού

Γραμμικός προγραμματισμός[5] (Linear programming –LP) ή γραμμική βελτιστοποίηση (Linear optimization) είναι μία μαθηματική μέθοδος μέσω της οποίας προσδιορίζεται ένας τρόπος ώστε να επιτευχθεί το βέλτιστο αποτέλεσμα σε ένα μαθηματικό μοντέλο, δεδομένης μιας σειράς περιορισμών που εκφράζονται ως γραμμικές συναρτήσεις. Το βέλτιστο αυτό αποτέλεσμα είναι συνήθως η μεγιστοποίηση του κέρδους ή η ελαχιστοποίηση του κόστους. Ο γραμμικός προγραμματισμός υπάγεται στη γενικότερη κατηγορία του μαθηματικού προγραμματισμού όπως αναφέρθηκε παραπάνω.

Η βελτιστοποίηση της γραμμικής συνάρτησης που ονομάζεται αντικειμενική αποτελεί τον κύριο σκοπό. Οι περιορισμοί έχουν τη μορφή ανισο/ισοτήτων και οι μεταβλητές είναι πραγματικοί θετικοί αριθμοί που ονομάζονται μεταβλητές απόφασης. Οι λύσεις εμφανίζονται στη εφικτή περιοχή η οποία καθορίζεται από τους περιορισμούς. Η βέλτιστη λύση εντοπίζεται εντός της εφικτής περιοχής και είναι η τιμή που μεγιστοποιεί ή ελαχιστοποιεί την αντικειμενική συνάρτηση.

4.3 Διατύπωση του Προβλήματος

Η παρούσα διπλωματική πραγματεύεται ένα πρόβλημα με το οποίο έχει έρθει αντιμέτωπος σχεδόν οποιοσδήποτε χρειάστηκε να μετακινηθεί εντός της Ελλάδας από κάποια πόλη σε κάποια άλλη είτε για επαγγελματικούς είτε για οποιοδήποτε άλλο σκοπό. Δηλαδή με ποιό μέσο θα επιλέξει να μετακινηθεί έτσι ώστε να πληρώσει το ελάχιστο συνολικό κόστος και σε περίπτωση που απαιτηθεί κάποια μετεπιβίβαση, ποια δρομολόγια πρέπει να επιλέξει ώστε να αφιχθεί στον χρόνο που επιθυμεί.

Συνεπώς αντικείμενο προς βελτιστοποίηση είναι η ελαχιστοποίηση του συνολικού κόστους δοθείσας της πόλης εκκίνησης - αφετηρίας (*Origin*) και προορισμού (*Destination*). Ο ταξιδιώτης – χρήστης επιλέγει το μέρος αναχώρησης και προορισμού καθώς και ένα σημείο ενδιαφέροντος(πόλη) που επιθυμεί να επισκεφτεί κατά τη διάρκεια του ταξιδιού του. Για τις μετακινήσεις του από πόλη σε πόλη έχει τη δυνατότητα να χρησιμοποιήσει κάποιο από τα τρία διαφορετικά μέσα μεταφοράς, τρένο – λεωφορείο (ΚΤΕΛ) - Ι.Χ. - πλοίο ή ακόμα και συνδυασμό αυτών, με διαδοχικές μετεπιβιβάσεις με σκοπό την ελαχιστοποίηση του κόστους ταξιδίου. Επιπλέον επιλέγει την ώρα αναχώρησης από την αφετηρία, την ώρα άφιξης στον τελικό του προορισμό καθώς και τον χρόνο παραμονής στη πόλη – σημείο ενδιαφέροντος. Δεδομένου όμως ότι τα μέσα μαζικής μεταφοράς (Τρένο-Κτελ-Πλοίο) έχουν συγκεκριμένα δρομολόγια αναχώρησης και επειδή η λύση που προκύπτει επιθυμούμε να είναι όσο το δυνατόν πιο κοντά στη πραγματικότητα, αυτά συγκεντρώθηκαν από τις βάσεις δεδομένων των υπεραστικών λεωφορείων, των σιδηροδρόμων και των πλοιοκτητριών εταιριών.

Όσον αφορά για το καθένα από τα μέσα μεταφοράς, το Ι.Χ. προσφέρει άμεση και ταχύτερη μετάβαση από πόλη σε πόλη ωστόσο στην μετάβαση με τρένο ή ακόμα και με λεωφορείο μπορεί να απαιτηθεί η μετεπιβίβαση σε κάποια πόλη διότι δεν υπάρχει απευθείας σύνδεση μεταξύ όλως των πόλεων. Τέλος υπάρχει και η περίπτωση μετακίνησης του ταξιδιώτη μέσα στην ίδια πόλη ώστε να μεταβεί από ένα terminal σε κάποιο άλλο για να αλλάξει μέσο μεταφοράς.

Τα δεδομένα που χρησιμοποιήθηκαν αφορούν τις δεκαπέντε (15) μεγαλύτερες πόλεις της Ελλάδας καθώς και αυτές που αποτελούν κόμβους μετεπιβίβασης για άλλα μέσα μεταφοράς, όπως το τρένο. Οι πόλεις αυτές είναι οι παρακάτω :

- ✓ Αθήνα
- ✓ Λάρισα
- ✓ Βόλος
- ✓ Θεσσαλονίκη
- ✓ Φάρσαλα
- ✓ Καρδίτσα
- ✓ Ιωάννινα
- ✓ Αλεξανδρούπολη
- ✓ Κατερίνη
- ✓ Τρίκαλα
- ✓ Πάτρα
- ✓ Ηγουμενίτσα
- ✓ Καβάλα
- ✓ Ξάνθη
- ✓ Ηράκλειο

4.4 Μαθηματικά Μοντέλα

Το συγκεκριμένο πρόβλημα αποτελεί ένα πρόβλημα δρομολόγησης πολλαπλών μέσων στο οποίο καλούμαστε να υπολογίσουμε την συντομότερη διαδρομή(*Shortest Path*). Επομένως θεωρήθηκε ένας κατευθυνόμενος γράφος με βάρη, με τους κόμβους να αποτελούν τις πόλεις της Ελλάδας, τις ακμές τις δυνατές μεταβάσεις και τα βάρη να αποτελούν το κόστος μετακίνησης. Προσεγγίστηκε με δυο διαφορετικούς τρόπους (M1,M2), όσον αφορά τη συσχέτιση μεταξύ των πόλεων και των μέσων μεταφοράς και για κάθε τέτοια προσέγγιση σχεδιάστηκαν δυο διαφορετικά μοντέλα (M1.1,M2.1) ανάλογα με τις μεταβλητές απόφασης, με την αντικειμενική συναρτηση όμως να παραμένει ίδια και στα τέσσερα.

Η μοντελοποίηση του προβλήματος έγινε με πρόγραμμα μικτού ακέραιου - γραμμικού προγραμματισμού με την αντικειμενική συνάρτηση και τους περιορισμούς να αποτελούν γραμμικές εξισώσεις όπως θα αναλύθει και παρακάτω.

4.4.1 Μοντέλο M1 με Διαφοροποίηση των Τερματικών Σταθμών (*Terminal)*

Στο συγκεκριμένο μοντέλο[2,3,5,6,7,14,15,16] θεωρούμε ότι κάθε πόλη αποτελείται απο τρείς (3) διαφορετικούς κόμβους, που αποτελούν τα *terminal* για κάθε μέσο, δηλαδή ο σταθμός των τρένων ,ο υπεραστικός σταθμός λεωφορείων και η κατοικία του ταξιδιώτη απο όπου μπορεί να ξεκινήσει με το όχημα του, με το πλήθος των κόμβων να καθορίζουν το μέγεθος του δικτύο N. Εξαίρεση αποτελέι η πόλη της Αθήνας όπου θεωρούμε ότι έχει ένα επιπλέον τερματικό σταθμό, το λιμάνι, καθώς και το Ηράκλειο που έχει μόνο έναν τερματικό σταθμό και είναι ομοίως το λιμάνι. Επομένως κάθε πόλη χαρακτηρίζεται απο τρείς (3) αριθμούς, για παράδειγμα η Αθήνα θα χαρακτηρίζεται απο τους αριθμούς 0,1,2. Ο πρώτος αριθμός 0 δηλώνει το terminal των τρένων, το 1 αντίστοιχα των

	Terminals			
	ΤΡΕΝΟ	ΚΤΕΛ	Ι.Χ.	ΠΛΟΙΟ
ΑΘΗΝΑ	0	1	2	42
ΛΑΡΙΣΑ	3	4	5	-
ΒΟΛΟΣ	6	7	8	-
ΘΕΣ/ΝΙΚΗ	9	10	11	-
ΦΑΡΣΑΛΑ	12	13	14	-
ΚΑΡΔΙΤΣΑ	15	16	17	-
ΙΩΑΝΝΙΝΑ	18	19	20	-
ΑΛΕΞ/ΠΟΛΗ	21	22	23	-
ΚΑΤΕΡΙΝΗ	24	25	26	-
ΤΡΙΚΑΛΑ	27	28	29	-
ΠΑΤΡΑ	30	31	32	-
ΗΓΟΥΜΕΝΙΤΣΑ	33	34	35	-
ΚΑΒΑΛΑ	36	37	38	-
ΞΑΝΘΗ	39	40	41	-
ΗΡΑΚΛΕΙΟ	-	-	-	43

Πίνακας 4.1: Κωδικοποίηση αριθμού πόλεων μοντέλου M1

λεωφορείων και το 3 του οχήματος, όπως ακριβώς παρουσιάζεται στον πίνακα 4.1 . Άν δηλαδη επιθυμούμε να μετακινηθούμε απο την Αθήνα στη Λάρισα με τρένο, ουσιαστικά μετακινούμαστε από τον κόμβο 0 στον κόμβο 3. Επιπλέον είναι δυνατόν να πραγματοποιηθούν μετακινήσεις και εσωτερικά των πόλεων από τον σταθμό τρένων της Αθήνας στον σταθμό λεωφορείων, δηλαδή απο τον κόμβο 0 στον κόμβο 1.

Πιο συγκεκριμένα για την καλύτερη ανάλυση του μαθηματικού μοντέλου θα αναφερθούμε στα επιμέρους χαρακτηριστικά στοιχεία του, δηλαδή στις παραμέτρους, στις μεταβλητές απόφασης, στην αντικειμενική συνάρτηση και τέλος στους περιορισμούς.

- *Δείκτες :*

i, j, k : Οι δείκτες όλων των κόμβων i, j, k = 0, 1, ..., N
n, z : Οι δείκτες των δρομολογίων n, z = 0, 1, 2, 3, 4

- *Σύνολα :*

N : Το πλήθος των κόμβων
Z : Το πλήθος των δρομολογίων

- *Παράμετροι :*

O : Ο κόμβος εκκίνησης (Ορίζεται απο τον χρήστη)
D : Ο κόμβος άφιξης (Ορίζεται απο τον χρήστη)
P : Ο κόμβος - σημείο ενδιαφέροντος που επιθυμεί να σταματήσει ο ταξιδιώτης (Ορίζεται απο τον χρήστη)
WTP : Χρονικό διάστημα παραμονής στο P (Ορίζεται απο τον χρήστη)
ToA : Επιθυμητή ώρα άφιξης στον προορισμό D (Ορίζεται απο τον χρήστη)
C_{ij} : Κόστος μετακίνησης από τον κόμβο i στον κόμβο j
t_{ij} : Χρόνος μετακίνησης απο τον κόμβο i στον κόμβο j
d_{ijn} : Ώρα αναχώρησης απο τον κόμβο i προς τον κόμβο j με το δρομολόγιο n
M : Μεγάλος αριθμός

Στις πόλεις που δεν υπάρχει απευθείας σύνδεση με κάποιο συγκεκριμένο μέσο το κόστος και ο χρόνος μετακίνησης ορίζεται με έναν μεγάλο αριθμό (200), όπως φαίνεται στους πίνακες 0.1 και 0.2, ενώ υπάρχει συγκεκριμένο κόστος και χρόνος μεταφοράς από *terminal* σε *terminal* στην ίδια πόλη. Επίσης ο πίνακας d_{ijn} στους κόμβους που δεν υπάρχει σύνδεση λαμβάνει την τιμή 24 καθότι είναι η μεγαλύτερη τιμή δρομολογίου απο όλα τα υπόλοιπα και δεν θα είναι δυνατόν να επιλεγεί.

- Μεταβλητές Απόφασης :

X_{ij} : Λαμβάνει την τιμή 1 εάν επιλέγεται η διαδρομή i,j και 0 διαφορετικά

Y_{in} : Λαμβάνει την τιμή 1 εάν ο ταξιδιώτης αναχωρεί απο τον κόμβο i με το δρομολόγιο n και 0 διαφορετικά

Z_{jn} : Λαμβάνει την τιμή 1 εάν ο ταξιδιώτης αφικνείται στον κόμβο j με το δρομολόγιο n και 0 διαφορετικά

DT_{ijn} : Ώρα αναχώρησης απο τον κόμβο i στον κόμβο j με το δρομολόγιο n

AT_{jin} : Ώρα άξιφης στον κόμβο j απο τον κόμβο i με το δρομολόγιο n

U_i : Σειρά με την οποία ο κόμβος i έχει επισκεφτεί

Οι πρώτες 3 μτεβλητές απόφασης είναι δυαδικές δηλαδή λαμβάνουν μόνο τις τιμές 0-1 ενώ η DT_{ijn} και η AT_{jin} αφούν χρόνο και είναι δυνατόν να λάβουν οποιαδήποτε τιμή. Η αντικειμενική συνάρτηση είναι η ελαχιστοποίηση του κόστους της συνολικής διαδρομής και παραμένει ίδια για όλα τα μοντέλα, όπως φαίνεται στην εξίσωση 4.1 .

$$Min\sum_{i=0}^{N}\sum_{j=0}^{N} X_{ij}C_{ij} \qquad i \neq j \tag{4.1}$$

Οι περιορισμοί του συγκεκριμένου μοντέλου είναι οι ακόλουθοι :

$$\sum_{j=0}^{N} X_{ij} \leq 1 \quad , \forall i \neq j \tag{4.3}$$

$$\sum_{j=0}^{N} X_{Oj} = 1 \quad , \; j \neq O \tag{4.4}$$

$$\sum_{i=0}^{N} X_{Di} = 0 \quad , i \neq D \tag{4.2}$$

$$\sum_{\substack{i=0 \\ i\neq j}}^{N} X_{ij} - \sum_{\substack{k=0 \\ k\neq j}}^{N} X_{jk} = 0 \quad \forall j \in N \tag{4.5}$$

$$\sum_{j=0}^{N} X_{jO} = 0 \quad , j \neq O \tag{4.6}$$

$$\sum_{i=0}^{N} X_{iD} = 1 \quad , i \neq D \tag{4.7}$$

$$X_{ij} + X_{ji} \leq 1 \quad \forall i, j, \; i \neq j \tag{4.9}$$

$$U_i - U_j + N * X_{ij} \leq N - 1 \quad \forall i, j, i \neq j \tag{4.8}$$

Οι περιορισμοί 4.2 προσδιορίζουν ότι απο οποιονδήποτε κόμβο i μπορεί μόνο μια φορά να γίνει μετάβαση προς κάποιον άλλο κόμβο j. Με άλλα λόγια είναι αδύνατον να μεταβώ απο τον ίδιο σταθμό σε κάποιον άλλο περισσότερες από μια φορά. Οι 4.3 και 4.4 καθορίζουν ότι πρέπει να εκκινήσουμε απο τον κόμβο αναχώρησης που θα ορίσουμε και να αφιχθούμε στον κόμβο προορισμού. Ουσιαστικά ορίζουν οτι πρέπει οπωσδήποτε απο τον κόμβο O να γίνει μετάβαση σε οποιονδήποτε κόμβο j ενώ από οποιονδήποτε κόμβο i πρέπει να υπάρξει μετάβαση στον κόμβο D. Το σύνολο των περιορισμών που περιγράφονται από τις εξισώσεις 4.5 καθορίζουν τη συνέχεια στο δίκτυο, δηλαδή όταν ο ταξιδιώτης αφιχθεί σε κάποιον κόμβο j θα πρέπει και να αναχωρήσει απο αυτόν εφόσον δεν είναι ο κόμβος τελικού προορισμού. Οι περιορισμοί 4.6 απαγορεύουν να επιστρέψουμε σε κάποιον κόμβο από τον οποίο αναχωρήσαμε νωρίτερα, δηλαδή δείχνει ότι η δρομολόγηση πραγματοποιείται μόνο προς μία κατεύθυνση χωρίς επιστροφή. Οι εξισώσεις 4.7 και 4.8 καθορίζουν ότι στη δρομολόγηση που θα επιλεχθεί δε θα πρέπει να υπάρχει διαδρομή που να καταλήγει στον κόμβο αφετηρίας ή αντίστοιχα που να αναχωρεί απο τον κόμβο προορισμού. Για την εξάλειψη των υποδιαδρομών που δημιουργούνται κατά την επίλυση του προβλήματος είναι απαραίτητος ο περιορισμό 4.9. Δηλαδή σε περίπτωση που επιλεχθεί η διαδρομή ij τότε οι κόμβοι αυτοί απέχουν κατά μια μονάδα και το αριστερό μέλος ισούται με το δεξί. Σε αντίθετη περίπτωση η διαφορά θα είναι μικρότερη η ίση το N-1. Πιο αναλυτικά οι συγκεκριμένοι περιορισμοί εξαναγκάζουν όλους τους σταθμούς του διαγράμματος να συνδεθούν σε τέτοια σειρά ώστε ο μέγιστος αριθμός κόμβων που παρεμβάλλονται ανάμεσα τους να είναι N-1.

Οι περιορισμοί που περιγράφηκαν παραπάνω αφορούσαν το μοντέλο αγνοώντας την ύπαρξη δρομολογίων στα μέσα μεταφοράς. Όσοι παρουσιάζοντα παρακάτω καθορίζουν τις χρονικές στιγμές άφιξης και αναχώρησης απο τους κόμβους.

$$Y_{O0} = 1 \tag{4.14}$$

$$\sum_{n=0}^{4} Y_{in} \leq 1 \quad , \forall i \tag{4.15}$$

$$\sum_{\substack{j=0 \\ i \neq j}}^{N} X_{ij} - \sum_{n=0}^{4} Y_{in} = 0 \quad \forall i \in N \tag{4.13}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{N} X_{ij} - \sum_{n=0}^{4} Z_{jn} = 0 \quad \forall j \in N \tag{4.12}$$

$$Y_{in} - Z_{jn} \leq (1 - X_{ij}) * M \quad \forall i, j, \ i \neq j \tag{4.20}$$

$$\begin{aligned} & d_{ijn} - (3 - X_{ij} - Y_{in} - Z_{jn}) * M \ \leq \ DT_{ijn} \\ & DT_{ijn} \ \leq \ d_{ijn} + (3 - X_{ij} - Y_{in} - Z_{jn}) * M \\ & \forall i, j, n \quad i \neq j \end{aligned} \tag{4.19}$$

$$\begin{aligned} & -X_{ij} * M \ \leq DT_{ijn} \leq \ X_{ij} * M \\ & \forall i, j \ , i \neq j \end{aligned} \tag{4.11}$$

$$\begin{aligned} & -Y_{in} * M \ \leq DT_{ijn} \leq \ Y_{in} * M \\ & \forall i, n \end{aligned} \tag{4.10}$$

$$\begin{aligned} & -Z_{jn} * M \ \leq DT_{ijn} \leq \ Z_{jn} * M \\ & \forall j, n \end{aligned} \tag{4.18}$$

$$\begin{aligned} & DT_{ikn} + t_{ik} * X_{ik} - DT_{kjz} \leq (4 - X_{ik} - Y_{in} - X_{kj} - Y_{kz}) * M \\ & \forall i, k, j, n, z \quad i \neq j, i \neq k, j \neq k \end{aligned} \tag{4.17}$$

$$\begin{aligned} & DT_{ijn} + t_{ij} * X_{ij} - (2 - X_{ij} - Z_{jn}) * M \leq AT_{jin} \\ & AT_{jin} \leq DT_{ijn} + t_{ij} * X_{ij} + (2 - X_{ij} - Z_{jn}) * M \\ & \forall i, j, n \ \ i \neq j \end{aligned} \tag{4.16}$$

$$-X_{ij} * M \leq AT_{jin} \leq X_{ij} * M \quad \forall i, j \ i \neq j \tag{4.21}$$

$$-Z_{jn} * M \leq AT_{jin} \leq Z_{jn} * M \quad \forall j, n \tag{4.22}$$

$$\sum_{i=0}^{N} \sum_{n=0}^{4} AT_{Din} \leq ToA \quad , i \neq D \tag{4.23}$$

Ο περιορισμός 4.10 δείχνει την υπόθεση που κάνουμε ότι ο ταξιδιώτης θα αναχωρεί κάθε φορά απο τον κόμβο αφετηρίας Ο με το πρώτο δρομολόγιο(n=0). Οι εξισώσεις 4.11 εν συνεχεία ορίζουν ότι απο κάθε κόμβο υπάρχει η δυνατότητα να αναχωρήσουμε μόνο μια φορά με κάποιο συγκεκριμένο δρομολόγιο. Το σύνολο των εξισώσεων 4.12 και 4.13 καθορίζουν τη σχέση μεταξύ της μεταβλητής απόφασης X_{ij} με την Y_{in} και την Z_{jn}, με άλλο λόγια όταν η X_{ij} λαμβάνει την τιμή 1 τότε σίγουρα θα αναχωρήσουμε απο τον κόμβο i και θα αφιχθούμε στον κόμβο j, οπότε και οι μεταβλητές Y_{in}, Z_{jn} θα λάβουν την τιμή 1 για οποιδήποτε δρομολόγιο n. Οι περιορισμοί 4.14 δηλώνουν επιπλέον τη σχέση που συνδέει τις μεταβλητές Y_{in} και Z_{jn} μεταξύ τους, δηλαδή όταν η μία γίνεται μονάδα θα πρέπει και η άλλη να γίνει μονάδα για το συγκεκριμένο δρομολόγιο n. Η μεταβλητή απόφασης DT_{ijn} θα πρέπει να παίρνει τους καθορισμένους χρόνους αναχώρησης των μέσων απο τον πίνακα d_{ijn}, των διαδρομών που επιλέγονται και αυτό καθορίζεται απο τον περιορισμό 4.15. Οι 4.16, 4.17 και 4.18 δηλώνουν ότι σε περίπτωση που μια διαδρομή δεν επιλεχτεί τότε ο χρόνος αναχώρησης απο τον κόμβο αυτό θα λάβει την τιμή 0. Οι περιορισμοί 4.19 δείχνουν τη συνέχεια μεταξύ των χρόνων αναχώρησης για διαδοχικούς κόμβους. Ομοίως οι εξισώσεις 4.20 ορίζουν τις τιμές που θα λάβει η μεταβλητή απόφασης που αφορά τον χρόνο άφιξης στους κόμβους που θα επιλεχτούν και ανάλογα οι περιορισμοί 4.21 και 4.22 μηδενίζουν τη μεταβλητή όταν ο ταξιδιώτης δε θα μεταβεί στον κόμβο j. Τέλος ο περιορισμός 4.23 καθορίζει τον μέγιστο επιτρεπτό χρόνο άφιξης στον προορισμός μας.

$$\sum_{i=0}^{N} X_{iP} = 1 \quad , i \neq P \tag{4.24}$$

$$AT_{Pin} + WTP - DT_{Pjn} \leq (3 - X_{Pj} - Y_{Pn} - Z_{Pn}) * M \quad \forall i, j, n \ i \neq j, P \neq i, P \neq j \tag{4.25}$$

Στην περίπτωση που επιθυμούμε να επισκεφτούμε κάποιο σημείο ενδιαφέροντος πριν αφιχθούμε στον τελικό μας προορισμό θα εισάγουμε στο μοντέλο τους παραπάνω περιορισμούς. Οι περιορισμού 4.24 δηλώνουν ότι πρέπει αναγκαστηκά να αφιχθούμε στον κόμβο P ενώ με τον 4.25 καθορίζουμε τον χρονικό διάστημα που επιθμούμε να παραμείνουμε στο σημείο αυτό. Επιπλέον στη περίπτωση που θέλουμε να εισάγουμε σημείο ενδιάφεροντος στη διαδρομή αρκεί να παραλείψουμε τους περιορισμούς 4.2, 4.6 και 4.11 ο οποίοι απαγορεύουν την αναχώρηση από έναν κόμβο πάνω απο μια φορά όπως και την επιστρεοφή σε κάποιον κόμβο.

4.4.2 Μοντέλο M1.1 με Διαφοροποίηση των Τερματικών Σταθμών (*Terminal*)

Στο μοντέλο[2,3,5,6,7,11,15,16] αυτό η θεώρηση για τον ορισμό κάθε πόλης σε 3 κόμβους παραμένει ίδια όπως και όλες οι παράμετροι, με την διαφορά ότι αλλάζουν οι μεταβλητές απόφασης και εν συνεχεία η αντικειμενική συνάρτηση και οι περιορισμοί.

- Μεταβλητές Απόφασης

X_{ijn} : Λαμβάνει την τιμή 1 εάν επιλέγεται η διαδρομή i,j με το δρομολόγιο n και 0 διαφορετικά

DT_{ijn} : Ώρα αναχώρησης απο τον κόμβο i στον κόμβο j με το δρομολόγιο n

AT_{jin} : Ώρα άξιφης στον κόμβο j απο τον κόμβο i με το δρομολόγιο n

U_i : Σειρά με την οποία ο κόμβος i έχει επισκεφτεί

Η ουσιαστική διαφορά σε σύγκριση με το προηγούμενο μοντέλο M1 έγκειται στους δείκτες της μεταβλητής X οι οποίοι αυξάνονται κατά έναν, αλλά το σύνολο των μεταβλητών απόφασης μειώνονται κατα δύο (2) διότι δε χρειάζονται πλέον οι Y και Z. Η αντικειμενική συνάρτηση και οι περιορισμοί φάινονται απο τις παρακάτω εξισώσεις

$$Min\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{n=0}^{4}X_{ijn}C_{ij} \quad ,i \neq j \qquad \textbf{(4.26)}$$

$$\sum_{j=0}^{N}\sum_{n=0}^{4}X_{ijn} \leq 1 \quad ,\forall i \neq j \qquad \textbf{(4.27)}$$

$$\sum_{j=0}^{N} X_{Oj0} = 1 \quad , j \neq O \tag{4.28}$$

$$\sum_{i=0}^{N} \sum_{n=0}^{4} X_{iDn} = 1 \quad , i \neq D \tag{4.29}$$

$$\sum_{\substack{i=0 \\ i \neq D}}^{N} \sum_{n=0}^{4} X_{ijn} - \sum_{\substack{k=0 \\ k \neq O}}^{N} \sum_{n=0}^{4} X_{jkn} = 0 \quad , \forall j \neq D \tag{4.30}$$

$$\sum_{j=0}^{N} \sum_{n=0}^{4} X_{Djn} = 0 \quad , j \neq D \tag{4.31}$$

$$\sum_{i=0}^{N} \sum_{n=0}^{4} X_{iOn} = 0 \quad , i \neq O \tag{4.32}$$

$$X_{ijn} + X_{jin} \leq 1 \quad , \forall i, j, n \ , i \neq j \tag{4.33}$$

$$U_i - U_j + N * X_{ijn} \leq N - 1 \quad , \forall i, j, n, i \neq j \tag{4.34}$$

Το σύνολο των περιορισμών που καθορίζονται απο τις εξισώσεις 4.27 έως 4.34 έχουν ακριβώς την ίδια σημασία με τους περιορισμούς του μοντέλου M1 που αναλύθηκαν παραπάνω με τη μόνη διαφορά τον δείκτη n που καθορίζουν το δρομολόγιο. Ομοίως παρακάτω θα παρουσιαστούν οι περιορισμοί που αφορούν τους χρόνους αναχώρησης και άφιξης οι οποίοι καθορίζονται απο τον πίνακα δρομολογίων.

$$\begin{aligned} & d_{ijn} - (1 - X_{ijn}) * M \leq DT_{ijn} \\ & DT_{ijn} \leq d_{ijn} + (1 - X_{ijn}) * M \\ & \forall i, j, n \ , i \neq j \end{aligned} \tag{4.35}$$

$$\begin{aligned} & -X_{ijn} * M \leq DT_{ijn} \leq X_{ijn} * M \\ & \forall i, j, n, \ i \neq j \end{aligned} \tag{4.36}$$

$$DT_{ikn} + t_{ik} * X_{ikn} - DT_{kjz} \le (2 - X_{ikn} - X_{kjz}) * M \quad \textbf{(4.37)}$$
$$\forall i,k,j,n,z \ , i \ne j, i \ne k$$

$$DT_{ijn} + t_{ij} * X_{ijn} - (1 - X_{ijn}) * M \le AT_{jin} \quad \textbf{(4.38)}$$
$$AT_{jin} \le DT_{ijn} + t_{ij} * X_{ijn} + (1 - X_{ijn}) * M$$
$$\forall i,j,n \ , i \ne j$$

$$-X_{ijn} * M \le AT_{jin} \le X_{ijn} * M \quad \textbf{(4.39)}$$
$$\forall i,j,n \ , i \ne j$$

$$\sum_{i=0}^{N} \sum_{n=0}^{4} AT_{Din} \le ToA \quad \textbf{(4.40)}$$

Τέλος για την προσθήκη του σημείου ενδιαφέροντος απαλείφουμε τους περιορισμούς 4.27, 4.32, 4.33 και να εισάγουμε τους παρακάτω :

$$\sum_{i=0}^{N} \sum_{n-0}^{4} X_{iPn} = 1 \ , i \ne P \quad \textbf{(4.41)}$$

$$AT_{Pin} + WTP - DT_{Pjn} \le (1 - X_{Pjn}) * M \quad \textbf{(4.42)}$$
$$\forall i,j,n \ , i \ne j$$

4.4.3 Μοντέλο Μ2 με Αντιστοίχηση Κόμβου με Πόλη

Στο μοντέλο[2,3,5,7,12,14,15,16] αυτό κάθε πόλη αντιστοιχεί σε έναν και μόνο κόμβο χωρίς να υπάρχουν διαφορετικοί σταθμοί (*terminals*) για κάθε μέσο. Το κάθε μέσο μεταφοράς χαρακτηρίζεται πλέον απο έναν επιπλέον δείκτη m που προστίθεται σε κάθε μια απο τις μεταβλητές

ΠΟΛΗ	ΚΟΜΒΟΣ	ΠΟΛΗ	ΚΟΜΒΟΣ
ΑΘΗΝΑ	0	ΚΑΤΕΡΙΝΗ	8
ΛΑΡΙΣΑ	1	ΤΡΙΚΑΛΑ	9
ΒΟΛΟΣ	2	ΠΑΤΡΑ	10
ΘΕΣ/ΝΙΚΗ	3	ΗΓΟΥΜΕΝΙΤΣΑ	11
ΦΑΡΣΑΛΑ	4	ΚΑΒΑΛΑ	12
ΚΑΡΔΙΤΣΑ	5	ΞΑΝΘΗ	13
ΙΩΑΝΝΙΝΑ	6	ΗΡΑΚΛΕΙΟ	14
ΑΛΕΞ/ΠΟΛΗ	7		

Πίνακας 4.2 : Κωδικοποίηση αριθμού πόλεων μοντέλου Μ2

απόφασης. Συνεπώς ο αριθμός των κόμβων N μειώνεται στο υποτριπλάσιο διότι ενώ στην προηγούμενη μοντελοποίηση είχαμε συνολικά 43 ενώ στη περίπτωση αυτή 15.

Συνεπώς οι παράμετροι, η αντικειμενική συνάρτηση θα μεταβληθούν όπως φαίνεται παρακάτω:

- *Δείκτες :*

i, j, k : Οι δείκτες όλων των κόμβων i, j, k = 0, 1, ..., N
m,c : Οι δείκτες των μέσων μεταφοράς m,n = 0,1,2,3
n, z : Οι δείκτες των δρομολογίων n, z = 0, 1, 2, 3, 4

- *Σύνολα :*

N : Ο αριθμός των κόμβων
Z : Το πλήθος των δρομολογίων
K : Το πλήθος των μέσων μεταφοράς

- *Παράμετροι*

O : Ο κόμβος εκκίνησης (Ορίζεται απο τον χρήστη)
D : Ο κόμβος άφιξης (Ορίζεται απο τον χρήστη)
P : Ο κόμβος - σημείο ενδιαφέροντος που επιθυμεί να σταματήσει ο ταξιδιώτης (Ορίζεται απο τον χρήστη)
WTP : Χρονικό διάστημα παραμονής στο P (Ορίζεται απο τον χρήστη)
ToA : Επιθυμητή ώρα άφιξης στον προορισμό D (Ορίζεται απο τον χρήστη)
C_{ijm} : Κόστος μετακίνησης από τον κόμβο i στον κόμβο j με το μέσο m
t_{ijm} : Χρόνος μετακίνησης απο τον κόμβο i στον κόμβο j με το μέσο m
d_{ijmn} : Ώρα αναχώρησης απο τον κόμβο i προς τον κόμβο j με το μέσο m και με το δρομολόγιο n
M : Μεγάλος αριθμός

Όμοια σε πόλεις που μεταξύ τους δεν υπάρχει απευθείας σύνδεση με κάποιο συγκεκριμένο μέσο το κόστος και ο χρόνος μετακίνησης ορίζεται με έναν μεγάλο αριθμό.

- Μεταβλητές Απόφασης

X_{ijm} : Λαμβάνει την τιμή 1 εάν επιλέγεται η διαδρομή i,j με το μέσο m και 0 σε αντίθετη περίπτωση

Y_{imn} : Λαμβάνει την τιμή 1 εάν ο ταξιδιώτης αναχωρεί απο τον κόμβο i με το μέσο m και με το δρομολόγιο n και 0 διαφορετικά

Z_{jmn} : Λαμβάνει την τιμή 1 εάν ο ταξιδιώτης αφικνείται στον κόμβο j με το μέσο m και με το δρομολόγιο n και 0 διαφορετικά

DT_{ijmn} : Ώρα αναχώρησης απο τον κόμβο i στον κόμβο j με το μέσο m και με το δρομολόγιο n

AT_{jimn} : Ώρα άξιφης στον κόμβο j απο τον κόμβο i με το το μέσο m και με το δρομολόγιο n

U_i : Σειρά με την οποία ο κόμβος i έχει επισκεφτεί

Το συγκεκριμένο μοντέλο όπως γίνεται αντιληπτό είναι παρόμοιο με το M1 που παρουσιάστηκε παραπάνω. Παρακάτω παρουσιάζεται η αντικειμενική συνάρτηση, η οποία ομοίως ελαχιστοποιεί το συνολικό κόστος ταξιδίου, και εν συνεχεία και οι περιορισμοί.

$$Min\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{m=0}^{3} X_{ijm}C_{ijm}\ ,\quad i \neq j \qquad \textbf{(4.43)}$$

$$\sum_{j=0}^{N}\sum_{m=0}^{3} X_{ijm} \leq 1 \quad ,\forall i \neq j \qquad \textbf{(4.44)}$$

$$\sum_{j=0}^{N}\sum_{m=0}^{3} X_{Ojm} = 1 \quad ,O \neq j \qquad \textbf{(4.46)}$$

$$\sum_{i=0}^{N}\sum_{m=0}^{3} X_{iDm} = 1 \quad ,i \neq D \qquad \textbf{(4.47)}$$

$$\sum_{\substack{i=0 \\ i\neq D}}^{N}\sum_{m=0}^{3} X_{ijm} - \sum_{\substack{k=0 \\ k\neq O}}^{N}\sum_{m=0}^{3} X_{jkm} = 0 \quad ,\forall j \neq D \qquad \textbf{(4.48)}$$

$$\sum_{j=0}^{N}\sum_{m=0}^{3} X_{Djm} = 0 \quad ,j \neq D \qquad \textbf{(4.45)}$$

$$\sum_{i=0}^{N}\sum_{m=0}^{3} X_{iOm} = 0 \quad , i \neq D \tag{4.49}$$

$$X_{ijm} + X_{jim} \leq 1 \quad , \forall i, j, m, i \neq j \tag{4.50}$$

$$U_i - U_j + N * X_{ijm} \leq N - 1 \quad , \forall i, j, m, i \neq j \tag{4.51}$$

Όπως αναλύθηκε και στο μοντέλο M1 οι εξισώσεις 4.44 καθορίζουν ότι απο κάθε κόμβο μπορούμε να αναχωρήσουμε μόνο μια φορά προς οποιοδήποτε άλλο κόμβο j και με οποιοδήποτε μέσο m. Οι περιορισμοί 4.45 και 4.46 δηλώνουν την αναχώρηση και άφιξη στην αφετηρία και στον προορισμό αντίστοιχα ενώ η 4.47 είναι οι περιορισμοί συνέχειας. Οι 4.48,4.49 και 4.50 δηλώνουν ότι η δρομολόγηση πραγματοποιείται μόνο προς μία κατεύθυνση. Τέλος το σύνολο των περιορισμών 4.51 απαλείφουν τις υποδιαδρομές που δημιουργούνται κατα την επίλυση του προβλήματος. Οι επόμενο περιορισμοί όπως και στη προηγούμενη ενότητα αφορούν τους περιορισμούς χρόνου άφιξης και αναχώρησης.

$$\sum_{m=0}^{3} Y_{Om0=1} \tag{4.53}$$

$$\sum_{m=0}^{3}\sum_{n=0}^{4} Y_{imn} \leq 1 \tag{4.52}$$

$$\sum_{\substack{j=0 \\ i \neq j}}^{N} X_{ijm} - \sum_{n=0}^{4} Y_{imn} = 0 \quad , \forall i, m \tag{4.55}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{N} X_{ijm} - \sum_{n=0}^{4} Z_{jmn} = 0 \quad , \forall j, m \tag{4.54}$$

$$Y_{imn} - Z_{jmn} \le (1 - X_{ijm}) * M \qquad \textbf{(4.56)}$$
$$\forall i, j, m, n \quad , i \ne j$$

$$d_{ijmn} - (3 - X_{ijm} - Y_{imn} - Z_{jmn}) * M \le DT_{ijmn} \qquad \textbf{(4.57)}$$
$$DT_{ijmn} \le d_{ijmn} + (3 - X_{ijm} - Y_{imn} - Z_{jmn}) * M$$
$$\forall i, j, m, n \quad , i \ne j$$

$$-X_{ijm} * M \le DT_{ijmn} \le X_{ijm} * M \qquad \textbf{(4.58)}$$
$$\forall i, j, m \quad , i \ne j$$

$$-Y_{imn} * M \le DT_{ijmn} \le Y_{imn} * M \qquad \textbf{(4.59)}$$
$$\forall i, j, m, n \quad , i \ne j$$

$$-Z_{jmn} * M \le DT_{ijmn} \le Z_{jmn} * M \qquad \textbf{(4.60)}$$
$$\forall i, j, m, n \quad , i \ne j$$

$$DT_{ijmn} + t_{ijm} * X_{ijm} - (2 - X_{ijm} - Z_{jmn}) * M \le AT_{jimn} \qquad \textbf{(4.61)}$$
$$AT_{jimn} \le DT_{ijmn} + t_{ijm} * X_{ijm} + (2 - X_{ijm} - Z_{jmn}) * M$$
$$\forall i, j, m, n, \quad , i \ne j$$

$$-X_{ijm} * M \le AT_{jimn} \le X_{ijm} * M \qquad \textbf{(4.62)}$$
$$\forall i, j, m, n, \quad , i \ne j$$

$$-Z_{jmn} * M \le AT_{jimn} \le Z_{jmn} * M \qquad \textbf{(4.63)}$$
$$\forall i, j, m, n, \quad , i \ne j$$

$$DT_{ikmn} + t_{ikm} * X_{ikm} - DT_{kjcz} \leq (4 - X_{ikm} - Y_{imn} - X_{kjc} - Y_{kcz}) * M \quad \forall i,k,j,m,c,n,z \quad , i \neq k, k \neq j \qquad \textbf{(4.64)}$$

$$\sum_{\substack{i=0 \\ i \neq D}}^{N} \sum_{m=0}^{3} \sum_{n=0}^{4} AT_{Dimn} \leq ToA \qquad \textbf{(4.65)}$$

Οι περιορισμολι 4.52 και 4.53 καθορίζουν την αναχώρηση απο τον αρχικό κόμβο με το πρώτο δρομολόγιο και παράλληλα ότι απο κάθε κόμβο προβλέπεται να αναχωρήσει ο ταξιδιώτης μόνο μια φορά. Οι εξισώσεις 4.54, 4.55, 4.56 καθορίζουν τη σχέση μεταξύ των μετ;αβλητών απόφασης X_{ijm},Y_{imn} και Z_{jmn} , δηλαδή όταν επιλέγεται η διαδρομή ij τότε με κάποιο δρομολόγιο n θα αναχωρήσουμε απο τον κόμβο i και αντίστοιχα θα αφιχθούμε με το ίδιο δρομολόγιο n στον κόμβο j. Εν συνεχεία το σύνολο των περιορισμών 4.57 έως και 4.60 καθορίζουν τις ώρες αναχώρησης που θα λάβει η μεταβλητή απόφασης DT_{ijmn} απο τον πίνακα με τα στοιχεία δρομολογίων για τις διαδρομές που θα επιλεχθούν, ενω σε αντίθετη περίπτωση θα λάβει την τιμή 0. Ομοίως και οι περιορισμοί 4.61 με 4.63 για τους χρόνους άφιξης στον κόμβο j. Οι περιορισμοί της εξίσωσης 4.64 καθορίζουν τη σχέση μεταξύ των διαδοχικών σημείων αναχώρησης ikj, με άλλα λόγια η αναχώρηση απο τον κόμβο k για τον j επιβάλλεται να είναι χρονικά μετά την αφιξη του στον k απο τον i.Τέλος οι περιορισμοί 4.65 καθορίζουν τον μέγιστο χρόνο που είναι δυνατόν να διαρκέσει το ταξίδι. Ομοίως για να εισάγουμε ένα σημείο ενδιαφέοντος στη διάρκεια του ταξιδιού μας είναι αναγκαίο να παραλείψουμε αρχικά τους περιορισμούς 4.44, 4.49, 4.50 καθώς και τους 4.53 οι οποίοι απαγορεύουν την αναχώρηση απο τον ίδιο κόμβο πάνω απο μία φορά και να προσθέσουμε τους παρακάτω.

$$\sum_{i=0}^{N} \sum_{m=0}^{3} X_{iPm} = 1 \quad , i \neq P \qquad \textbf{(4.66)}$$

$$AT_{Pimn} + WTP - DT_{Pjmn} \leq (3 - X_{Pjm} - Y_{Pmn} - Z_{Pmn}) * M \quad \forall i,j,m,n \quad , i \neq j \qquad \textbf{(4.67)}$$

4.4.4 Μοντέλο M2.1 με Αντιστοίχηση Κόμβου με Πόλη

Σε αντιστοιχία με το μοντέλο M1.1 που παρουσιάστηκε στην υποενότητα 4.4.2 όμοια και στο μοντέλο[2,3,5,6,7,11,12,,15,16] αυτό η θεώρηση για τον ορισμό κάθε πόλης με έναν και μόνο κόμβο παραμένει ίδια όπως και όλες οι παράμετροι, με την διαφορά ότι μεταβάλλονται οι μεταβλητές απόφασης και εν συνεχεία η αντικειμενική συνάρτηση και οι περιορισμοί.

- Μεταβλητές Απόφασης

X_{ijmn} : Λαμβάνει την τιμή 1 εάν επιλέγεται η διαδρομή i,j με το μέσο m και με το με το δρομολόγιο n, ενώ 0 διαφορετικά

DT_{ijmn} : Ώρα αναχώρησης απο τον κόμβο i στον κόμβο j με το μέσο m με το δρομολόγιο n, ενώ 0 διαφορετικά

AT_{jimn} : Ώρα άξιφης στον κόμβο j απο τον κόμβο i με το μέσο m με το δρομολόγιο n ενώ 0 διαφορετικά

U_i : Σειρά με την οποία ο κόμβος i έχει επισκεφτεί

Εν συγκρίση με το μοντέλο M2.1 η διαφορά έγκειται στους δείκτες της μεταβλητής X οι οποίοι αυξάνονται κατά έναν, αλλά το σύνολο των μεταβλητών απόφασης μειώνονται κατα δύο (2) διότι δε χρειάζονται πλέον οι Y και Z. Η αντικειμενική συνάρτηση και οι περιορισμοί φάινονται απο τις παρακάτω εξισώσεις.

$$Min\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4}X_{ijmn}C_{ijm} \quad ,i \neq j \tag{4.68}$$

$$\sum_{j=0}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4}X_{ijmn} \leq 1 \quad ,\forall i \neq j \tag{4.71}$$

$$\sum_{j=0}^{N}\sum_{m=0}^{3}X_{Ojm0} = 1 \quad ,\forall j \neq O \tag{4.69}$$

$$\sum_{i=0}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4}X_{iDmn} = 1 \quad ,\forall i \neq D \tag{4.70}$$

$$\sum_{\substack{i=0\\ i\neq j}}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4} X_{ijmn} - \sum_{\substack{k=0\\ k\neq j}}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4} X_{jkmn} = 0 \qquad \textbf{(4.72)}$$

$$\sum_{j=0}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4} X_{Djmn} = 0 \quad , j \neq D \qquad \textbf{(4.73)}$$

$$\sum_{i=0}^{N}\sum_{m=0}^{3}\sum_{n=0}^{4} X_{iOmn} = 0 \quad , i \neq O \qquad \textbf{(4.74)}$$

$$X_{ijmn} + X_{jimn} \leq 1 \qquad \textbf{(4.75)}$$
$$\forall i, j, m, n \quad , i \neq j$$

$$U_i - U_j + N * X_{ijmn} \leq N - 1 \qquad \textbf{(4.76)}$$
$$\forall i, j, m, n \quad , i \neq j$$

Ομοίως οι παραπάνω περιορισμοί καθορίζουν ακριβώς ότι αναφέραμε και στο μοντέλο Μ2. Οι περιορισμοί 4.69 δηλώνουν ότι μια διαδρομή είναι δυνατόν να επιλέγεται το πολύ μία φορά για κάθε δρομολόγιο και για κάθε μέσο ενώ οι 4.70 και 4.71 καθορίζουν την αναχώρηση και την άφιξη απο την αφετηρία και στον προορισμό αντίστοιχα. Το σύνολο των εξισώσεων που περιγράφονται απο την 4.72 καθορίζουν τη συνέχεια που πρέπει να υπάρχει μεταξύ των διαδοχικών κόμβων, δηλαδή την αναχώρηση απο έναν κόμβο που προηγουμένως έχουμε αφιχθεί χωρίς αυτός να είναι ο τελικός προορισμός μας. Οι 4.73, 4.74 και 4.75 καθορίζουν οτι η δρομολόγηση είναι μονο προς μια κατεύθυνση ενώ οι 4.76 εξαλήφει τις υποδιαδρομές. Όσον αφορά τους χρόνους άφιξης και αναχώρησης αυτοί καθορίζονται απο τους παρακάτω.

$$d_{ijmn} - (1 - X_{ijmn}) * M \leq DT_{ijmn} \qquad \textbf{(4.77)}$$
$$DT_{ijmn} \leq d_{ijmn} + (1 - X_{ijmn}) * M$$
$$\forall i, j, m, n \quad , i \neq j$$

$$-X_{ijmn} * M \leq DT_{ijmn} \leq X_{ijmn} * M \qquad \textbf{(4.78)}$$
$$\forall i, j, m, n \quad , i \neq j$$

$$DT_{ikmn} + t_{ikm} * X_{ikmn} - DT_{kjcz} \leq (2 - X_{ikmn} - X_{kjcz}) * M \qquad \textbf{(4.79)}$$
$$\forall i, k, j, m, n, c, z$$
$$i \neq k, k \neq j$$

$$DT_{ijmn} + t_{ijm} * X_{ijmn} - (1 - X_{ijmn}) * M \leq AT_{jimn} \quad (4.80)$$
$$AT_{jimn} \leq DT_{ijmn} + t_{ijm} * X_{ijmn} + (1 - X_{ijmn}) * M$$
$$\forall i, j, m, n \quad , i \neq j$$

$$-X_{ijmn} * M \leq AT_{jimn} \leq X_{ijmn} * M \quad (4.81)$$
$$\forall i, j, m, n \quad , i \neq j$$

$$\sum_{i=0}^{N} \sum_{m=0}^{3} \sum_{n=0}^{4} AT_{Dimn} \leq ToA \quad , i \neq D \quad (4.82)$$

Οι περιορισμοί που περιγράφονται στην 4.77 και 4.78 καθορίζουν τους χρόνους αναχώρησης ενώ επιθυμούμε να λαμβάνουν την τιμή 0 όταν δεν πραγματοποιείται η διαδρομή ij.Οι εξισώσεις 4.79 καθορίζουν την συνέχεια μεταξύ των χρόνων αναχώρησης. Οι 4.80 και 4.81 τους χρόνους άφιξης στους κόμβους που έχουν επιλεχθεί και επιπλέον οριοθετούμε τον μέγιστο χρόνο άφιξης στον προορισμό μέσω του 4.82. Τέλος για το σημείο ενδιαφέροντος απαλείφουμε τους 4.69, 4.74, 4.75 και προσθέτουμε τους παρακάτω.

$$\sum_{i=0}^{N} \sum_{m=0}^{3} \sum_{n=0}^{4} X_{iPmn} = 1 \quad , i \neq P \quad (4.83)$$

$$AT_{Pimn} + WTP - DT_{Pjmn} \leq (1 - X_{Pjmn}) * M \quad (4.84)$$
$$\forall i, j, m, n \quad , i \neq P, j \neq P$$

Τέλος οι 4.83 και 4.84 καθορίζουν αφενός μεν την άφιξη στο σημείο ενδιαφέροντος της επιθυμίας μας αλλά και τον χρόνο παραμονής σε αυτό.

4.5 **Σύνοψη**

Στο συγκεκριμένο κεφάλαιο αναλύθηκε το πρόβλημα δρομολόγησης που πραγματεύεται η παρούσα διπλωματική στην επικράτεια της Ελλάδας με 4 διαφορετικά μέσα μεταφοράς. Παρουσιάστηκαν 4 διαφορετικά μαθηματικά μοντέλα του προβλήματος τα οποία προσδιορίζουν ακριβώς την βέλτιστη λύση. Οι μοντελοποιήσεις βασίζονται σε μικτό γραμμικό ακέραιο προγραμματισμό και είναι δυνατόν με διαφορετικά δεδομένα εισαγωγής κόστους μετακινήσεων, χρόνουμ ετακινήσεων και δρομολογίων να εφαρμοστούν και να αποδώσουν λύση για οποιοδήποτε

δίκτυο πόλεων. Στο επόμενο παρουσιάζονται τα αποτελέσματα που προέκυψαν απο την επίλυση των μοντέλων αυτών.

ΚΕΦΑΛΑΙΟ ΠΕΜΠΤΟ:
Επίλυση Μοντέλων

5.1 Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζονται τα αποτελέσματα απο την επίλυση των μοντέλων που παρουσιάστηκαν στο προηγούμενο κεφάλαιο για τρία (3) συγκεκριμένα παραδείγματα δρομολόγησης. Αναλύεται η απόδοση των μοντέλων και πραγματοποιείται μια συγκριτική ανάλυση μεταξύ των 4 αυτών μοντέλων πάνω στα παραδείγματα αυτά.

5.2 Ανάπτυξη κώδικα

Τα μοντέλα που παρουσιάστηκαν στο προηγούμενο κεφάλαιο αναπτύχθηκαν σε γλώσσα προγραμματισμού C++ και το λογισμικό που χρησιμοποιήθηκε ήταν το *Microsoft Visual Studio 2008*. Για την επίλυσή τους χρησιμοποιήθηκε solver *IBM CPLEX Optimization Studio v.12.4*.

Η υλοποίηση των προγραμμάτων έλαβε χώρα στο εργαστήριο Οργάνωνσης Παραγωγής της Πολυτεχνικής σχολής του Πανεπιστημίου Θεσσαλίας, που βρίσκεται στο τμήμα Μηχανολόγων Μηχανικών. Τα χαρακτηριστικά του υπολογιστή που χρησιμοποιήθηκε είναι τα εξής:

- Επεξεργαστής: Intel(R) Pentium(R) D CPU 3.00GHz,
- Εγκατεστημένη μνήμη: 1,00 GB,
- Λογισμικό: Windows 7 Professional 32-bit.

Σε κάθε ένα απο τα τέσσερα (4) μαθηματικά μοντέλα εισάχθηκαν με τη μορφή πινάκων τα δεδομένα χρόνου και κόστους μεταφοράς του οχήματος απο την ιστοσελίδα *www.viamichelin.com* με τις ακόλουθες επιλογές :

- Είδος Οχήματος : Hatchback
- Είδος Καυσίμου : Βενζίνη
- Κόστος Καυσίμου : 1,6 €

Τα αντίστοιχα στοιχεία που αφορούν μετακίνηση με λεωφορεί(ΚΤΕΛ), σιδηρόδρομο(ΟΣΕ) και πλοίο, αντλήθηκαν απο τις αντίστοιχες ιστοσελίδες. Αξίζει να σημειωθεί ότι όλα τα μοντέλα είναι συμμετρικά όσον αφορά τους χρόνους και το κόστος μετακίνησης για κάθε μέσο δηλαδή ισχύει ότι :

- $C_{ij} = C_{ji}$
- $t_{ij} = t_{ji}$

5.3 Επίλυση Μοντέλων

Για την αξιολόγηση των μοντέλων επιλέχθηκαν 3 συγκεκριμένα παραδείγματα δρομολόγησης, τα οποία επιλύθηκαν με το λογισμικό που αναφέρθηκε, και με τα 4 μοντέλα που παρουισάστηκαν. Καθένα απο τα παραδείγματα αυτά εκτελέστηκαν 3 επιπλέον φορές με σταδιακά αυξανόμενο αριθμό κόμβων ενώ το πλήθος των μέσων μεταφοράς παραμένει σταθερός.

5.3.1 Πρώτο Παράδειγμα Δρομολόγησης

Το πρώτο παράδειγμα που θα παρουσιαστεί αφορά την διαδρομή Αθήνα - Θεσσαλονίκη με ενδιάμεσο σταθμό(σημείο ενδιαφέροντος) την πόλη του Βόλου και θα εκτελεστεί για όλα τα μοντέλα ξεχωριστά. Στους ακόλουθους πίνακες 5.1, 5.2, 5.3 και 5.4 φαίνεται ακριβώς το παράδειγμα που εξετάσαμε όπου μεταβάλλεται μόνο ο αριθμός κόμβων για διάφορες τιμές χρόνου παραμονής και ελάχιστου χρόνου άφιξης στον τελικό προορισμό ενώ στα αντίστοιχα διαγράμματα 5.1 και 5.2 παρουσιάζεται και σχηματικά πως αυξάνεται ο χρόνος επίλυσης για κάθε ένα απ τα μοντέλα αυξάνοντας τα πλήθος των κόμβων.

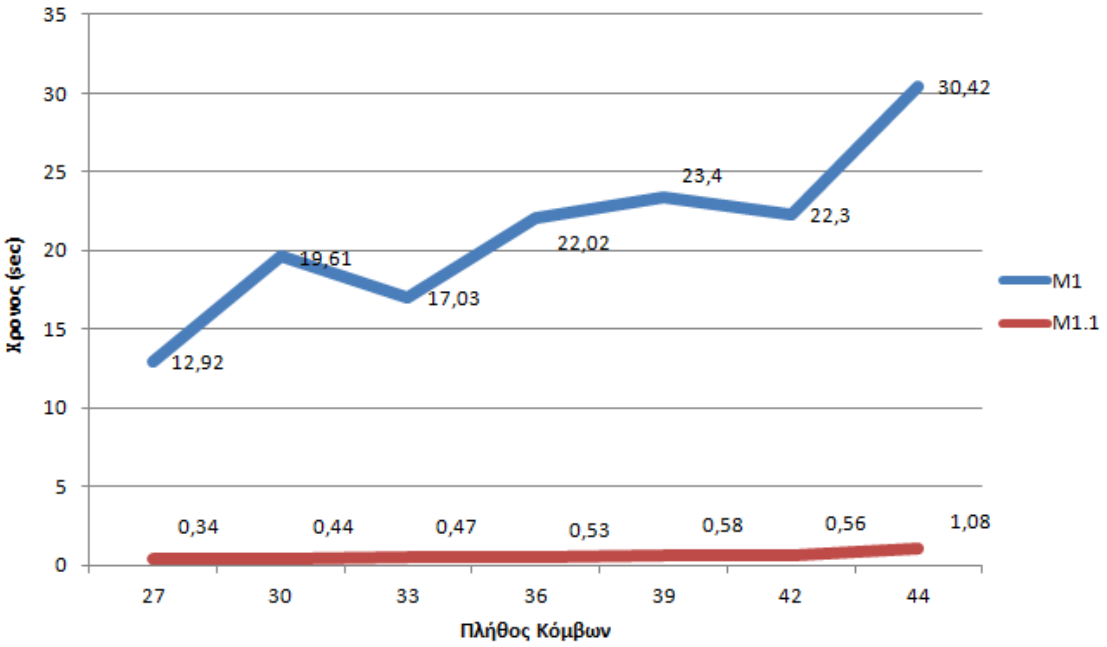
Απο την επίλυση του κώδικα η τιμή της αντικειμενικής που προκύπτει όπως φαίνεται και απο τους πίνακες 5.1 και 5.2 παραμένει ίδια για όσο και αν αυξάνται οι κόμβοι καθότι οι επιπλέον κομβοι που εισάγονται δεν επηρεάζουν τη βέλτιστη διαδρομή. Ωστόσο όπως γίνεται εύκολα αντιληπτό, η τιμή της αντικειμενικής θα ήταν δυνατον να μεταβληθεί στην περίπτωση που εισάγαμε έναν κόμβο διαμέσω το οποίου θα προέκυπτε μια διαδρομή, δεδομένου ότι θα πληρεί τους χρονικούς περιορισμούς καθώς και τον ενδιάμεσο σταθμό. Η διαδρομή που ακολουθήθηκε ήταν η ίδια και ειναι η παρκάτω :

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
27	M1	3	22	12,92	53
30	M1	3	22	19,61	53
33	M1	3	22	16,03	53
36	M1	3	22	22,02	53
39	M1	3	22	23,4	53
42	M1	3	22	22,3	53
44	M1	3	22	30,42	53

Πίνακας 5.1 : Χρόνοι υλοποίησης και τιμή αντικειμενικής μοντέλου M1 1^{ου} παραδείγματος

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
27	M1.1	3	22	0,34	53
30	M1.1	3	22	0,44	53
33	M1.1	3	22	0,47	53
36	M1.1	3	22	0,53	53
39	M1.1	3	22	0,58	53
42	M1.1	3	22	0,56	53
44	M1.1	3	22	1,08	53

Πίνακας 5.2 : Χρόνοι υλοποίησης και τιμή αντικειμενικής μοντέλου M1.1 1^{ου} παραδείγματος



Διάγραμμα 5.1 : Χρόνος υλοποίησης συναρτήση αριθμού κόμβων μοντέλων M1 και M1.1, 1^{ου} παραδείγματος

- Εσωτερική μετακίνηση στη πόλη της Αθήνας και άφιξη στο σταθμό ΚΤΕΛ
  DT = 6, AT = 7
- Αθήνα - Λάρισα με Λεωφορείο
  DT = 7, AT = 10.7
- Λάρισα - Βολος με Λεωφορείο
  DT = 13, AT = 14

- Εσωτερική μετακίνηση στο Βόλο και άφιξη στο PoI
  DT = 14, AT = 14.3
- Εσωτερική μετακίνηηση απο το PoI στο σταθμό Τρένων
  DT = 17, AT = 17.3
- Βόλος - Λάρισα με Τρένο
  DT = 17.5, AT = 18.3
- Λάρισα - Θεσσαλονίκη με Τρένο
  DT = 18.3, AT = 19.8
- Εσωτερική μετακίνηση στη πόλη της Θεσσαλονίκης και άφιξη στο PoI
  DT = 21, AT = 21.5

Απο τους χρόνους αναχώρησης και άφιξης συμπαιρένουμε ότι ικανοποιούνται όλοι οι περιορισμοί που έχουμε θέσει. Δηλαδή ο ταξιδιώτης φθάνει στο PoI στις 14,3 και αναχωρεί απο αυτό στις 17 καθώς επίσης φθάνει στον τελικό του προορισμό στις 21,5. Όπως είναι ευνόητο η διαδρομή αυτή δεν είναι δυνατόν να είναι η μόνη πιθανή ακόμη για την ίδια τιμή της αντικειμενικής. Δηλαδή υπάρχει περίπτωση να εμφανίζονται δυο διαφορετικές διαδρομές οι οποίες θα ικανοποιούν τα κριτήρια και θα εμφανίζουν ίδια τιμή αντικειμενικής συνάρτησης. Αξίζει να σημειωθεί ότι ο χρόνος έχει οριστεί στη κλίμακα του ένα όπως φαίνεται απο τους χρόνους άφιξης και αναχώρησης.

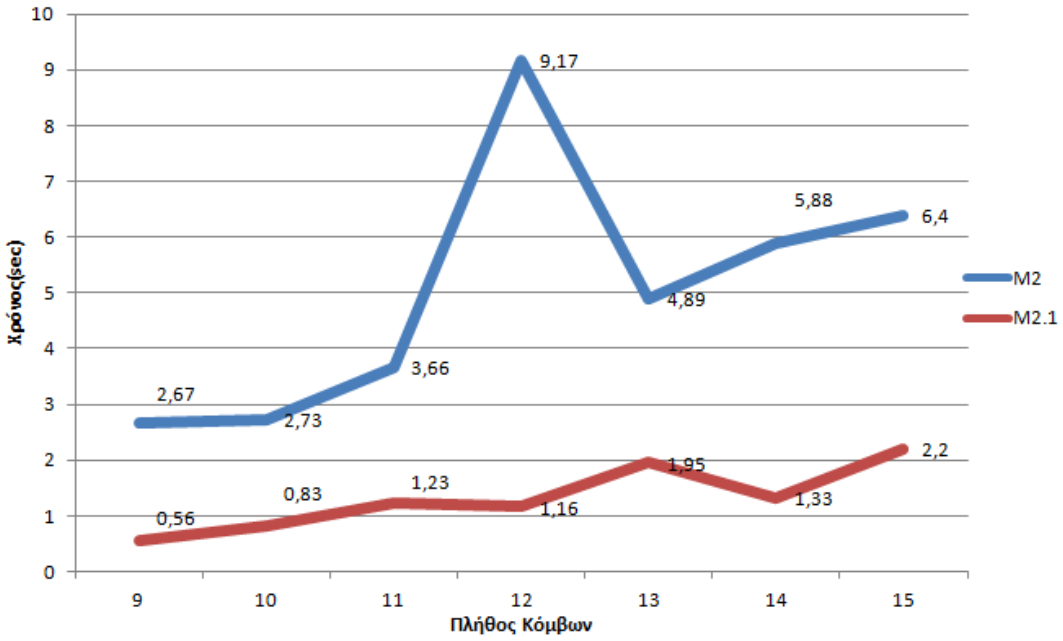
Τα επόμενα δύο μοντέλα M2 και M2.2 εμφανίζουν μειωμένη τιμή αντικειμενικής και μειωμένο χρόνο υλοποίησης συγκριτικά με τα δυο προηγούμενα όπως φαίνεται στους πίνακες 5.3 και 5.4. Η μειωμένη αντικειμενική οφείλεται στο γεγονός ότι δεν υπάρχουν μετακινήσεις στο εσωτερικό των πόλεων καθότι όπως έχει αναλυθεί στο προηγούμενο κεφάλαιο κάθε πόλη αντιπροσωπεύεται καθαρά απο έναν και μόνο κόμβο.

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
9	M2	3	22	2,67	50
10	M2	3	22	2,73	50
11	M2	3	22	3,66	50
12	M2	3	22	9,17	50
13	M2	3	22	4,89	50
14	M2	3	22	5,88	50
15	M2	3	22	6,4	50

Πίνακας 5.3 : Χρόνος υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M2 1^{ου} παραδείγματος

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
9	M2.1	3	22	0,56	50
10	M2.1	3	22	0,83	50
11	M2.1	3	22	1,23	50
12	M2.1	3	22	1,16	50
13	M2.1	3	22	1,95	50
14	M2.1	3	22	1,33	50
15	M2.1	3	22	2,2	50

Πίνακας 5.4 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M2.1 1^{ου} παραδείγματος



Διάγραμμα 5.2 : Χρόνος υλοποίησης συναρτήση αριθμού κόμβων μοντέλων M2 και M2.1, 1^{ου} παραδείγματος

5.3.2 Δεύτερο Παράδειγμα Δρομολόγησης

Στο δεύτερο παράδειγμα θα εξεταστεί η διαδρομή Ιωάννινα - Αθήνα με ενδιάμεσο σταθμό την πόλη της Πάτρας (PoI). Στη περίπτωση επιλέχθηκε μικρότερος χρόνος παραμονής στο PoI και αργότερη άφιξη στον τελικό προορισμό με αποτέλεσμα να επιτρέπουμε στον κώδικα να καταλήξει στην βέλτιστη διαδρομή ταχύτερα. Ομοίως εκτελέστηκε για κάθε ένα μοντέλο ξεχωριστά με

αυξανόμενο αριθμό κόμβων αλλά απο την εισαγωγή της πόλης της Πάτρας και έπειτα. Στους πίνακες 5.5 και 5.6 απεικονίζονται οι χρόνοι υλοποίησης για τα μοντέλα M1 και M1.1 καθώς και η τιμή της αντικειμενικής συνάρτησης.

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
33	M1	2	24	6,77	52,4
36	M1	2	24	6,3	52,4
39	M1	2	24	22,86	52,4
42	M1	2	24	7,38	52,4
44	M1	2	24	9,48	52,4

Πίνακας 5.5 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M1 2^{ου} παραδείγματος

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
33	M1.1	2	24	0,27	52,4
36	M1.1	2	24	0,22	52,4
39	M1.1	2	24	0,25	52,4
42	M1.1	2	24	0,58	52,4
44	M1.1	2	24	0,55	52,4

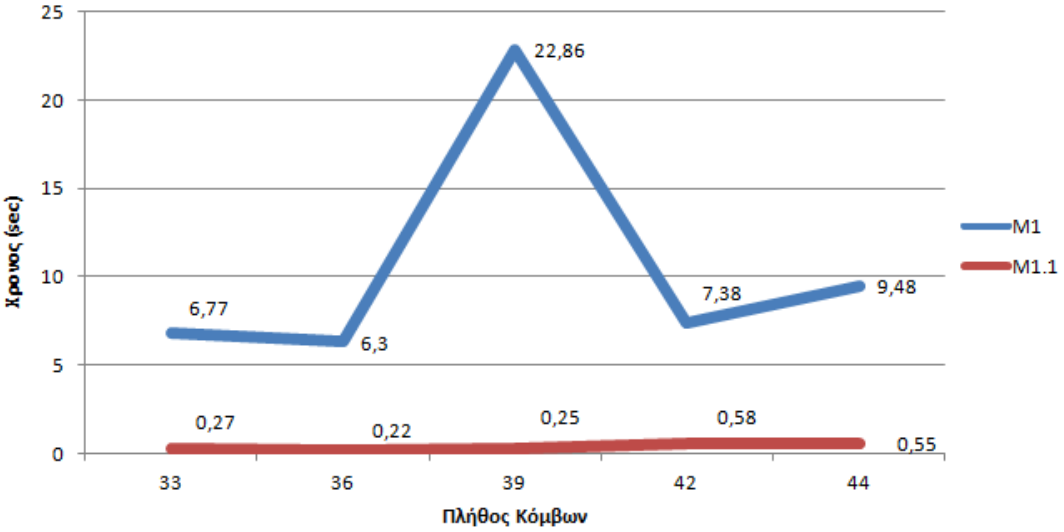
Πίνακας 5.6 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M1.1 2^{ου} παραδείγματος

Απο τους παραπάνω πίνακες συμπαιραίνουμε ότι ομοίως η τιμή της αντικειμενικής συνάρτησης παραμένει σταθερή όσο και αν μεταβάλλονται οι κόμβοι διότι οι προστιθέμενες πόλεις στο πρόβλημα μας δεν είναι δυνατόν να επηρεάσουν και να μεταβάλλουν την βέλτιστη διαδρομή.

Τα μοντέλα M1 και M1.1 έδωσαν μια βέλτιστη διαδρομή ωστόσο με δυο διαφορετικούς χρόνους παραμονής στους σταθμούς. Η διαδρομή είναι η παρακάτω :

- Ιωάννινα - Πάτρα με I.X και άφιξη στο PoI
  DT = 7, AT = 11
- Εσωτερική μετακίνηση στη πόλη της Πάτρας και άφιξη στο σταθμό τρένων
  DT = 17, AT = 17.5
- Πάτρα - Αθήνα με Τρένο
  DT = 19, AT = 20.5
- Εσωτερική μετακίνηση στη πόλη της Αθήνας και άφιξη στον προορισμό

DT = 21 , AT = 22



Διάγραμμα 5.3 : Χρόνος Υλοποίησης συναρτήση αριθμού κόμβων μοντέλων M2 και M2.1, 2^{ου} παραδείγματος

Το WTP έχει οριστεί 2 ώρες και το οποίο και υλοποιείται διότι παραμένει στο σημείο ενδιαφέροντος 6 ώρες ενώ παράλληλα η άφιξη στον τελικό προορισμό είναι στις 22. Καθότι δεν υπάρχει κάποιος επιπλέον περιορισμός που να καθορίζει τον χρόνο παραμονής σε οποιοδήποτε άλλο κόμβο πέραν του σημείου ενδιαφέροντος, η δεύτερη λύση που δόθηκε μεταβάλλει μόνο τον χρόνο παραμονής στο PoI που ωστόσο παραμένει μέσα στο όριο των δύο ωρών που θέσαμε αρχικά και ειναι η παρακάτω :

- Ιωάννινα - Πάτρα με I.X και άφιξη στο PoI
  DT = 7, AT = 11
- Εσωτερική μετακίνηση στη πόλη της Πάτρας και άφιξη στο σταθμό τρένων
  DT = 14, AT = 14.5
- Πάτρα - Αθήνα με Τρένο
  DT = 19, AT = 20.5
- Εσωτερική μετακίνηση στη πόλη της Αθήνας και άφιξη στον προορισμό
  DT = 21 , AT = 22

Τα μονέλα M2 και M2.1 όπως και στο παράδειγμα ένα απέδωσαν τη βέλτιστη λύση σε μικρότερο χρόνο αλλά με μικρότερη τιμή αντικειμενικής συνάρτησης με το M1 και M1.1 καθότι δε συμπεριλαμβάνει το κόστος μετακίνησης εντός των πόλεων.

Η διαδρομή απο τα δύο μοντέλα αυτά ειναι η παρακάτω :

- Ιωάννινα - Πάτρα με I.X και άφιξη στο PoI
  DT = 7, AT = 11
- Πάτρα - Αθήνα με Τρένο
  DT = 19, AT = 20.5

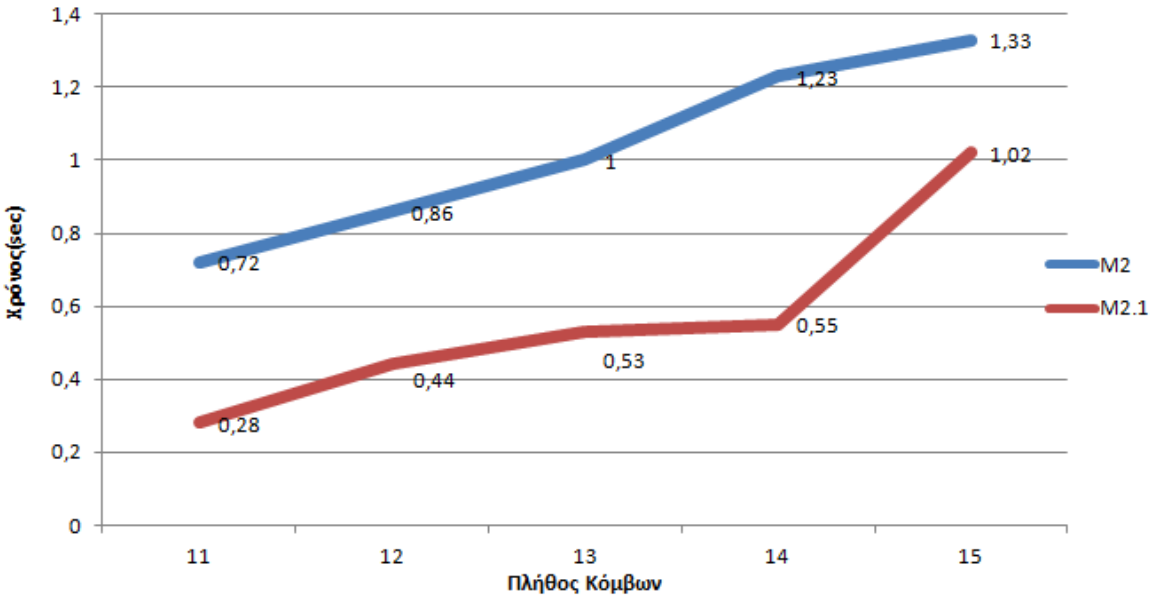
Το συνολικό κόστος μετακίνησης είναι 50, δηλαδή 2,3 μικρότερο απο τα μοντέλα M1 και M1.1 καθότι δεν υπάρχει το κόστος των μετακινήσεων εντός των πόλεων.

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
11	M2	2	24	0,72	50
12	M2	2	24	0,86	50
13	M2	2	24	1	50
14	M2	2	24	1,23	50
15	M2	2	24	1,33	50

Πίνακας 5.7 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M2 2^{ου} παραδείγματος

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
11	M2.1	2	24	0,28	50
12	M2.1	2	24	0,44	50
13	M2.1	2	24	0,53	50
14	M2.1	2	24	0,55	50
15	M2.1	2	24	1,02	50

Πίνακας 5.8 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M2.1 2^{ου} παραδείγματος



Διάγραμμα 5.4 : Χρόνος υλοποίησης συναρτήση αριθμού κόμβων μοντέλων M2 και M2.1, 2^{ου} παραδείγματος

5.3.3 Τρίτο Παράδειγμα Δρομολόγησης

Το τελευταίο παράδειγμα που θα παρουσιαστεί αφορά την διαδρομή Ιωάννινα - Λάρισα με σημείο ενδιαφέροντος την πόλη των Τρικάλων. Εδώ επιλέχτηκε μικρότερη διαδρομή με μεγαλύτερο WTP στο πόλη που επιθυμούμε να επισκεφτούμε αλλά και νωρίτερος χρόνος άφιξης, (ToA) στον προορισμό μας.Στους επόμενους πίνακες φαίνονται οι χρόνοι υλοποίησης των μοντέλων και η τιμή της αντικειμενικής.

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
30	M1	2	21	4,08	22,5
33	M1	2	21	5,28	22,5
36	M1	2	21	8,01	22,5
39	M1	2	21	5,08	22,5
42	M1	2	21	7,11	22,5
44	M1	2	21	6,9	22,5

Πίνακας 5.9 :Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M1 3^{ο} παραδείγματος

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
30	M1.1	2	21	0,23	22,5
33	M1.1	2	21	0,53	22,5
36	M1.1	2	21	0,58	22,5
39	M1.1	2	21	0,59	22,5
42	M1.1	2	21	0,66	22,5
44	M1.1	2	21	0,69	22,5

Πίνακας 5.10 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M1.1 3^{ου} παραδείγματος

Η τιμή της αντικειμενικής συνάρτησης παραμένει σταθερή καθώς αυξάνουμε των αριθμό των κόμβων που εισάγουμε στο πρόβλημα καθώς δεν επηρεάζουν τη βέλτιστη διαδρομή και κατ επέκταση τα μέσα που θα επιλέξει να χρησιμοποιήσει ο ταξιδιώτης. Τα μοντέλα M1 και M1.1. απέδωσαν την παρακάτω διαδρομή :

- Εσωτερική μετακίνηση στη πόλη των Ιωαννίνων και άφιξη στο σταθμό των ΚΤΕΛ
  DT = 6, AT = 6,3
- Ιωάννινα - Τρίκαλα με ΚΤΕΛ
  DT = 9, AT = 11,9
- Εσωτερική μετακίνηση στη πόλη των Τρικάλων και άφιξη στο PoI
  DT = 14, AT = 14.4
- Τρίκαλα - Λάρισα με I.X
  DT = 20 , AT = 21

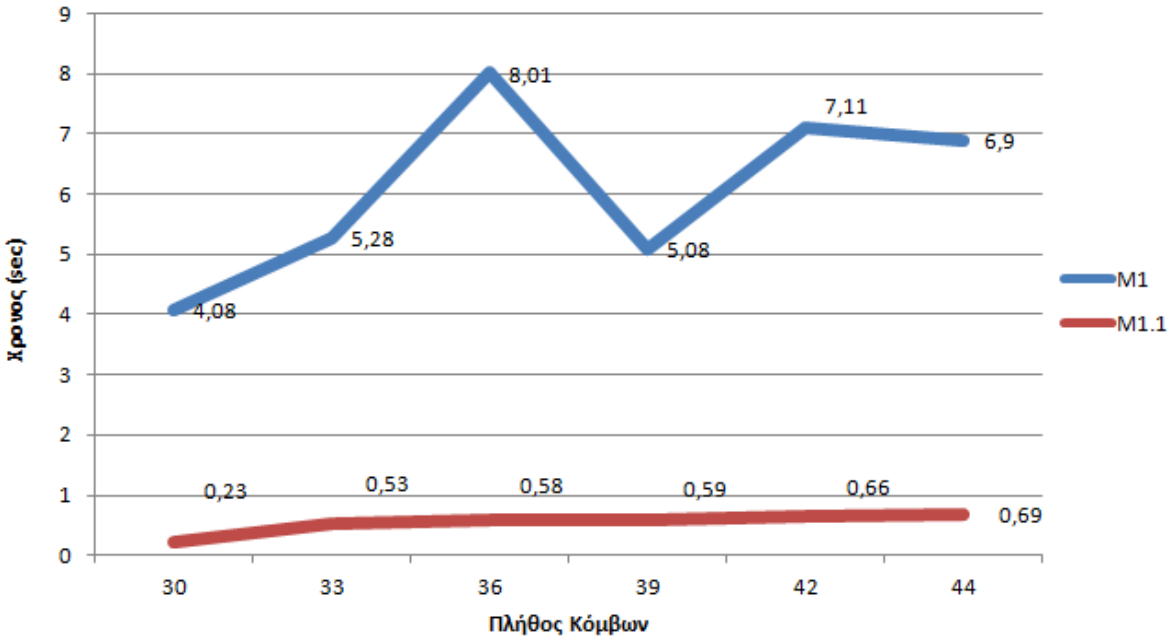
Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
10	M2	2	21	0,55	20
11	M2	2	21	0,75	20
12	M2	2	21	0,94	20
13	M2	2	21	1,13	20
14	M2	2	21	1,36	20
15	M2	2	21	1,41	20

Πίνακας 5.11 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M2 3^{ου} παραδείγματος

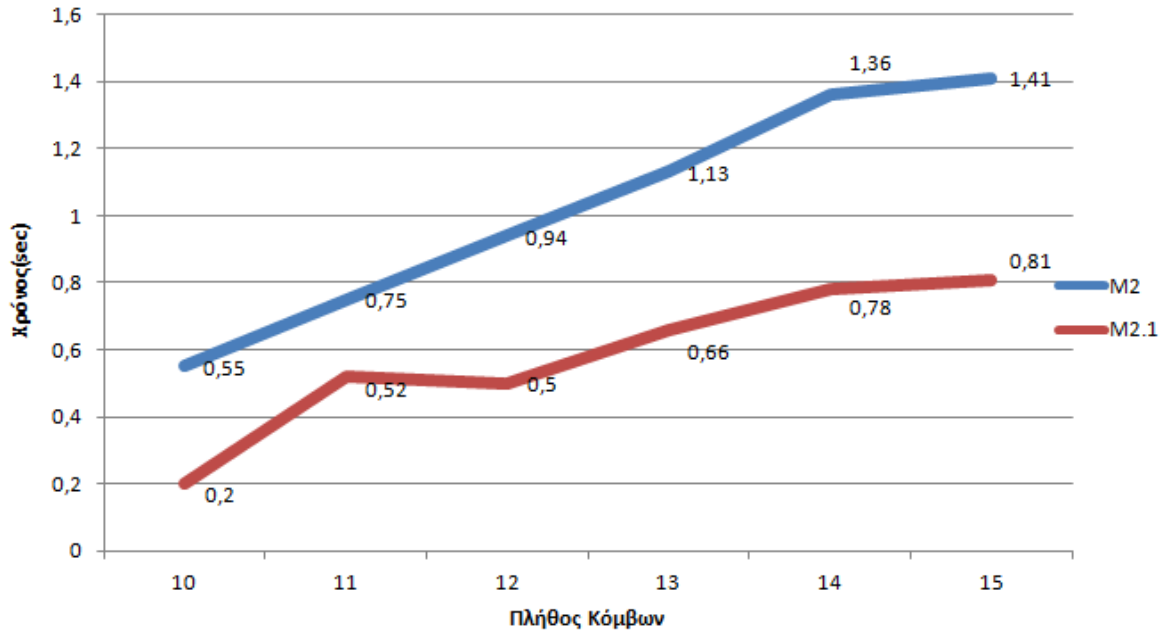
Τα μονέλα M2 και M2.1 όπως και στο παράδειγμα ένα απέδωσαν τη βέλτιστη λύση σε μικρότερο χρόνο αλλά με μικρότερη τιμή αντικειμενικής συνάρτησης με το M1 και M1.1 καθότι δε συμπεριλαμβάνει το κόστος μετακίνησης εντός των πόλεων.

Κόμβοι (N)	Μοντέλο	WTP	ToA	Χρονος Υλοποίησης (sec)	Τιμή Αντ/κής
10	M2.1	2	21	0,2	20
11	M2.1	2	21	0,52	20
12	M2.1	2	21	0,5	20
13	M2.1	2	21	0,66	20
14	M2.1	2	21	0,78	20
15	M2.1	2	21	0,81	20

Πίνακας 5.12 : Χρόνοι υλοποίησης και τιμή αντικειμενικής συνάρτησης μοντέλου M2.1 3^{ου} παραδείγματος



Διάγραμμα 5.5 : Χρόνος υλοποίησης συναρτήση αριθμού κόμβων μοντέλων M1 και M1.1, 3^{ιο} παραδείγματος



Διάγραμμα 5.6 : Χρόνος Υλοποίησης συναρτήση αριθμού κόμβων μοντέλων M2 και M2.1, 3^{ιο} παραδείγματος

Η βέλτιστη διαδρομή των μοντέλω M2 και M2.1 είναι η ακόλουθη :

- Ιωάννινα - Τρίκαλα με ΚΤΕΛ και άφιξη στο PoI
  DT = 9, AT = 11,9
- Τρίκαλα - Λάρισα με ΚΤΕΛ
  DT =19 , AT = 20.4

Το συνολικό κόστος μετακίνησης είναι 20, δηλαδή 2,5 μικρότερο απο τα μοντέλα M1 και M1.1 καθότι δεν υπάρχει το κόστος των μετακινήσεων εντός των πόλεων. Στο παράδειγμα αυτό παρατηρούμε ότι στη διαδρομή Τρίκαλα - Λάρισα επιλέγεται μέσο μεταφοράς λεωφορείο με αποτέλεσμα να βγαίνει φθηνότερο το συνολικό κόστος. Η διαφορά με τα μοντέλα M1 και M1.1 έγκειται στο γεγονός ότι για την μεταφορά εντός των πόλεων έχουν εισαχθεί συγκεκριμένα δρομολόγια, επομένως με τις συγκεκριμένες απαιτήσεις που έχουμε θέσει (*WTP*, *ToA*) τα δύο προαναφερθέντα μοντέλα μας δίνουν διαφορετική δρομολόγηση.

5.4 Συγκριτική Ανάλυση Μοντέλων

Βασικό στοιχείο για τη σύγκριση και τελικά την επιλογή του αποτελεσματικότερου μοντέλου είναι ο υπολογιστικός χρόνος όπως γίνεται εύκολα αντιληπτό. Επομένος ο χρόνος αναμονής για να λάβουμε την βέλτιστη διαδρομή απο τον κώδικα θα αποτελέσει το βασικό μέγεθος σύγκρισης.

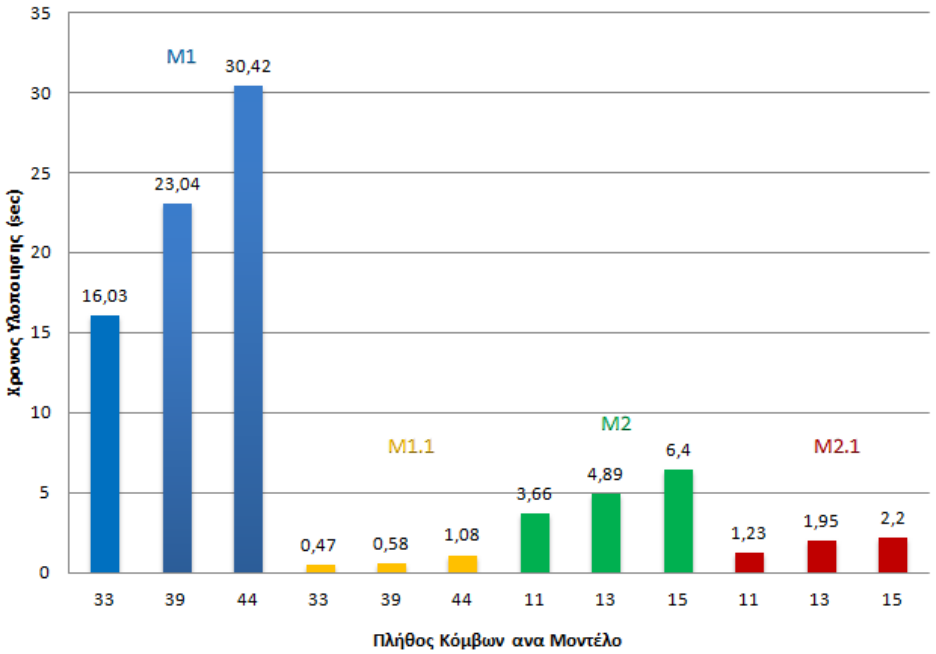
Συγκρίνοντας συνολικά και τα τέσσερα μοντέλα που σχεδιάστηκαν μπορούμε να παρατηρήσουμε ότι οι διαφορές συναντόνται κυρίως στον αριθμό των κόμβων, των μεταβλητών απόφασης και των περιορισμών. Αρχικά τα μοντέλα M1 και M1.1 έχουν ακριβώς ίδιο αριθμό κόμβων (πόλεων) διότι έχουν την ίδια θεώρηση όπως έχει αναφερθεί σε προηγούμενο κεφάλαιο, ωστόσο οι διαφορές στους υπολογιστικούς χρόνους οφείλονται στο διαφορετικό πλήθος μεταβλητών απόφασης και περιορισμών. Με το μεν μοντέλο M1 να έχει εμφανώς μικρότερο αριθμό μεταβλητών απόφασης καθότι εμπεριέχει έναν δείκτη λιγότερο αλλά έναν αρκετά μεγαλύτερο αριθμό περιορισμών. Ομοίως τα μοντέλα M2 και M2.1 εμφανίζουν ακριβώς τα ίδια χαρακτηριστικά στη μεταξύ τους σύγκριση διότι το M2 εμφανίζει και αυτό μικρότερο αριθμό μεταβλητών απόφασης. Σε σύγκριση τώρα των M1 και M2 μεταξύ τους η μεγαλύτερη διαφορά παρουσιάζεται στο πλήθος των κόμβων που στα μεν M1 και M1.1 είναι τριπλάσιο σε σχέση με τα M2 και M2.1 με αποτέλεσμα τα τελευταία να εμφανίζουν μικρότερο αριθμό μεταβλητών απόφασης συγκριτικα με τα πρώτα δύο και επομένως και μικρότερο υπολογιστικό χρόνο.

	M1	M1.1	M2	M2.1
X	i×j	i×j×n	i×j×m	i×j×m×n
Y	i×n	-	i×m×n	-
Z	j×n	-	j×m×n	-
DT	i×j×n	i×j×n	i×j×m×n	i×j×m×n
AT	j×i×n	j×i×n	j×i×m×n	j×i×m×n
U	i	i	i	i

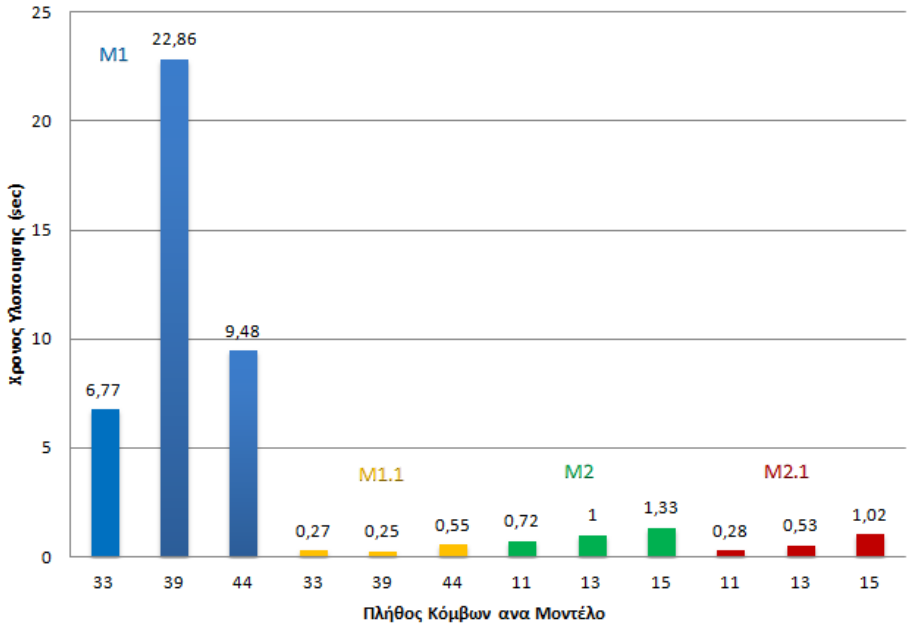
Πίνακας 5.13 : Αριθμός Μεταβλητών Απόφασης για κάθε μοντέλο

Από τα διαγράμματα που παρουσιάστηκαν στην προηγούμενη ενότητα 5.3 για κάθε ένα παραάδειγμα δρομολόγησης παρατηρούμε ότι σε ορισμένες περιπτώσεις τα μοντέλα M1 και M1.1 ότι εμφανίζουν μια απότομη αύξηση του χρόνου υλοποίησης σε κάποιο συγκεκριμένο αριθμό κόμβων ενώ τα μοντέλα M2 και M2.1 έχουν ομαλή αύξηση του χρόνου υλοποίησης.

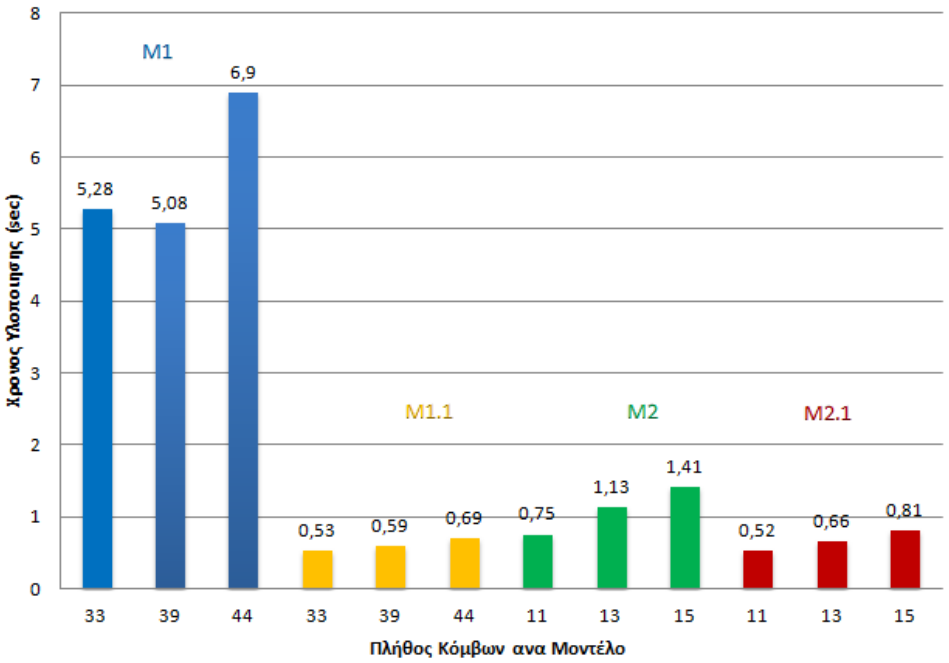
Απο τον πίνακα 5.13 προκύπτει ότι συνολικά τον μεγαλύτερο αριθμό μεταβλητών, δεδομένου των δεικτών για το πρόβλημα μας, τον έχει αρχικά το μοντέλο M1.1 και εν συνεχεία το M1, M2.1 και τέλος το M2. Ωστόσο σημαντικό ρόλο στον υπολογιστικό χρόνο παίζει και ο αριθμός των περιορισμών με τα μοντέλα M1 και M1.1 να υπερέχουν σε αριθμό και εν συνεχεία τα M2 και M2.1. Ωστόσο για να εξαχθούν πιο σωστά συμπεράσματα αρκεί για τα 3 παραδείγματα που προηγήθηκαν να συγκρίνουμε τους υπολογιστικούς χρόνους που παρατηρήθηκαν και για το λόγο αυτό παρουσιάζονται τα παρακάτω συγκριτικά διαγράμματα 5.7, 5.8 και 5.9.



Διάγραμμα 5.7 : Χρόνοι Υλοποίησης 1ου Παραδείγματος



Διάγραμμα 5.8 : Χρόνοι Υλοποίησης 2ου Παραδείγματος



Διάγραμμα 5.9 : Χρόνοι Υλοποίησης 3ου Παραδείγματος

Όπως φαίνεται και απο τα τρία διαγράμματα μπορούμε εύκολα να συμπεράνουμε ότι το μοντέλο με τον μικρότερο υπολογιστικό χρόνο είναι το M1.1 το οποίο έχει μεγαλύτερο αριθμό κόμβων συγκριτικά με τα M2 και M2.1 . Επιπλέον υπερέχει απο τα M2 και M2.1 διότι περιγράφει ρεαλιστικότερα το μοντέλο μας καθώς συμπεριλαμβάνει τους χρόνους αλλά και το κόστος μετακίνησης απο *terminal* σε *terminal* μέσα στην ίδια πόλη με αποτέλεσμα να παρέχει πιο ρεαλιστικά συμπεράσματα.

5.5 **Σύνοψη**

Στο κεφάλαιο αυτό παρουσιάστηκαν τα πειραματικά αποτελέσματα απο την επίλυση των μοντέλων μέσω τριών παραδειγμάτων δρομολόγησης. Παρουσιάστηκε το αποτέλεσμα της αντικειμενικής συνάρτησης και ο υπολογιστικός χρόνος που απαιτήθηκε. Τέλος πραγματοποιήθηκε μια συγκριτική ανάλυση των τεσσάρων μοντέλων με βασικό κριτήριο τον υπολογιστικό χρόνο. Στο επόμενο και τελευταίο κεφάλαιο αναφέρονται τα σσυμπεράσματα συνολικά απο τη συγκεκριμένη διπλωματική εργασία

ΚΕΦΑΛΑΙΟ ΕΚΤΟ:
ΣΥΜΠΕΡΑΣΜΑΤΑ

Ο αντικειμενικός σκοπός της παρούσας διπλωματικής ήταν η εύρεση της οικονομικότερης διαδρομής για έναν ταξιδιώτη που επιθυμεί να μεταβεί από μια πόλη της Ελλάδας σε κάποια άλλη με οποιοδήποτε μέσο μεταφοράς ή ακόμα και με συνδασμό αυτών. Ωστόσο τα μοντέλα που παρουσιάστηκαν είναι δυνατόν με διαφορετικά δεδομένα εισαγωγής κόστους μετακινήσεων, χρόνου μετακινήσεων και δρομολογίων να εφαρμοστούν και να αποδώσουν λύση για οποιοδήποτε δίκτυο πόλεων.

Αρχικά στα δεύτερο κεφάλαιο αναφέρθηκαν στα βασικότερα στοιχεία της θεωρίας γράφων και δικτύων. Στη σύντομη ιστορική αναδρομή παρουσιάστηκαν ορισμένες απο τις βασικότερες μελέτες στην ιστορία των γράφων και δικτύων. Εν συνεχεία αναπτύχθηκε η έννοια του γράφου και δόθηκαν οι βασικοί ορισμοί. Ομοίως παρουσιάστηκαν και οι βασικότερες έννοιες για τα δίκτυα. Όλα τα παραπάνω αποτελούν το βασικό θεωρητικό υπόβραθρό πάνω στο οποίο βασίστηκε η προσομοίωση του προβλήματος που εξετάστηκε καθώς και όλα σχεδόν τα προβλήματα δρομολόγησης καθότι αποτελούν ένα απο τα κυριότερα αντικείμενα μελέτης της επιστημονικής κοινότητας.

Στο θεωρητικό κομμάτι της εργασίας και στο τρίτο κεφάλαιο δόθηκε ένας ορισμός για την δρομολόγηση (*Routing*). Επίσης παρουσιάστηκαν το πρόβλημα του πλανόδιου πωλητή (*TSP*), το πρόβλημα δρομολόγησης οχημάτων (*VRP*) και το πρόβλημα της συντομότερης διαδρομής (*SPP*) που αποτελούν τα κυριότερα προβλήματα δρομολόγησης και το τελευταίο είναι αυτό που αφορά το πρόβλημα του ταξιδιώτη που προσεγγίστηκε.

Στο πειραματικό κομμάτι της διπλωματικής, αρχικά παρουσιάστηκε και αναλύθηκε το πρόβλημα της εύρεσης της βέλτιστης διαδρομής σε ένα δίκτυο συγκεκριμένων πόλεων που επιλέχθηκαν απο την επικράτεια της Ελλάδας. Τα τέσσερα μοντέλα που παρουσιάστηκαν εν συνεχεία βασίστηκαν σε μικτό ακέραιο γραμμικό προγραμματισμό με την αντικειμενική συνάρτηση να ελαχιστοποιεί το συνολικό κόστος μετακίνησης και είναι όμοια για όλα. Επίσης και τα τέσσερα μοντέλα χρησιμοποιούν συνδυασμό δυαδικών και συνεχών μεταβλητών απόφασης. Ειδοποιός διαφορά των μοντέλων αποτελεί η θεώρηση για τους τερματικούς σταθμούς (*terminals*) των μέσων μεταφοράς για κάθε πόλη. Επομένως μπορούμε να τα χωρίσουμε σε δύο κατηγορίες, σε αυτά όπου κάθε κόμβος χαρακτηρίζει ένα *terminal* συγκεκριμένου μέσου μεταφοράς και αφορά τα M1 και M1.1 μοντέλα και σε αυτά όπου κάθε κόμβος χαρακτηρίζει μια πόλη χωρίς να υπάρχει διαφοροποίηση για το κάθε *terminal* και αφορά τα μοντέλα M2 και M2.1 . Με τα την πρώτη θεώρηση και τα M1 και M1.1 όπως γίνεται αντιληπτό να είναι σαφώς πιο ρεαλιστικά. Έπειτα παρουσιάστηκαν και επεξηγήθηκαν οι δείκτες, τα σύνολα, οι παράμετροι, οι μεταβλητές απόφασης, η αντικειμενική συνάρτηση και οι περιορισμοί για κάθε ένα απο αυτά.

Στο πέμπτο κεφάλαιο της εργασίας επιλέχθηκαν τρία τυχαία ρεαλιστικά προβλήματα δρομολόγησης τα οποία και υλοποιήθηκαν σε προγραμματιστικό περιβάλλον και στα τέσσερα μαθηματικά μοντέλα. Για την επίλυση τους, χρησιμοποιήθηκε η γλώσσα προγραμματισμού C++ και το λογισμικό CPLEX. Οι χρόνοι υλοποίησης αποτέλεσαν το βασικό μέτρος σύγκρισης της αποτελεσματικότητας των μοντέλων διότι αφενός μεν η κύρια απαίτηση απο ένα τέτοιο λογισμικό είναι η ταχύτερη απόδοση της βέλτιστης λύσης αφετέρου και τα τέσσερα μοντέλα απέδιδαν την ίδια διαδρομή. Απο την σύγκριση αυτή αποδείχθηκε ότι το M1.1 απέδιδε ταχύτερα την βέλτιστη λύση και επομένως επιλέχθηκε ως το αποτελεσματικότερο διότι προσομοιάζει ρεαλιστικότερα τα προβλήματα δρομολόγησης δεδομένου ότι κάθε κόμβος χαρακτηρίζει ένα *terminal*. Συγκεκριμένα τα M2 και M2.1 ήταν τα αμέσως επόμενα με ταχύτερο χρόνο υλοποίησης και τελευταίο το M1.

Εν κατακλείδι, τα παραπάνω μοντέλα είναι δυνατόν να εμπλουτιστούν περαιτέρω δεδομένου ότι κάθε ταξιδιώτης επιδιώκει να ελαχιστοποιήσει τοσο το κόστος μετακίνησης όσο και τον χρόνο μετακίνησης (επιτυγχάνεται μέχρι κάποιο σημείο) αλλά και το συνολικό αριθμό μετεπιβιβάσεων. Επιπλέον δε λαβμάνονται υπόψιν κριτήρια όπως η πληρότητα των μέσων μεταφοράς, οι τυχόν καθυστερήσεις που στη πράξη μπορεί να μεταβάλλει ολόκληρη τη διαδρομή ακόμα και κάποια μετακίνηση εντός της πόλης που μπορεί να πραγματοποιήσει ο ταξιδιώτης πεζός. Επίσης περαιτέρω δρομολόγια εντός των πόλεων και μεγαλύτερος αριθμός κόμβων(πόλεις) θα ήταν χρήσιμο να προστεθούν έτσι ώστε να καταστεί δυνατόν να χρησιμοποιηθεί απο έναν πραγματικό ταξιδιώτη.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ravindra K. Ahuja, Thomas L. Magnanti, James B. Orlin (1993), "Network Flows - Theory, Algorithms and Applications", *Prentice Hall, Upper Saddle River, New Jersey*

[2] Luigi Moccia, Jean-Francois Cordeau, Gilbert Laporte, Stefan Ropke, Maria Pia Valentini (2008), "Modeling and Solving a Multimodal Routing Problem with Timetables and Time Windows" , *Universite de Montreal, Pavillon Palasis - Prince*

[3] Rahim A. Abbaspour, Farhad Samadzadegan (2011), "Time - dependent personal tour planning and scheduling in metropolises", *Elsevier Ltd*

[4] Seungmo Kang, Yanfeng Ouyang (2010), "The traveling purchaser problem with stochastic prices : Exact and approximate algorithms ", *Elsevier Ltd*

[5] Mikael Andersson, Peter Lindroth (2005), "Route optimization applied to school transports - A method combining column generation with greedy heuristics ", *Department of Mathematics, Chalmers University of Technology, Goteborg University*

[6] Geraldine Heilporn, Luigi De Giovanni, Martine Labbe (2006), "Optimization models for the single delay management problem in public transportation", *European Journal of Operational Research 189, 762-774, Elsevier Ltd*

[7] Tolga Bektas (2004), "The multiple traveling salesman problem : an overview of formulations and solution procedures", *Omega 34, 209-219, Elsevier Ltd*

[8] Donald Davendra (2010), "Travelling Salesman Problem, Theory and Applications", *InTech*

[9] Yosef Sheffi (1985), "Urban Transportation Networks : Equilibrium Analysis with Mathematical Programming Methods", *Prentice - Hall inc, Englewood Cliffs, New Jersey*

[10] O. A. Oduwole (1995), "Multimodal Transport Network Systems Interface, Interaction Coordination: A Specification for Control Systems Integration ", *Transportation Systems Engineering Research Center New Jersey Intitue of Technology, Newark, NJ, U.S.A.*

[11] Aybike Ozdemirel, Burak Gokgur, Deniz Tursel Eliiyi (2012), "An assignment and routing problem with time windows and capacity restriction", *Elsevier Ltd 54 149-158*

[12] Yopo Chan, S. F. Baker (2005), "The Multiple Depot, Multiple Traveling Salesman Facility-Location Problem : Vehicle Range, Service Frequency, and Heuristic Imlementations" , *Elsevier , Mathematical and Computer Modelling 41 1035-1053*

[13] Tao Zhang, W. A. Chaovalitwongse, Yuejie Zhang (2012), "Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick ups and deliveries", *Elsevier, Computer& Operations Research 39 2277-2290*

[14] Paolo Toth, Daniele Vigo (2002), "The Vehicle Routing Problem", *Society for Industrial and Applied Mathematics Philadelphia*

[15] M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh, F. Soumis (1988), "Vehicle Routing with Time Windows : Optimization and Approximation", *Elsevier Science Publishers B.V.*

[16] Tonci Caric, Hrvoje Gold (2008), "Vehicle Routing Problem", *In-Tech, Vienna, Austria*

[17] Reinhard Diestel (2005), "Graph Theory", *Springer - Verlag Heidelberg, New York,* Third *Edition*

[18] J. A. Bondy, U. S. R. Murty (1976), "Graph Theory with Applications", *The Macmillan Press Ltd, Great Britain*

[19] Graph Theory, (2011) , http://www.en.wikipedia.org./wiki/Graph_Theory

[20] http://www.viamechelin.com

[21] http://www.ktelattikis.gr

[22] http://www.ktellarisas.gr

[23] http://www.ktelvolou.gr

[24] http://www.ktel-karditsas.gr

[25] http://www.ktelioannina.gr

[26] http://www.ktelevrou.gr

[27] http://www.ktelpierias.gr

[28] http://ktel-trikala.gr

[29] http://www.ktelachaias.gr

[30] http://www.ktel-thesprotias.gr

[31] http://www.ktelkavalas.gr

[32] http://www.ktelxanthis.gr

[33] http://www.minoan.gr

[34] https://tickets.trainose.gr/dromologia

ΠΑΡΑΡΤΗΜΑ Α
Κώδικας C++, Μοντέλο M1, Πλήθος Κόμβων 30

```
#include <ilcplex/ilocplex.h>
#include <stdio.h>
#define N 30
ILOSTLBEGIN


int i,j,a,k,n,z;
const int imax=30;            // arithmos komvwn
const int jmax=30;
const int kmax=30;
const int nmax=5;            // arithmos dromologiwn
const int zmax=5;
const int WTP=1;
const int ToA=20;
const int O=0;
const int P=3;
const int D=9;
const int amax=1;
const int M=1000000;

int main (int argc, char **argv)
{

//-----------Arxikopoihsh pinaka kostous----------------------------------//

double C[imax][jmax];

for (i=0;i<imax;i++){
    for (j=0;j<jmax;j++){
            C[i][j]=200;
    }
}

//--------------------------Pinakas Kostous-------------------------//

//-----------Kostos metaforas me TRENO apo komvo se komvo----------------------//
C[0][3]=30;          C[0][9]=50;          C[0][12]=31;
C[0][24]=40;         C[3][0]=30;          C[3][6]=5;
C[3][9]=12;          C[3][12]=9;          C[3][24]=12;
C[6][3]=5;           C[9][0]=50;          C[9][3]=12;
```

```
C[9][12]=22;          C[9][21]=23;          C[9][24]=5;
C[12][0]=31;          C[12][3]=9;           C[12][9]=22;
C[12][15]=3;          C[12][24]=13;         C[15][12]=3;
C[21][9]=23;          C[24][0]=40;          C[24][3]=12;
C[24][9]=5;           C[24][12]=13;         C[27][12]=3.5;
C[12][27]=3.5;        C[15][27]=2.5;        C[27][15]=2.5;

//-----------Kostos metaforas me KTEL apo komvo se komvo----------------------//
C[1][4]=25;           C[1][7]=40;           C[1][10]=40;
C[1][13]=25;          C[1][16]=32;          C[1][19]=50;
C[1][22]=75;          C[4][1]=25;           C[4][7]=8;
C[4][10]=20;          C[4][13]=5.5;         C[4][16]=7;
C[4][19]=18;          C[7][1]=40;           C[7][4]=8;
C[7][10]=26;          C[7][13]=6.5;         C[7][19]=25;
C[10][1]=40;          C[10][4]=20;          C[10][7]=26;
C[10][13]=23;         C[10][16]=18;         C[10][19]=28;
C[10][22]=30;         C[13][1]=25;          C[13][4]=5.5;
C[13][7]=6.5;         C[13][10]=23;         C[13][16]=4;
C[16][1]=32;          C[16][4]=7;           C[16][10]=18;
C[16][13]=4;          C[16][19]=20;         C[19][1]=50;
C[19][4]=18;          C[19][7]=25;          C[19][10]=28;
C[19][16]=20;         C[22][1]=75;          C[22][10]=30;
C[1][25]=30;          C[25][1]=30;          C[25][4]=13.5;
C[4][25]=13.5;        C[25][10]=8;          C[10][25]=8;
C[28][1]=28;          C[1][28]=28;          C[28][4]=6;
C[4][28]=6;           C[28][7]=14;          C[7][28]=14;
C[28][10]=20;         C[10][28]=20;         C[28][13]=7;
C[13][28]=7;          C[28][16]=3.5;        C[16][28]=3.5;
C[28][19]=14;         C[19][28]=14;

//-----------Kostos metaforas me I.X. apo komvo se komvo----------------------//
C[2][5]=50;           C[2][8]=60;           C[2][11]=78;
C[2][14]=35;          C[2][17]=42.5;        C[2][20]=79.5;
C[2][23]=117;         C[5][2]=50;           C[5][8]=10;
C[5][11]=27;          C[5][14]=6;           C[5][17]=6.5;
C[5][20]=23;          C[5][23]=65;          C[8][2]=60;
C[8][5]=10;           C[8][11]=35.5;        C[8][14]=8;
C[8][17]=16.5;        C[8][20]=33;          C[8][23]=75;
C[11][2]=78;          C[11][5]=27;          C[11][8]=35.5;
C[11][14]=33;         C[11][17]=30;         C[11][20]=39;
C[11][23]=40;         C[14][2]=40;          C[14][5]=6;
C[14][8]=8;           C[14][11]=33;         C[14][17]=6;
C[14][20]=26;         C[14][23]=70;         C[17][2]=42.5;
C[17][5]=6.5;         C[17][8]=16.5;        C[17][11]=30;
C[17][14]=6;          C[17][20]=25;         C[17][23]=72;
C[20][2]=79.5;        C[20][5]=23;          C[20][8]=33;
```

```
C[20][11]=39;          C[20][14]=26;          C[20][17]=25;
C[20][23]=76;          C[23][2]=117;          C[23][5]=65;
C[23][8]=75;           C[23][11]=40;          C[23][14]=70;
C[23][17]=72;          C[23][20]=76;          C[26][2]=68;
C[2][26]=68;           C[26][5]=17;           C[5][26]=17;
C[26][8]=25.5;         C[8][26]=25.5;         C[26][11]=11;
C[11][26]=11;          C[26][14]=22.5;        C[14][26]=22.5;
C[26][17]=23;          C[17][26]=23;          C[26][20]=36.5;
C[20][26]=36.5;        C[26][23]=49;          C[23][26]=49;
C[29][2]=46;           C[2][29]=46;           C[29][5]=7;
C[5][29]=7;            C[29][8]=17.2;         C[8][29]=17.2;
C[29][11]=34;          C[11][29]=34;          C[29][14]=9.8;
C[14][29]=9.8;         C[29][17]=4;           C[17][29]=4;
C[29][20]=16.6;        C[20][29]=16.6;        C[29][23]=72.5;
C[23][29]=72.5;        C[29][26]=23;          C[26][29]=23;

//-----------Kostos metaforas apo terminal se terminal sthn idia poli ----------//
C[0][1]=1.4;           //---Athina---//
C[1][0]=1.4;
C[0][2]=1.4;
C[2][0]=1.4;
C[1][2]=1.4;
C[2][1]=1.4;

C[3][4]=1;             //---Larisa---//
C[4][3]=1;
C[4][5]=1;
C[5][4]=1;
C[3][5]=1;
C[5][3]=1;

C[6][8]=0.8;           //---Volos---//
C[8][6]=0.8;
C[7][8]=0.8;
C[8][7]=0.8;
C[6][8]=0.8;
C[8][6]=0.8;

C[9][11]=1;            //---Thessaloniki---//
C[11][9]=1;
C[10][11]=1;
C[11][10]=1;
C[9][10]=1;
C[10][9]=1;

C[12][14]=0.6;         //---Farsala---//
```

```
C[14][12]=0.6;
C[13][14]=0.6;
C[14][13]=0.6;
C[13][12]=0.6;
C[12][13]=0.6;

C[15][17]=0.6;              //---Karditsa--//
C[17][15]=0.6;
C[15][16]=0.6;
C[16][15]=0.6;
C[16][17]=0.6;
C[17][16]=0.6;

C[19][20]=0.7;              //---Iwannina---//
C[20][19]=0.7;

C[21][22]=1;              //---Aleksandroupoli--//
C[22][21]=1;
C[21][23]=1;
C[23][21]=1;
C[23][22]=1;
C[22][23]=1;

C[24][25]=1;              //---Katerini--//
C[25][24]=1;
C[24][26]=1;
C[26][24]=1;
C[25][26]=1;
C[26][25]=1;

C[27][28]=0.8;              //---Trikala--//
C[28][27]=0.8;
C[27][29]=0.8;
C[29][27]=0.8;
C[28][29]=0.8;
C[29][28]=0.8;
//-------------------------------------------------------------------------//

//------------------Arxikopoihsh pinaka xronou----------------------------------//
double t[imax][jmax];

for (i=0;i<imax;i++){
    for (j=0;j<jmax;j++){
            t[i][j]=200;
    }
}
```

```
//-------------------Pinakas Xronou---------------------------------------------//

//-----------Xronos metaforas me TRENO apo komvo se komvo----------------------//
t[0][3]=4;          t[0][9]=5.5;        t[0][12]=3.65;
t[0][24]=4.5;       t[3][0]=4;          t[3][6]=0.8;
t[3][9]=1.5;        t[3][12]=0.4;       t[3][24]=0.8;
t[6][3]=0.8;        t[9][0]=5.5;        t[9][3]=1.5;
t[9][12]=1.65;      t[9][21]=5.8;       t[9][24]=0.8;
t[12][0]=3.65;      t[12][3]=0.4;       t[12][9]=1.65;
t[12][15]=0.3;      t[12][24]=1;        t[15][12]=0.3;
t[21][9]=5.8;       t[24][0]=4.5;       t[24][3]=0.8;
t[24][9]=0.8;       t[24][12]=1;        t[27][12]=0.7;
t[12][27]=0.7;      t[15][27]=0.4;      t[27][15]=0.4;

//-----------Xronos metaforas me KTEL apo komvo se komvo----------------------//
t[1][4]=3.7;        t[1][7]=3.8;        t[1][10]=6;
t[1][13]=4;         t[1][16]=5;         t[1][19]=8;
t[1][22]=10;        t[4][1]=3.7;        t[4][7]=1;
t[4][10]=2.5;       t[4][13]=1.4;       t[4][16]=1.4;
t[4][19]=4.3;       t[7][1]=3.8;        t[7][4]=1;
t[7][10]=2.9;       t[7][13]=1.6;       t[7][19]=4.8;
t[10][1]=6;         t[10][4]=2.5;       t[10][7]=2.9;
t[10][13]=3.5;      t[10][16]=3.5;      t[10][19]=4;
t[10][22]=4;        t[13][1]=4;         t[13][4]=1.4;
t[13][7]=1.6;       t[13][10]=3.5;      t[13][16]=1;
t[16][1]=5;         t[16][4]=1.4;       t[16][10]=3.5;
t[16][13]=1;        t[16][19]=3.8;      t[19][1]=8;
t[19][4]=4.3;       t[19][7]=4.8;       t[19][10]=4;
t[19][16]=3.8;      t[22][1]=10;        t[22][10]=4;
t[25][1]=6;         t[1][25]=6;         t[25][4]=1.8;
t[4][25]=1.8;       t[25][10]=1.4;      t[10][25]=1.4;
t[28][1]=5.5;       t[7][28]=2.5;       t[28][10]=3.9;
t[10][28]=3.9;      t[28][13]=2;        t[13][28]=2;
t[28][16]=1;        t[16][28]=1;        t[28][19]=2.9;
t[19][28]=2.9;

//-----------Xronos metaforas me I.X. apo komvo se komvo----------------------//
t[2][5]=3.5;        t[2][8]=3.4;        t[2][11]=5.7;
t[2][14]=3.5;       t[2][17]=4.25;      t[2][20]=6.5;
t[2][23]=8.5;       t[5][2]=3.5;        t[5][8]=0.9;
t[5][11]=2;         t[5][14]=1.1;       t[5][17]=1;
t[5][20]=3.5;       t[5][23]=5;         t[8][2]=3.4;
t[8][5]=0.9;        t[8][11]=2.6;       t[8][14]=1.3;
t[8][17]=2.3;       t[8][20]=4;         t[8][23]=5.5;
```

```
t[11][2]=5.7;          t[11][5]=2;          t[11][8]=2.6;
t[11][14]=3;           t[11][17]=3;         t[11][20]=3;
t[11][23]=3;           t[14][2]=3.5;        t[14][5]=1.1;
t[14][8]=1.3;          t[14][11]=3;         t[14][17]=0.8;
t[14][20]=4;           t[14][23]=6;         t[17][2]=4.25;
t[17][5]=1;            t[17][8]=2.3;        t[17][11]=3;
t[17][14]=0.8;         t[17][20]=3;         t[17][23]=6;
t[20][2]=6.5;          t[20][5]=3.5;        t[20][8]=4;
t[20][11]=3;           t[20][14]=4;         t[20][17]=3;
t[20][23]=6;           t[23][2]=8.5;        t[23][5]=5;
t[23][8]=5.5;          t[23][11]=3;         t[23][14]=6;
t[23][17]=6;           t[23][20]=6;         t[26][2]=5;
t[2][26]=5;            t[26][5]=1.3;        t[5][26]=1.3;
t[26][8]=1.9;          t[8][26]=1.9;        t[26][11]=1;
t[11][26]=1;           t[26][14]=2;         t[14][26]=2;
t[26][17]=2.5;         t[17][26]=2.5;       t[26][20]=3;
t[20][26]=3;           t[26][23]=3.9;       t[23][26]=3.9;
t[29][2]=4.9;          t[2][29]=4.9;        t[29][5]=1;
t[5][29]=1;            t[29][8]=2;          t[8][29]=2;
t[29][11]=3;           t[11][29]=3;         t[29][14]=1.6;
t[14][29]=1.6;         t[29][17]=0.7;       t[17][29]=0.7;
t[29][20]=2;           t[20][29]=2;         t[29][23]=6;
t[23][29]=6;           t[29][26]=2.5;       t[26][29]=2.5;

//-----------Xronos metaforas apo terminal se terminal sthn idia poli --------//
t[0][1]=1;             //---Athina---//
t[1][0]=1;
t[0][2]=1;
t[2][0]=1;
t[1][2]=1;
t[2][1]=1;

t[3][4]=0.4;           //---Larisa---//
t[4][3]=0.4;
t[4][5]=0.4;
t[5][4]=0.4;
t[3][5]=0.4;
t[5][3]=0.4;

t[6][8]=0.3;           //---Volos---//
t[8][6]=0.3;
t[7][8]=0.3;
t[8][7]=0.3;
t[6][8]=0.3;
t[8][6]=0.3;
```

```
t[9][11]=0.5;              //----Thessaloniki---//
t[11][9]=0.5;
t[10][11]=0.5;
t[11][10]=0.5;
t[9][10]=0.5;
t[10][9]=0.5;

t[12][14]=0.2;           //---Farsala---//
t[14][12]=0.2;
t[13][14]=0.2;
t[14][13]=0.2;
t[13][12]=0.2;
t[12][13]=0.2;

t[15][17]=0.3;           //---Karditsa---//
t[17][15]=0.3;
t[15][16]=0.3;
t[16][15]=0.3;
t[16][17]=0.3;
t[17][16]=0.3;

t[19][20]=0.3;           //---Iwannina---//
t[20][19]=0.3;

t[15][17]=0.8;          //---Aleksandroupoli---//
t[17][15]=0.8;
t[15][16]=0.8;
t[16][15]=0.8;
t[16][17]=0.8;
t[17][16]=0.8;

t[24][25]=0.4;          //---Katerini---//
t[25][24]=0.4;
t[24][26]=0.4;
t[26][24]=0.4;
t[25][26]=0.4;
t[26][25]=0.4;

t[27][28]=0.4;            //---Trikala--//
t[28][27]=0.4;
t[27][29]=0.4;
t[29][27]=0.4;
t[28][29]=0.4;
t[29][28]=0.4;
//-------------------------------------------------------------------------//
```

```
//---------------------Orismos Pinaka Dromologiwn--------------------------------------//
double d[imax][jmax][nmax];

for (i=0;i<imax;i++){
     for (j=0;j<jmax;j++){
      for (n=0;n<nmax;n++){
                d[i][j][n]=24;
         }
   }
}
//-------------------Dromologia trenwn-----------------------------------------------//
d[0][3][0]=7.3;              //--Athina--Larisa--//
d[0][3][1]=10.3;
d[0][3][2]=14.3;
d[0][3][3]=16.3;
d[0][3][4]=18.3;

d[3][0][0]=7.5;             //--Larisa--Athina--//
d[3][0][1]=11.5;
d[3][0][2]=13.5;
d[3][0][3]=16.5;
d[3][0][4]=19.5;
//-----------------//
d[3][6][0]=8.5;             //--Larisa--Volos--//
d[3][6][1]=11.5;
d[3][6][2]=14.5;
d[3][6][3]=18.5;
d[3][6][4]=20.5;

d[6][3][0]=7;               //--Volos--Larisa--//
d[6][3][1]=11.5;
d[6][3][2]=15.5;
d[6][3][3]=17.5;
d[6][3][4]=19.5;
//-----------------//

d[0][9][0]=7.3;             //--Athina--Thessaloniki--//
d[0][9][1]=11.3;
d[0][9][2]=14.3;
d[0][9][3]=16.3;
d[0][9][4]=18.3;

d[9][0][0]=7;              //--Thessaloniki--Athina--//
d[9][0][1]=11;
d[9][0][2]=14;
d[9][0][3]=17;
```

```
d[9][0][4]=19;
//-----------------//
d[9][3][0]=7;          //--Thessaloniki--Larisa--//
d[9][3][1]=10;
d[9][3][2]=15;
d[9][3][3]=17;
d[9][3][4]=19;

d[3][9][0]=11.3;       //--Larisa--Thessaloniki--//
d[3][9][1]=14.3;
d[3][9][2]=16.3;
d[3][9][3]=18.3;
d[3][9][4]=20.3;
//-----------------//
d[0][12][0]=7.3;         //--Athina-Farsala--//
d[0][12][1]=10.3;
d[0][12][2]=12.3;
d[0][12][3]=14.3;
d[0][12][4]=16.3;

d[12][0][0]=7.9;         //--Farsala--Athina--//
d[12][0][1]=8.9;
d[12][0][2]=11.9;
d[12][0][3]=13.9;
d[12][0][4]=16.9;
//-----------------//
d[12][3][0]=10.95;     //--Farsala--Larisa--//
d[12][3][1]=13.95;
d[12][3][2]=15.95;
d[12][3][3]=17.95;
d[12][3][4]=19.95;

d[3][12][0]=7.5;       //--Larisa--Farsala--//
d[3][12][1]=11.5;
d[3][12][2]=13.5;
d[3][12][3]=16.5;
d[3][12][4]=19.5;
//-----------------//
d[12][9][0]=10.95;     //--Farsala--Thessaloniki--//
d[12][9][1]=13.95;
d[12][9][2]=15.95;
d[12][9][3]=17.95;
d[12][9][4]=19.95;

d[9][12][0]=7;         //--Thessaloniki--Farsala--//
d[9][12][1]=10;
```

```
d[9][12][2]=13;
d[9][12][3]=16;
d[9][12][4]=19;
//-----------------//
d[12][15][0]=11;       //--Farsala--Karditsa--//
d[12][15][1]=14;
d[12][15][2]=16;
d[12][15][3]=18;
d[12][15][4]=20;

d[9][12][0]=7;         //--Karditsa--Farsala--//
d[9][12][1]=11;
d[9][12][2]=13;
d[9][12][3]=16;
d[9][12][4]=18;
//-----------------//
d[21][9][0]=6;       //--Aleksandroupoli--Thessaloniki--//
d[21][9][1]=15;
d[21][9][2]=24;
d[21][9][3]=24;
d[21][9][4]=24;

d[9][21][0]=7;         //--Thessaloniki--Aleksandroupoli--//
d[9][21][1]=16;
d[9][21][2]=24;
d[9][21][3]=24;
d[9][21][4]=24;
//-----------------//
d[24][0][0]=6;          //--Katerini--Athina--//
d[24][0][1]=10;
d[24][0][2]=13;
d[24][0][3]=16;
d[24][0][4]=19;

d[0][24][0]=7;         //--Athina--Katerini--//
d[0][24][1]=10;
d[0][24][2]=13;
d[0][24][3]=17;
d[0][24][4]=19;
//-----------------//
d[24][3][0]=6;          //--Katerini--Larisa--//
d[24][3][1]=11;
d[24][3][2]=14;
d[24][3][3]=17;
d[24][3][4]=19;
```

```
d[3][24][0]=7.5;        //--Larisa--Katerini--//
d[3][24][1]=10;
d[3][24][2]=14.5;
d[3][24][3]=16.5;
d[3][24][5]=18.5;
//------------------//
d[24][9][0]=7;          //--Katerini--Thessaloniki--//
d[24][9][1]=12;
d[24][9][2]=15;
d[24][9][3]=17;
d[24][9][4]=19;

d[9][24][0]=7;         //--Thessaloniki--Katerini--//
d[9][24][1]=12;
d[9][24][2]=14.5;
d[9][24][3]=16.5;
d[9][24][4]=18;
//------------------//
d[24][12][0]=6;         //--Katerini--Farsala--//
d[24][12][1]=11;
d[24][12][2]=13;
d[24][12][3]=16;
d[24][12][4]=19;

d[12][24][0]=9;         //--Farsala--Katerini--//
d[12][24][1]=11;
d[12][24][2]=16;
d[12][24][3]=18;
d[12][24][4]=20;
//------------------//
d[27][12][0]=6;         //--Trikala--Farsala--//
d[27][12][1]=8.5;
d[27][12][2]=14;
d[27][12][3]=18;
d[27][12][4]=20;

d[12][27][0]=6;         //--Farsala--Trikala--//
d[12][27][1]=8;
d[12][27][2]=12.5;
d[12][27][3]=18.5;
d[12][27][4]=20.5;
//------------------//
d[27][15][0]=6;         //--Trikala--Karditsa--//
d[27][15][1]=8.5;
d[27][15][2]=18;
d[27][15][3]=20;
```

```
d[27][15][4]=22;

d[15][27][0]=6;        //--Karditsa--Trikala--//
d[15][27][1]=7.5;
d[15][27][2]=13;
d[15][27][3]=19;
d[15][27][4]=21;
//-------------------------------------------------------------//

//-----------------Dromologia KTEL-----------------------------------------------//
d[1][4][0]=7;                //--Athina--Larisa--//
d[1][4][1]=12.5;
d[1][4][2]=14;
d[1][4][3]=16.5;
d[1][4][4]=19;

d[1][7][0]=7.5;              //--Athina--Volos--//
d[1][7][1]=12;
d[1][7][2]=14;
d[1][7][3]=17;
d[1][7][4]=20;

d[4][1][0]=7;                //--Larisa--Athina--//
d[4][1][1]=11;
d[4][1][2]=14;
d[4][1][3]=16;
d[4][1][4]=20;
//----------------//
d[4][7][0]=8;                //--Larisa--Volos--//
d[4][7][1]=13;
d[4][7][2]=16;
d[4][7][3]=18;
d[4][7][4]=20;

d[7][4][0]=7;              //--Volos--Larisa--//
d[7][4][1]=12;
d[7][4][2]=14;
d[7][4][3]=17;
d[7][4][4]=20;

d[7][1][0]=7.5;            //--Volos--Athina--//
d[7][1][1]=12;
d[7][1][2]=14.5;
d[7][1][3]=17;
d[7][1][4]=19.5;
```

```
d[7][10][0]=7;              //--Volos--Thessaloniki--//
d[7][10][1]=11.5;
d[7][10][2]=14;
d[7][10][3]=17;
d[7][10][4]=20;

d[7][13][0]=8;              //--Volos--Farsala--//
d[7][13][1]=12.5;
d[7][13][2]=14;
d[7][13][3]=16;
d[7][13][4]=18;
//----------------//
d[1][10][0]=7;              //--Athina--Thessaloniki--//
d[1][10][1]=13;
d[1][10][2]=15;
d[1][10][3]=17;
d[1][10][4]=19;

d[10][1][0]=7;              //--Thessaloniki--Athina--//
d[10][1][1]=10;
d[10][1][2]=13;
d[10][1][3]=15;
d[10][1][4]=18;

d[10][7][0]=7.5;            //--Thessaloniki--Volos--//
d[10][7][1]=11;
d[10][7][2]=14.5;
d[10][7][3]=16;
d[10][7][4]=18;
//----------------//
d[10][4][0]=8;            //--Thessaloniki--Larisa--//
d[10][4][1]=12;
d[10][4][2]=14;
d[10][4][3]=17;
d[10][4][4]=19;

d[4][10][0]=8;            //--Larisa--Thessaloniki--//
d[4][10][1]=12.5;
d[4][10][2]=14.5;
d[4][10][3]=17;
d[4][10][4]=19.5;
//----------------//
d[1][13][0]=7;          //--Athina--Farsala--//
d[1][13][1]=12.5;
d[1][13][2]=14;
d[1][13][3]=16;
```

```
d[1][13][4]=18.5;

d[13][1][0]=7;          //--Farsala--Athina--//
d[13][1][1]=11;
d[13][1][2]=14;
d[13][1][3]=16;
d[13][1][4]=18;
//-----------------//
d[13][4][0]=8;          //--Farsala--Larisa--//
d[13][4][1]=12.5;
d[13][4][2]=14;
d[13][4][3]=17.5;
d[13][4][4]=18.5;

d[4][13][0]=7;          //--Larisa--Farsala--//
d[4][13][1]=12.5;
d[4][13][2]=14;
d[4][13][3]=16.5;
d[4][13][4]=19;

d[13][7][0]=8;          //--Farsala--Volos--//
d[13][7][1]=12.5;
d[13][7][2]=14;
d[13][7][3]=16.5;
d[13][7][4]=18;
//-----------------//
d[13][10][0]=8;         //--Farsala--Thessaloniki--//
d[13][10][1]=12.5;
d[13][10][2]=14.5;
d[13][10][3]=17.5;
d[13][10][4]=19;

d[10][13][0]=8;         //--Thessaloniki--Farsala--//
d[10][13][1]=12.5;
d[10][13][2]=15;
d[10][13][3]=17;
d[10][13][4]=19;
//-----------------//
d[13][16][0]=8;       //--Farsala--Karditsa--//
d[13][16][1]=12;
d[13][16][2]=14;
d[13][16][3]=16;
d[13][16][4]=18;

d[16][13][0]=7;         //--Karditsa--Farsala--//
d[16][13][1]=11;
```

```
d[16][13][2]=14;
d[16][13][3]=16;
d[16][13][4]=19;

d[16][1][0]=7;        //--Karditsa--Athina--//
d[16][1][1]=12;
d[16][1][2]=15;
d[16][1][3]=17.5;
d[16][1][4]=19;

d[16][10][0]=7;        //--Karditsa--Thessaloniki--//
d[16][10][1]=11.5;
d[16][10][2]=14.5;
d[16][10][3]=17;
d[16][10][4]=20;

d[16][4][0]=8;        //--Karditsa--Larisa--//
d[16][4][1]=11;
d[16][4][2]=14;
d[16][4][3]=16;
d[16][4][4]=18;

d[4][16][0]=7;        //--Larisa--Karditsa--//
d[4][16][1]=11;
d[4][16][2]=14;
d[4][16][3]=16.5;
d[4][16][4]=18.5;
//---------------//
d[19][1][0]=8;        //--Iwannina--Athina--//
d[19][1][1]=11;
d[19][1][2]=13;
d[19][1][3]=15.5;
d[19][1][4]=17.5;

d[1][19][0]=6.5;        //--Athina--Iwannina--//
d[1][19][1]=11;
d[1][19][2]=13;
d[1][19][3]=15.5;
d[1][19][4]=17;

d[19][4][0]=7;        //--Iwannina--Larisa--//
d[19][4][1]=12;
d[19][4][2]=14.5;
d[19][4][3]=18.5;
d[19][4][4]=19;
```

```
d[4][19][0]=7;          //--Larisa--Iwannina--//
d[4][19][1]=12;
d[4][19][2]=15;
d[4][19][3]=18;
d[4][19][4]=20;

d[19][7][0]=7;          //--Iwannina--Volos--//
d[19][7][1]=15;
d[19][7][2]=18;
d[19][7][3]=20;
d[19][7][4]=22;

d[7][19][0]=8;          //--Volos--Iwannina--//
d[7][19][1]=13.5;
d[7][19][2]=17.5;
d[7][19][3]=20;
d[7][19][4]=22;

d[19][10][0]=7;          //--Iwannina--Thessaloniki--//
d[19][10][1]=14;
d[19][10][2]=16;
d[19][10][3]=18.5;
d[19][10][4]=20.5;

d[10][19][0]=8;          //--Thessaloniki--Iwannina--//
d[10][19][1]=12;
d[10][19][2]=15.5;
d[10][19][3]=18.5;
d[10][19][4]=20.5;

d[19][16][0]=8;          //--Iwannina--Karditsa--//
d[19][16][1]=11;
d[19][16][2]=14;
d[19][16][3]=18;
d[19][16][4]=20;

d[16][19][0]=8;          //--Karditsa--Iwannina--//
d[16][19][1]=11.5;
d[16][19][2]=14;
d[16][19][3]=18;
d[16][19][4]=20;

d[22][10][0]=6;          //--Aleksandroupoli--Thessaloniki--//
d[22][10][1]=10;
d[22][10][2]=11;
d[22][10][3]=14;
```

```
d[22][10][4]=16.5;

d[10][22][0]=8;          //--Thessaloniki--Aleksandroupoli--//
d[10][22][1]=12.5;
d[10][22][2]=14.5;
d[10][22][3]=16.5;
d[10][22][4]=19;
//-----------------------Katerini-----------------------------------------------------//

d[25][1][0]=9.5;          //--Katerini--Athina--//
d[25][1][1]=15;
d[25][1][2]=24;
d[25][1][3]=24;
d[25][1][4]=24;

d[1][25][0]=9.5;          //--Athina--Katerini--//
d[1][25][1]=15;
d[1][25][2]=17;
d[1][25][3]=22;
d[1][25][4]=24;
//---------------//
d[25][4][0]=9.5;          //--Katerini--Larisa--//
d[25][4][1]=15;
d[25][4][2]=24;
d[25][4][3]=24;
d[25][4][4]=24;

d[4][25][0]=12.5;          //--Larisa--Katerini--//
d[4][25][1]=18;
d[4][25][2]=20;
d[4][25][3]=22;
d[4][25][4]=24;
//---------------//
d[25][4][0]=6.5;          //--Katerini--Thessaloniki--//
d[25][4][1]=8;
d[25][4][2]=10.5;
d[25][4][3]=12;
d[25][4][4]=24;

d[4][25][0]=9.5;          //--Thessaloniki--Katerini--//
d[4][25][1]=15;
d[4][25][2]=17;
d[4][25][3]=22;
d[4][25][4]=24;
//--------------------Trikala-----------------------------------------------------//
d[28][1][0]=7;          //--Trikala--Athina--//
```

```
d[28][1][1]=11;
d[28][1][2]=15;
d[28][1][3]=17.5;
d[28][1][4]=18.5;

d[1][28][0]=7.5;          //--Athina--Trikala--//
d[1][28][1]=11.5;
d[1][28][2]=15.5;
d[1][28][3]=17;
d[1][28][4]=18;
//---------------//
d[28][4][0]=6.5;          //--Trikala--Larisa--//
d[28][4][1]=10;
d[28][4][2]=15.5;
d[28][4][3]=17;
d[28][4][4]=19;

d[4][28][0]=6.5;          //--Larisa--Trikala--//
d[4][28][1]=10.5;
d[4][28][2]=14.5;
d[4][28][3]=17;
d[4][28][4]=19.5;
//---------------//
d[28][7][0]=7;          //--Trikala--Volos--//
d[28][7][1]=11.5;
d[28][7][2]=15;
d[28][7][3]=19;
d[28][7][4]=24;

d[7][28][0]=6.5;          //--Volos--Trikala--//
d[7][28][1]=11;
d[7][28][2]=15;
d[7][28][3]=19;
d[7][28][4]=24;
//---------------//
d[28][10][0]=7.5;          //--Trikala--Thessaloniki--//
d[28][10][1]=10.5;
d[28][10][2]=13.5;
d[28][10][3]=16;
d[28][10][4]=20;

d[10][28][0]=8.5;          //--Thessaloniki--Trikala--//
d[10][28][1]=12;
d[10][28][2]=15;
d[10][28][3]=17;
d[10][28][4]=18.5;
```

```
//---------------//
d[28][13][0]=7;          //--Trikala--Farsala--//
d[28][13][1]=11.5;
d[28][13][2]=15;
d[28][13][3]=19;
d[28][13][4]=24;

d[13][28][0]=6.5;          //--Farsala--Trikala--//
d[13][28][1]=11;
d[13][28][2]=15;
d[13][28][3]=19;
d[13][28][4]=24;
//---------------//
d[28][16][0]=8;          //--Trikala--Karditsa--//
d[28][16][1]=11;
d[28][16][2]=14;
d[28][16][3]=17;
d[28][16][4]=19;

d[16][28][0]=6.5;          //--Karditsa--Trikala--//
d[16][28][1]=11;
d[16][28][2]=15;
d[16][28][3]=17;
d[16][28][4]=19;
//---------------//
d[28][19][0]=8.5;          //--Trikala--Iwannina--//
d[28][19][1]=15;
d[28][19][2]=15.5;
d[28][19][3]=24;
d[28][19][4]=24;

d[19][28][0]=9;          //--Iwannina--Trikala--//
d[19][28][1]=15;
d[19][28][2]=24;
d[19][28][3]=24;
d[19][28][4]=24;

//------------------"Dromologia" oxhmatos---------------------------------------//

//-----------------------------Athina-----------------------------------------//
d[2][5][0]=7;            //--Athina--Larisa--//
d[2][5][1]=10;
d[2][5][2]=12;
d[2][5][3]=14;
d[2][5][4]=18;
```

```
d[2][8][0]=7;            //--Athina--Volos--//
d[2][8][1]=10;
d[2][8][2]=12;
d[2][8][3]=14;
d[2][8][4]=18;

d[2][11][0]=7;            //--Athina--Thessaloniki--//
d[2][11][1]=10;
d[2][11][2]=12;
d[2][11][3]=14;
d[2][11][4]=18;

d[2][14][0]=7;            //--Athina--Farsala--//
d[2][14][1]=10;
d[2][14][2]=12;
d[2][14][3]=14;
d[2][14][4]=18;

d[2][17][0]=7;            //--Athina--Karditsa--//
d[2][17][1]=10;
d[2][17][2]=12;
d[2][17][3]=14;
d[2][17][4]=18;
//-------------------------------------------------------------------------//
d[5][2][0]=7;            //--Larisa--Athina--//
d[5][2][1]=10;
d[5][2][2]=12;
d[5][2][3]=14;
d[5][2][4]=18;


d[5][8][0]=7;            //--Larisa--Volos--//
d[5][8][1]=10;
d[5][8][2]=12;
d[5][8][3]=14;
d[5][8][4]=18;

d[5][11][0]=7;            //--Larisa--Thessaloniki--//
d[5][11][1]=10;
d[5][11][2]=12;
d[5][11][3]=14;
d[5][11][4]=18;

d[5][14][0]=7;            //--Larisa--Farsala--//
d[5][14][1]=10;
```

```
d[5][14][2]=12;
d[5][14][3]=14;
d[5][14][4]=18;

d[5][17][0]=7;            //--Larisa--Karditsa--//
d[5][17][1]=10;
d[5][17][2]=12;
d[5][17][3]=14;
d[5][17][4]=18;
//-------------------------------------------------------------------------//
d[8][2][0]=7;            //--Volos--Athina--//
d[8][2][1]=10;
d[8][2][2]=12;
d[8][2][3]=14;
d[8][2][4]=18;


d[8][5][0]=7;            //--Volos--Larisa--//
d[8][5][1]=10;
d[8][5][2]=12;
d[8][5][3]=14;
d[8][5][4]=18;

d[8][11][0]=7;            //--Volos--Thessaloniki--//
d[8][11][1]=10;
d[8][11][2]=12;
d[8][11][3]=14;
d[8][11][4]=18;


d[8][14][0]=7;            //--Volos--Farsala--//
d[8][14][1]=10;
d[8][14][2]=12;
d[8][14][3]=14;
d[8][14][4]=18;

d[8][17][0]=7;            //--Volos--Karditsa--//
d[8][17][1]=10;
d[8][17][2]=12;
d[8][17][3]=14;
d[8][17][4]=18;
//-------------------------------------------------------------------------//
d[11][2][0]=7;            //--Thesssaloniki--Athina--//
d[11][2][1]=10;
d[11][2][2]=12;
d[11][2][3]=14;
```

```
d[11][2][4]=18;

d[11][5][0]=7;          //--Thesssaloniki--Larisa--//
d[11][5][1]=10;
d[11][5][2]=12;
d[11][5][3]=14;
d[11][5][4]=18;

d[11][8][0]=7;          //--Thesssaloniki--Volos--//
d[11][8][1]=10;
d[11][8][2]=12;
d[11][8][3]=14;
d[11][8][4]=18;

d[11][14][0]=7;          //--Thesssaloniki--Farsala--//
d[11][14][1]=10;
d[11][14][2]=12;
d[11][14][3]=14;
d[11][14][4]=18;


d[11][17][0]=7;          //--Thesssaloniki--Karditsa--//
d[11][17][1]=10;
d[11][17][2]=12;
d[11][17][3]=14;
d[11][17][4]=18;
//-------------------------------------------------------------------------//
d[14][2][0]=7;          //--Farsala--Athina--//
d[14][2][1]=10;
d[14][2][2]=12;
d[14][2][3]=14;
d[14][2][4]=18;


d[14][5][0]=7;          //--Farsala--Larisa--//
d[14][5][1]=10;
d[14][5][2]=12;
d[14][5][3]=14;
d[14][5][4]=18;


d[14][8][0]=7;          //--Farsala--Volos--//
d[14][8][1]=10;
d[14][8][2]=12;
d[14][8][3]=14;
d[14][8][4]=18;
```

```
d[14][11][0]=7;            //--Farsala--Thessaloniki--//
d[14][11][1]=10;
d[14][11][2]=12;
d[14][11][3]=14;
d[14][11][4]=18;

d[14][17][0]=7;            //--Farsala--Karditsa--//
d[14][17][1]=10;
d[14][17][2]=12;
d[14][17][3]=14;
d[14][17][4]=18;
//-------------------------------------------------------------------------//
d[17][2][0]=7;             //--Karditsa--Athina--//
d[17][2][1]=10;
d[17][2][2]=12;
d[17][2][3]=14;
d[17][2][4]=18;


d[17][5][0]=7;             //--Karditsa--Larisa--//
d[17][5][1]=10;
d[17][5][2]=12;
d[17][5][3]=14;
d[17][5][4]=18;


d[17][8][0]=7;             //--Karditsa--Volos--//
d[17][8][1]=10;
d[17][8][2]=12;
d[17][8][3]=14;
d[17][8][4]=18;

d[17][11][0]=7;            //--Karditsa--Thessaloniki--//
d[17][11][1]=10;
d[17][11][2]=12;
d[17][11][3]=14;
d[17][11][4]=18;


d[17][14][0]=7;            //--Karditsa--Farsala--//
d[17][14][1]=10;
d[17][14][2]=12;
d[17][14][3]=14;
d[17][14][4]=18;
//-------------------------------------------------------------------------------------------------//
```

```
d[20][2][0]=7;        //--Iwannina--Athina--//
d[20][2][1]=10;
d[20][2][2]=12;
d[20][2][3]=14;
d[20][2][4]=18;

d[2][20][0]=7;        //--Athina--Iwannina--//
d[2][20][1]=12;
d[2][20][2]=14;
d[2][20][3]=18;
d[2][20][4]=20;

d[20][5][0]=6;         //--Iwannina--Larisa--//
d[20][5][1]=11;
d[20][5][2]=14;
d[20][5][3]=16;
d[20][5][4]=19;

d[5][20][0]=7;         //--Larisa--Iwannina--//
d[5][20][1]=13;
d[5][20][2]=15;
d[5][20][3]=19;
d[5][20][4]=21;

d[20][8][0]=6;         //--Iwannina--Volos--//
d[20][8][1]=11;
d[20][8][2]=14;
d[20][8][3]=17;
d[20][8][4]=19;

d[8][20][0]=7;         //--Volos--Iwannina--//
d[8][20][1]=11;
d[8][20][2]=14;
d[8][20][3]=17;
d[8][20][4]=19;

d[20][11][0]=7;          //--Iwannina--Thessaloniki--//
d[20][11][1]=11;
d[20][11][2]=13;
d[20][11][3]=16;
d[20][11][4]=18;

d[11][20][0]=7;          //--Thessaloniki--Iwannina--//
d[11][20][1]=11;
d[11][20][2]=13;
d[11][20][3]=17;
```

```
d[11][20][4]=19;

d[20][14][0]=6;         //--Iwannina--Farsala--//
d[20][14][1]=10;
d[20][14][2]=12;
d[20][14][3]=14;
d[20][14][4]=17;

d[14][20][0]=7;         //--Farsala--Iwannina--//
d[14][20][1]=10;
d[14][20][2]=14;
d[14][20][3]=16;
d[14][20][4]=18;

d[20][17][0]=6;         //--Iwannina--Karditsa--//
d[20][17][1]=12;
d[20][17][2]=14;
d[20][17][3]=16;
d[20][17][4]=18;

d[17][20][0]=8;         //--Karditsa--Iwannina--//
d[17][20][1]=13;
d[17][20][2]=16;
d[17][20][3]=17;
d[17][20][4]=19;
//-------------------------------------------------------------------------------//
d[23][2][0]=7;        //--Aleksandroupoli--Athina--//
d[23][2][1]=10;
d[23][2][2]=14;
d[23][2][3]=16;
d[23][2][4]=18;

d[2][23][0]=7;        //--Athina--Aleksandroupoli--//
d[2][23][1]=12;
d[2][23][2]=14;
d[2][23][3]=18;
d[2][23][4]=19;

d[23][5][0]=7;        //--Aleksandroupoli--Larisa--//
d[23][5][1]=10;
d[23][5][2]=12;
d[23][5][3]=14;
d[23][5][4]=18;

d[5][23][0]=7;        //--Larisa--Aleksandroupoli--//
d[5][23][1]=12;
```

```
d[5][23][2]=14;
d[5][23][3]=18;
d[5][23][4]=20;

d[23][8][0]=7;        //--Aleksandroupoli--Volos--//
d[23][8][1]=10;
d[23][8][2]=12;
d[23][8][3]=14;
d[23][8][4]=18;

d[8][23][0]=7;        //--Volos--Aleksandroupoli--//
d[8][23][1]=12;
d[8][23][2]=14;
d[8][23][3]=18;
d[8][23][4]=20;

d[23][11][0]=7;        //--Aleksandroupoli--Thessaloniki--//
d[23][11][1]=10;
d[23][11][2]=14;
d[23][11][3]=16;
d[23][11][4]=18;

d[11][23][0]=7;        //--Thessaloniki--Aleksandroupoli--//
d[11][23][1]=12;
d[11][23][2]=14;
d[11][23][3]=18;
d[11][23][4]=19;

d[23][14][0]=7;        //--Aleksandroupoli--Farsala--//
d[23][14][1]=10;
d[23][14][2]=12;
d[23][14][3]=14;
d[23][14][4]=18;

d[14][23][0]=7;        //--Farsala--Aleksandroupoli--//
d[14][23][1]=12;
d[14][23][2]=14;
d[14][23][3]=18;
d[14][23][4]=20;

d[23][17][0]=7;        //--Aleksandroupoli--Karditsa--//
d[23][17][1]=10;
d[23][17][2]=12;
d[23][17][3]=14;
d[23][17][4]=18;
```

```
d[17][23][0]=7;        //--Karditsa--Aleksandroupoli--//
d[17][23][1]=12;
d[17][23][2]=14;
d[17][23][3]=18;
d[17][23][4]=20;

d[23][20][0]=7;        //--Aleksandroupoli--Iwannina--//
d[23][20][1]=10;
d[23][20][2]=12;
d[23][20][3]=14;
d[23][20][4]=18;

d[20][23][0]=7;        //--Iwannina--Aleksandroupoli--//
d[20][23][1]=12;
d[20][23][2]=14;
d[20][23][3]=18;
d[20][23][4]=20;
//-------------------------------------------------------------------------------//
d[26][2][0]=7;        //--Katerini--Athina--//
d[26][2][1]=12;
d[26][2][2]=14;
d[26][2][3]=18;
d[26][2][4]=20;

d[2][26][0]=7;        //--Athina--Katerini--//
d[2][26][1]=12;
d[2][26][2]=14;
d[2][26][3]=18;
d[2][26][4]=20;


d[26][5][0]=7;        //--Katerini--Larisa--//
d[26][5][1]=12;
d[26][5][2]=14;
d[26][5][3]=18;
d[26][5][4]=20;

d[5][26][0]=7;        //--Larisa--Katerini--//
d[5][26][1]=12;
d[5][26][2]=14;
d[5][26][3]=18;
d[5][26][4]=20;

d[26][8][0]=7;        //--Katerini--Volos--//
d[26][8][1]=12;
d[26][8][2]=14;
```

```
d[26][8][3]=18;
d[26][8][4]=20;

d[8][26][0]=7;        //--Volos--Katerini--//
d[8][26][1]=12;
d[8][26][2]=14;
d[8][26][3]=18;
d[8][26][4]=20;

d[26][11][0]=7;       //--Katerini--Thessaloniki--//
d[26][11][1]=12;
d[26][11][2]=14;
d[26][11][3]=18;
d[26][11][4]=20;

d[11][26][0]=7;       //--Thessaloniki--Katerini--//
d[11][26][1]=12;
d[11][26][2]=14;
d[11][26][3]=18;
d[11][26][4]=20;

d[26][14][0]=7;       //--Katerini--Farsala--//
d[26][14][1]=12;
d[26][14][2]=14;
d[26][14][3]=18;
d[26][14][4]=20;

d[14][26][0]=7;       //--Farsala--Katerini--//
d[14][26][1]=12;
d[14][26][2]=14;
d[14][26][3]=18;
d[14][26][4]=20;

d[26][17][0]=7;       //--Katerini--Karditsa--//
d[26][17][1]=12;
d[26][17][2]=14;
d[26][17][3]=18;
d[26][17][4]=20;

d[17][26][0]=7;       //--Karditsa--Katerini--//
d[17][26][1]=12;
d[17][26][2]=14;
d[17][26][3]=18;
d[17][26][4]=20;

d[26][20][0]=7;       //--Katerini--Iwannina--//
```

```
d[26][20][1]=12;
d[26][20][2]=14;
d[26][20][3]=18;
d[26][20][4]=20;

d[20][26][0]=7;       //--Iwannina--Katerini--//
d[20][26][1]=12;
d[20][26][2]=14;
d[20][26][3]=18;
d[20][26][4]=20;

d[26][23][0]=7;       //--Katerini--Alexandroupoli--//
d[26][23][1]=12;
d[26][23][2]=14;
d[26][23][3]=18;
d[26][23][4]=20;

d[23][26][0]=7;       //--Alexandroupoli--Katerini--//
d[23][26][1]=12;
d[23][26][2]=14;
d[23][26][3]=18;
d[23][26][4]=20;
//-------------------------------------------------------------------------------//
d[29][2][0]=7;       //--Trikala--Athina--//
d[29][2][1]=12;
d[29][2][2]=14;
d[29][2][3]=18;
d[29][2][4]=20;

d[2][29][0]=7;       //--Athina--Trikala--//
d[2][29][1]=12;
d[2][29][2]=14;
d[2][29][3]=18;
d[2][29][4]=20;


d[29][5][0]=7;       //--Trikala--Larisa--//
d[29][5][1]=12;
d[29][5][2]=14;
d[29][5][3]=18;
d[29][5][4]=20;

d[5][29][0]=7;       //--Larisa--Trikala--//
d[5][29][1]=12;
d[5][29][2]=14;
d[5][29][3]=18;
```

```
d[5][29][4]=20;

d[29][8][0]=7;        //--Trikala--Volos--//
d[29][8][1]=12;
d[29][8][2]=14;
d[29][8][3]=18;
d[29][8][4]=20;

d[8][29][0]=7;        //--Volos--Trikala--//
d[8][29][1]=12;
d[8][29][2]=14;
d[8][29][3]=18;
d[8][29][4]=20;

d[29][11][0]=7;        //--Trikala--Thessaloniki--//
d[29][11][1]=12;
d[29][11][2]=14;
d[29][11][3]=18;
d[29][11][4]=20;

d[11][29][0]=7;        //--Thessaloniki--Trikala--//
d[11][29][1]=12;
d[11][29][2]=14;
d[11][29][3]=18;
d[11][29][4]=20;

d[29][14][0]=7;        //--Trikala--Farsala--//
d[29][14][1]=12;
d[29][14][2]=14;
d[29][14][3]=18;
d[29][14][4]=20;

d[14][29][0]=7;        //--Farsala--Trikala--//
d[14][29][1]=12;
d[14][29][2]=14;
d[14][29][3]=18;
d[14][29][4]=20;

d[29][17][0]=7;        //--Trikala--Karditsa--//
d[29][17][1]=12;
d[29][17][2]=14;
d[29][17][3]=18;
d[29][17][4]=20;

d[17][29][0]=7;        //--Karditsa--Trikala--//
d[17][29][1]=12;
```

```
d[17][29][2]=14;
d[17][29][3]=18;
d[17][29][4]=20;

d[29][20][0]=7;        //--Trikala--Iwannina--//
d[29][20][1]=12;
d[29][20][2]=14;
d[29][20][3]=18;
d[29][20][4]=20;

d[20][29][0]=7;        //--Iwannina--Trikala--//
d[20][29][1]=12;
d[20][29][2]=14;
d[20][29][3]=18;
d[20][29][4]=20;

d[29][23][0]=7;        //--Trikala--Alexandroupoli--//
d[29][23][1]=12;
d[29][23][2]=14;
d[29][23][3]=18;
d[29][23][4]=20;

d[23][29][0]=7;        //--Alexandroupoli--Trikala--//
d[23][29][1]=12;
d[23][29][2]=14;
d[23][29][3]=18;
d[23][29][4]=20;

d[29][26][0]=7;        //--Trikala--Katerini--//
d[29][26][1]=12;
d[29][26][2]=14;
d[29][26][3]=18;
d[29][26][4]=20;

d[26][29][0]=7;        //--Katerini--Trikala--//
d[26][29][1]=12;
d[26][29][2]=14;
d[26][29][3]=18;
d[26][29][4]=20;
//-------------------------------------------------------------------------------//

//------------------------Dromologia entos polewn--------------------------------//
d[0][1][0]=6;          //---Athina----//
d[0][1][1]=11;
d[0][1][2]=14;
d[0][1][3]=17;
```

```
d[0][1][4]=21;

d[0][2][0]=6;
d[0][2][1]=11;
d[0][2][2]=14;
d[0][2][3]=17;
d[0][2][4]=21;

d[1][0][0]=6;
d[1][0][1]=11;
d[1][0][2]=14;
d[1][0][3]=17;
d[1][0][4]=21;

d[2][0][0]=6;
d[2][0][1]=11;
d[2][0][2]=14;
d[2][0][3]=17;
d[2][0][4]=21;

d[1][2][0]=6;
d[1][2][1]=11;
d[1][2][2]=14;
d[1][2][3]=17;
d[1][2][4]=21;

d[2][1][0]=6;
d[2][1][1]=11;
d[2][1][2]=14;
d[2][1][3]=17;
d[2][1][4]=21;
//-----------------------------//
d[3][4][0]=6;          //---Larisa----//
d[3][4][1]=11;
d[3][4][2]=14;
d[3][4][3]=17;
d[3][4][4]=21;

d[4][3][0]=6;
d[4][3][1]=11;
d[4][3][2]=14;
d[4][3][3]=17;
d[4][3][4]=21;

d[3][5][0]=6;
d[3][5][1]=11;
```

```
d[3][5][2]=14;
d[3][5][3]=17;
d[3][5][4]=21;

d[5][3][0]=6;
d[5][3][1]=11;
d[5][3][2]=14;
d[5][3][3]=17;
d[5][3][4]=21;

d[4][5][0]=6;
d[4][5][1]=11;
d[4][5][2]=14;
d[4][5][3]=17;
d[4][5][4]=21;

d[5][4][0]=6;
d[5][4][1]=11;
d[5][4][2]=14;
d[5][4][3]=17;
d[5][4][4]=21;
//-----------------------------------------------//
d[6][7][0]=6;          //---Volos----//
d[6][7][1]=11;
d[6][7][2]=14;
d[6][7][3]=17;
d[6][7][4]=21;

d[7][6][0]=6;
d[7][6][1]=11;
d[7][6][2]=14;
d[7][6][3]=17;
d[7][6][4]=21;

d[6][8][0]=6;
d[6][8][1]=11;
d[6][8][2]=14;
d[6][8][3]=17;
d[6][8][4]=21;

d[8][6][0]=6;
d[8][6][1]=11;
d[8][6][2]=14;
d[8][6][3]=17;
d[8][6][4]=21;
```

```
d[8][7][0]=6;
d[8][7][1]=11;
d[8][7][2]=14;
d[8][7][3]=17;
d[8][7][4]=21;

d[7][8][0]=6;
d[7][8][1]=11;
d[7][8][2]=14;
d[7][8][3]=17;
d[7][8][4]=21;
//-----------------------------------------------------------//
d[9][10][0]=6;          //---Thessaloniki----//
d[9][10][1]=11;
d[9][10][2]=14;
d[9][10][3]=17;
d[9][10][4]=21;

d[10][9][0]=6;
d[10][9][1]=11;
d[10][9][2]=14;
d[10][9][3]=17;
d[10][9][4]=21;

d[9][11][0]=6;
d[9][11][1]=11;
d[9][11][2]=14;
d[9][11][3]=17;
d[9][11][4]=21;

d[11][9][0]=6;
d[11][9][1]=11;
d[11][9][2]=14;
d[11][9][3]=17;
d[11][9][4]=21;

d[10][11][0]=6;
d[10][11][1]=11;
d[10][11][2]=14;
d[10][11][3]=17;
d[10][11][4]=21;

d[11][10][0]=6;
d[11][10][1]=11;
d[11][10][2]=14;
d[11][10][3]=17;
```

```
d[11][10][4]=21;
//-----------------------------------------------------------//
d[12][13][0]=6;          //---Farsala----//
d[12][13][1]=11;
d[12][13][2]=14;
d[12][13][3]=17;
d[12][13][4]=21;

d[13][12][0]=6;
d[13][12][1]=11;
d[13][12][2]=14;
d[13][12][3]=17;
d[13][12][4]=21;

d[12][14][0]=6;
d[12][14][1]=11;
d[12][14][2]=14;
d[12][14][3]=17;
d[12][14][4]=21;

d[14][12][0]=6;
d[14][12][1]=11;
d[14][12][2]=14;
d[14][12][3]=17;
d[14][12][4]=21;

d[13][14][0]=6;
d[13][14][1]=11;
d[13][14][2]=14;
d[13][14][3]=17;
d[13][14][4]=21;

d[14][13][0]=6;
d[14][13][1]=11;
d[14][13][2]=14;
d[14][13][3]=17;
d[14][13][4]=21;
//-----------------------------------------------------------//
d[15][16][0]=6;          //---Karditsa----//
d[15][16][1]=11;
d[15][16][2]=14;
d[15][16][3]=17;
d[15][16][4]=21;

d[16][15][0]=6;
d[16][15][1]=11;
```

```
d[16][15][2]=14;
d[16][15][3]=17;
d[16][15][4]=21;

d[15][17][0]=6;
d[15][17][1]=11;
d[15][17][2]=14;
d[15][17][3]=17;
d[15][17][4]=21;

d[17][15][0]=6;
d[17][15][1]=11;
d[17][15][2]=14;
d[17][15][3]=17;
d[17][15][4]=21;

d[16][17][0]=6;
d[16][17][1]=11;
d[16][17][2]=14;
d[16][17][3]=17;
d[16][17][4]=21;

d[17][16][0]=6;
d[17][16][1]=11;
d[17][16][2]=14;
d[17][16][3]=17;
d[17][16][4]=21;
//-------------------------------------------------------------//
d[19][20][0]=6;          //---Iwannina----//
d[19][20][1]=11;
d[19][20][2]=14;
d[19][20][3]=17;
d[19][20][4]=21;

d[20][19][0]=6;
d[20][19][1]=11;
d[20][19][2]=14;
d[20][19][3]=17;
d[20][19][4]=21;
//-------------------------------------------------------------//
d[21][22][0]=6;          //---Aleksandroupoli----//
d[21][22][1]=11;
d[21][22][2]=14;
d[21][22][3]=17;
d[21][22][4]=21;
d[22][21][0]=6;
```

```
d[22][21][1]=11;
d[22][21][2]=14;
d[22][21][3]=17;
d[22][21][4]=21;

d[21][23][0]=6;
d[21][23][1]=11;
d[21][23][2]=14;
d[21][23][3]=17;
d[21][23][4]=21;

d[23][21][0]=6;
d[23][21][1]=11;
d[23][21][2]=14;
d[23][21][3]=17;
d[23][21][4]=21;

d[22][23][0]=6;
d[22][23][1]=11;
d[22][23][2]=14;
d[22][23][3]=17;
d[22][23][4]=21;

d[23][22][0]=6;
d[23][22][1]=11;
d[23][22][2]=14;
d[23][22][3]=17;
d[23][22][4]=21;
//-----------------------------------------------------------//
d[24][25][0]=6;          //---Katerini----//
d[24][25][1]=11;
d[24][25][2]=14;
d[24][25][3]=17;
d[24][25][4]=21;

d[25][24][0]=6;
d[25][24][1]=11;
d[25][24][2]=14;
d[25][24][3]=17;
d[25][24][4]=21;

d[24][26][0]=6;
d[24][26][1]=11;
d[24][26][2]=14;
d[24][26][3]=17;
d[24][26][4]=21;
```

```
d[26][24][0]=6;
d[26][24][1]=11;
d[26][24][2]=14;
d[26][24][3]=17;
d[26][24][4]=21;

d[25][26][0]=6;
d[25][26][1]=11;
d[25][26][2]=14;
d[25][26][3]=17;
d[25][26][4]=21;

d[26][25][0]=6;
d[26][25][1]=11;
d[26][25][2]=14;
d[26][25][3]=17;
d[26][25][4]=21;
//-------------------------------------------------------------//
d[27][28][0]=6;          //---Trikala----//
d[27][28][1]=11;
d[27][28][2]=14;
d[27][28][3]=17;
d[27][28][4]=21;

d[28][27][0]=6;
d[28][27][1]=11;
d[28][27][2]=14;
d[28][27][3]=17;
d[28][27][4]=21;

d[27][29][0]=6;
d[27][29][1]=11;
d[27][29][2]=14;
d[27][29][3]=17;
d[27][29][4]=21;

d[29][27][0]=6;
d[29][27][1]=11;
d[29][27][2]=14;
d[29][27][3]=17;
d[29][27][4]=21;

d[28][29][0]=6;
d[28][29][1]=11;
d[28][29][2]=14;
```

```
d[28][29][3]=17;
d[28][29][4]=21;

d[29][28][0]=6;
d[29][28][1]=11;
d[29][28][2]=14;
d[29][28][3]=17;
d[29][28][4]=21;
//-------------------------------------------------------------//

IloEnv env;
try {
IloModel model (env);

typedef IloArray<IloNumArray> IloNumMatrix2x2;
typedef IloArray<IloNumMatrix2x2> IloNumMatrix3x3;
typedef IloArray<IloNumMatrix3x3> IloNumMatrix4x4;
typedef IloArray<IloNumMatrix4x4> IloNumMatrix5x5;

typedef IloArray<IloNumVarArray> IloNumVarMatrix2x2;
typedef IloArray<IloNumVarMatrix2x2> IloNumVarMatrix3x3;
typedef IloArray<IloNumVarMatrix3x3> IloNumVarMatrix4x4;
typedef IloArray<IloNumVarMatrix4x4> IloNumVarMatrix5x5;


typedef IloArray<IloRangeArray> IloRangeMatrix2x2;
typedef IloArray<IloRangeMatrix2x2> IloRangeMatrix3x3;
typedef IloArray<IloRangeMatrix3x3> IloRangeMatrix4x4;
typedef IloArray<IloRangeMatrix4x4> IloRangeMatrix5x5;

IloCplex cplex(env);
//----------Metavliti Apofashs X ---------------------------------------//

IloNumVarMatrix2x2 Xij(env,0);
for (i=0;i<imax;i++){
    IloNumVarArray Xj(env,0);
    for (j=0;j<jmax;j++){
                    char Metakinisi[70];
                    sprintf(Metakinisi,"Xij(i%d,j%d)",i,j);
        IloNumVar X(env,0,1,ILOINT,Metakinisi);
        Xj.add(X);
            }
            Xij.add(Xj);
    }
//-------------------------------------------------------------------//
```

```
//----------Metavliti Apofashs Y ---------------------------------------//

IloNumVarMatrix2x2 Yin(env,0);
for (i=0;i<imax;i++){
    IloNumVarArray Yn(env,0);
    for (n=0;n<nmax;n++){
                    char Anaxor[70];
                    sprintf(Anaxor,"Yin(i%d,n%d)",i,n);
        IloNumVar Y(env,0,1,ILOINT,Anaxor);
        Yn.add(Y);
            }
            Yin.add(Yn);
    }

//-----------------------------------------------------------------------//

//----------Metavliti Apofashs Z ---------------------------------------//

IloNumVarMatrix2x2 Zjn(env,0);
for (j=0;j<jmax;j++){
    IloNumVarArray Zn(env,0);
    for (n=0;n<nmax;n++){
                    char Afik[70];
                    sprintf(Afik,"Zjn(j%d,n%d)",j,n);
        IloNumVar Z(env,0,1,ILOINT,Afik);
        Zn.add(Z);
            }
            Zjn.add(Zn);
    }

//-----------------------------------------------------------------------//

//----------Metavliti Apofashs AT----------------------------------------//


IloNumVarMatrix3x3 ATjin(env,0);
for (j=0;j<imax;j++){
IloNumVarMatrix2x2 ATin(env,0);
for (i=0;i<imax;i++){
    IloNumVarArray ATn(env,0);
            for (n=0;n<nmax;n++){
                    char XronAna[70];
                    sprintf(XronAna,"ATjin(j%d,j%d,n%d)",j,i,n);
        IloNumVar AT(env,0,24,ILOFLOAT,XronAna);
        ATn.add(AT);
            }
```

```
                ATin.add(ATn);
        }
    ATjin.add(ATin);
}
//-------------------------------------------------------------------------//

//----------Metavliti Apofashs DT-----------------------------------------//

IloNumVarMatrix3x3 DTijn(env,0);
for (i=0;i<imax;i++){
IloNumVarMatrix2x2 DTjn(env,0);
for (j=0;j<jmax;j++){
    IloNumVarArray DTn(env,0);
                for (n=0;n<nmax;n++){
                        char XronAna[70];
                        sprintf(XronAna,"DTijn(i%d,j%d,n%d)",i,j,n);
            IloNumVar DT(env,0,24,ILOFLOAT,XronAna);
            DTn.add(DT);
                }
                DTjn.add(DTn);
    }
  DTijn.add(DTjn);
}

//-------------------------------------------------------------------//


//--------------------Metavliti U Sub Tour-----------------------------------//

 IloNumVarArray Ui(env,0);
    for (i=0;i<imax;i++){
                        char Seira[70];
                        sprintf(Seira,"Ui(i%d)",i);
            IloNumVar U(env,1,N,ILOFLOAT,Seira);
            Ui.add(U);
                }

//-------------------------------------------------------------------------//

//-----------------PERIORISMOI-----------------------------------------------//



//------------------------------Periorismos Metavlhths Apofashsh Σxij<=1 -------------//

IloRangeArray Per2i(env,0);
```

```
for (i=0;i<imax;i++){
    IloExpr expr(env,0);
     for (j=0;j<jmax;j++){
                if(i!=j){
     expr+=Xij[i][j];
     }
     }
     char P2[60];
     sprintf(P2,"Per2i(i%d)",i);
     float LB=-IloInfinity,UB=1;
     IloRange Per2(env,LB,expr,UB,P2);
     expr.end();
     model.add(Per2);
     Per2i.add(Per2);
}
//-------------------------------------------------------------------------------------------------//

//---------------Periorismos ekinisis-----------------------------------------------------------//
                        //IloRangeArray orgna(env,0);
                //                  for (a=0;a<amax;a++){
                                            IloExpr expr(env,0);
                                                    for (j=0;j<jmax;j++){
                                                        if (j!=O){
                                                    expr+=Xij[O][j];
                                                        }
                                                    }
                                            char or[60];
                                            sprintf(or,"orgna(a)");
                                            float LB=1,UB=1;
                                            IloRange orgn(env,LB,expr,UB,or);
                                            expr.end();
                                            model.add(orgn);
                                    //orgna.add(orgn);
                                    //}
    //-------------------------------------------------------------------------------------------------//


//---------------Periorismos Afiksis------------------------------------------------------------//

    IloRangeArray desta(env,0);
                                    for (a=0;a<amax;a++){
                                            IloExpr expr(env,0);
                                            for (i=0;i<imax;i++){
                                                    if(i!=D){
                                                    expr+=Xij[i][D];
                                                    }
```

```
                                        }
                                        char de[60];
                                        sprintf(de,"desti(a%d)",a);
                                        float LB=1,UB=1;
                                        IloRange dest(env,LB,expr,UB,de);
                                        expr.end();
                                        model.add(dest);
                                        desta.add(dest);
                                }
//-------------------------------------------------------------------------------------------------//


//--------------------Periorismos afiksis-anaxwrisis apo komvo-----------------------------------//

 IloRangeArray Afiksi_Anaxorisij(env,0);
for (j=0;j<jmax;j++){
    if(j!=D){
        IloExpr expr(env,0);
        IloExpr expr1(env,0);
        IloExpr expr2(env,0);
        for (i=0;i<imax;i++){
        expr1+=Xij[i][j];

        }
        for (k=0;k<kmax;k++){
                        if(k!=O){
                    expr2+=Xij[j][k];
                        }
        }
        expr+=expr2-expr1;
                                        char Demand_B[60];
                                        sprintf(Demand_B,"Afiksi_Anaxorisij(j%d)",j);
                                        float LB=0,UB=0;
                                        IloRange Afiksi_Anaxorisi(env,LB,expr,UB,Demand_B);
                                        expr.end();
            expr1.end();
            expr2.end();
                                        model.add(Afiksi_Anaxorisi);
                                        Afiksi_Anaxorisij.add(Afiksi_Anaxorisi);
    }
    else{
        IloRangeArray terma(env,0);
                        for (a=0;a<amax;a++){
                                        IloExpr expr(env,0);
                                        for (i=0;i<imax;i++){
                                                if(i!=D){
```

```
                                        expr+=Xij[D][i];
                                        }
                                }
                                char te[60];
                                sprintf(te,"termi(a%d)",a);
                                float LB=0,UB=0;
                                IloRange term(env,LB,expr,UB,te);
                                expr.end();
                                model.add(term);
                                desta.add(term);
                        }

    }
}

//-------------------------------------------------------------------------------------------//

//----------------------Periorismos Mh Epistrofhs-------------------------------------//

    IloRangeMatrix2x2 Mh_Epistrofhij(env,0);
    for (i=0;i<imax;i++){
            IloRangeArray Mh_Epistrofhj(env,0);
            for (j=0;j<jmax;j++){
                    IloExpr expr(env,0);
                    if(i!=j){
                            expr+=Xij[i][j]+Xij[j][i]-1;
                    }
                    char Epistr[60];
                    sprintf(Epistr,"Mh_Epistrofhij(i%d,j%d)",i,j);
                    float LB=-IloInfinity,UB=0;
                    IloRange Mh_Epistrofh(env,LB,expr,UB,Epistr);
                    expr.end();
                    model.add(Mh_Epistrofh);
                    Mh_Epistrofhj.add(Mh_Epistrofh);
                    }
            Mh_Epistrofhij.add(Mh_Epistrofhj);
    }

//-------------------------------------------------------------------------//




//----------------------Periorismos Mh Epistrofhs sthn Afetiria----------------//
        IloRangeArray afeta(env,0);
                        for (a=0;a<amax;a++){
```

```
                                        IloExpr expr(env,0);
                                          for (j=0;j<jmax;j++){
                                                  if(j!=O){
                                                  expr+=Xij[j][O];
                                                  }
                                        }
                                        char af[60];
                                        sprintf(af,"afetj(a%d)",a);
                                        float LB=0,UB=0;
                                        IloRange afet(env,LB,expr,UB,af);
                                        expr.end();
                                        model.add(afet);
                                        desta.add(afet);
                                }
//-----------------------------------------------------------------//


//-----------------------------------------Periorismoi Xronou-----------------------------------//



//-----------------------------------Xroniki Stigmi(Epithumia) Anaxwrisis----------------//

        IloRangeArray Per11a(env,0);
          for (a=0;a<amax;a++){
                  IloExpr expr(env,0);
                                        expr+= Yin[O][0]-1 ;
                char P11[60];
                sprintf(P11,"Per11a(a%d)",a);
                int LB=0, UB=0;
                IloRange Per11(env,LB,expr,UB,P11);
                                         expr.end();
                model.add(Per11);
                Per11a.add(Per11);

                                }
//-------------------------------------------------------------------------------------------//


//-----------------------------Periorismos Metavlhths Apofashsh Yin<=1 ---------------------//

IloRangeArray Per3i(env,0);
for (i=0;i<imax;i++){
    IloExpr expr(env,0);
      for (n=0;n<nmax;n++){
                if(i!=O){
```

```
        expr+=Yin[i][n];
        }
        }
        char P3[60];
        sprintf(P3,"Per3i(i%d)",i);
        float LB=-IloInfinity,UB=1;
        IloRange Per3(env,LB,expr,UB,P3);
        expr.end();
        model.add(Per3);
        Per3i.add(Per3);
}
//-------------------------------------------------------------------------------------------//


//-----------------------------------Xronikos Periorismos_6-----------------------------------//

IloRangeArray Per6i(env,0);
for (i=0;i<imax;i++){
            IloExpr expr(env,0);
            IloExpr expr3(env,0);
            IloExpr expr4(env,0);
            for (j=0;j<jmax;j++){
                                    if(i!=j){
            expr3+=Xij[i][j];
                        }
                        }
            for (n=0;n<nmax;n++){
                            expr4+=Yin[i][n];
            }
            expr+=expr3-expr4;
                                            char P6[60];
                                            sprintf(P6,"Per6i(i%d)",i);
                                            float LB=0,UB=0;
                                            IloRange Per6(env,LB,expr,UB,P6);
                                            expr.end();
                expr3.end();
                expr4.end();
                                            model.add(Per6);
                                            Per6i.add(Per6);
    }
//-------------------------------------------------------------------------------------------------------//


//-----------------------------------  ΣXij = ΣZjn  ----------------------------------------------------//

IloRangeArray Per77j(env,0);
```

```
for (j=0;j<jmax;j++){
            IloExpr expr(env,0);
            IloExpr expr3(env,0);
            IloExpr expr4(env,0);
            for (i=0;i<imax;i++){
                                    if(i!=j){
            expr3+=Xij[i][j];
                        }
                        }
            for (n=0;n<nmax;n++){
                            expr4+=Zjn[j][n];
            }
            expr+=expr3-expr4;
                                            char P77[60];
                                            sprintf(P77,"Per77j(j%d)",j);
                                            float LB=0,UB=0;
                                            IloRange Per77(env,LB,expr,UB,P77);
                                            expr.end();
                expr3.end();
                expr4.end();
                                            model.add(Per77);
                                            Per77j.add(Per77);
    }
//-------------------------------------------------------------------------------------------//


//-----------------------------------  Yin - Zjn <= (1-Xij)M  ----------------------------------------//

IloRangeMatrix3x3 Per22ijn(env,0);
                for (i=0;i<imax;i++){
                IloRangeMatrix2x2 Per22jn(env,0);
                for (j=0;j<jmax;j++){
             IloRangeArray Per22n(env,0);
                  for (n=0;n<nmax;n++){
                    IloExpr expr(env,0);
                            if(i!=j){
                                    expr+= Yin[i][n]-Zjn[j][n]-(1-Xij[i][j])*M ;
                            }
                            char P22[60];
                            sprintf(P22,"Per22ijn(i%d,j%d,n%d)",i,j,n);
                            float LB=-IloInfinity,UB=0;
                            IloRange Per22(env,LB,expr,UB,P22);
                            expr.end();
                            model.add(Per22);
                            Per22n.add(Per22);
                            }
```

```
                Per22jn.add(Per22jn);
            }
                Per22ijn.add(Per22jn);
            }
        //-------------------------------------------------------------------------------------------------------------//


        //--------------------------------- dijn - (3-Xij-Yin-Zjn)*M <= DTijn  ----------------------------//

        IloRangeMatrix3x3 Per7ijn(env,0);
            for (i=0;i<imax;i++){
            IloRangeMatrix2x2 Per7jn(env,0);
            for (j=0;j<jmax;j++){
                IloRangeArray Per7n(env,0);
                for (n=0;n<nmax;n++){
                    IloExpr expr(env,0);
                    if(i!=j){
                        expr+=  (d[i][j][n])  -  (3-Xij[i][j]-Yin[i][n]-Zjn[j][n])*M  -
DTijn[i][j][n];
                    }
                    char P7[60];
                    sprintf(P7,"Per7ijn(i%d,j%d,n%d)",i,j,n);
                    float LB=-IloInfinity,UB=0;
                    IloRange Per7(env,LB,expr,UB,P7);
                    expr.end();
                    model.add(Per7);
                    Per7n.add(Per7);
                    }
                Per7jn.add(Per7n);
            }
            Per7ijn.add(Per7jn);
            }

        //-------------------------------------------------------------------------------------------------------------//


        //------------------------------ DTijn <= dijn + (3-Xij-Yin-Zjn)*M  ----------------------------------//

        IloRangeMatrix3x3 Per8ijn(env,0);
            for (i=0;i<imax;i++){
            IloRangeMatrix2x2 Per8jn(env,0);
            for (j=0;j<jmax;j++){
                IloRangeArray Per8n(env,0);
                for (n=0;n<nmax;n++){
                    IloExpr expr(env,0);
                    if(i!=j){
```

```
                                expr+=  DTijn[i][j][n]  -  (d[i][j][n])  -  (3-Xij[i][j]-Yin[i][n]-
Zjn[j][n])*M;
                        }
                        char P8[60];
                        sprintf(P8,"Per8ijn(i%d,j%d,n%d)",i,j,n);
                        float LB=-IloInfinity,UB=0;
                        IloRange Per8(env,LB,expr,UB,P8);
                        expr.end();
                        model.add(Per8);
                        Per8n.add(Per8);
                        }
                Per8jn.add(Per8n);
        }
        Per8ijn.add(Per8jn);
        }
    //-------------------------------------------------------------------------------------------------//

    //------------------------------- -Xij*M <= DTijn ------------------------------------------------//

        IloRangeMatrix3x3 Per9ijn(env,0);
        for (i=0;i<imax;i++){
                IloRangeMatrix2x2 Per9jn(env,0);
                for (j=0;j<jmax;j++){
            IloRangeArray Per9n(env,0);
                 for (n=0;n<nmax;n++){
                  IloExpr expr(env,0);
                        if(i!=j){
                                expr+=-(Xij[i][j])*M - DTijn[i][j][n] ;
                        }
                        char P9[60];
                        sprintf(P9,"Per9ijn(i%d,j%d,n%d)",i,j,n);
                        float LB=-IloInfinity,UB=0;
                        IloRange Per9(env,LB,expr,UB,P9);
                        expr.end();
                        model.add(Per9);
                        Per9n.add(Per9);
                        }
                Per9jn.add(Per9n);
        }
                Per9ijn.add(Per9jn);
        }

    //-------------------------------------------------------------------------------------------------//
    //----------------------------  DTijn <= Xijn*M  ---------------------------------------//

        IloRangeMatrix3x3 Per10ijn(env,0);
```

```
    for (i=0;i<imax;i++){
      IloRangeMatrix2x2 Per10jn(env,0);
            for (j=0;j<jmax;j++){
            IloRangeArray Per10n(env,0);
            for (n=0;n<nmax;n++){
            IloExpr expr(env,0);
                  if(i!=j){
                        expr+= DTijn[i][j][n]- (Xij[i][j])*M ;
                  }
                  char P10[60];
                  sprintf(P10,"Per10ijn(i%d,j%d,n%d)",i,j,n);
                  float LB=-IloInfinity,UB=0;
                  IloRange Per10(env,LB,expr,UB,P10);
                  expr.end();
                  model.add(Per10);
                  Per10n.add(Per10);
                  }
            Per10jn.add(Per10n);
    }
Per10ijn.add(Per10jn);
    }
//-------------------------------------------------------------------------------------------------//


//---------------------------------Periorismos -Yin*M <= DTijn   ------------------------------//

    IloRangeMatrix3x3 Per90ijn(env,0);
    for (i=0;i<imax;i++){
            IloRangeMatrix2x2 Per90jn(env,0);
            for (j=0;j<jmax;j++){
        IloRangeArray Per90n(env,0);
             for (n=0;n<nmax;n++){
              IloExpr expr(env,0);
                  if(i!=j){
                        expr+=-(Yin[i][n])*M - DTijn[i][j][n] ;
                  }
                  char P90[60];
                  sprintf(P90,"Per90ijn(i%d,j%d,n%d)",i,j,n);
                  float LB=-IloInfinity,UB=0;
                  IloRange Per90(env,LB,expr,UB,P90);
                  expr.end();
                  model.add(Per90);
                  Per90n.add(Per90);
                  }
            Per90jn.add(Per90n);
    }
```

```
            Per90ijn.add(Per90jn);
    }
//-------------------------------------------------------------------------------------------------//


//----------------------------------Periorismos DTijn <= Yin*M-------------------------------------//

    IloRangeMatrix3x3 Per91ijn(env,0);
    for (i=0;i<imax;i++){
     IloRangeMatrix2x2 Per91jn(env,0);
            for (j=0;j<jmax;j++){
            IloRangeArray Per91n(env,0);
            for (n=0;n<nmax;n++){
            IloExpr expr(env,0);
                    if(i!=j){
                            expr+= DTijn[i][j][n]- (Yin[i][n])*M ;
                    }
                    char P91[60];
                    sprintf(P91,"Per91ijn(i%d,j%d,n%d)",i,j,n);
                    float LB=-IloInfinity,UB=0;
                    IloRange Per91(env,LB,expr,UB,P91);
                    expr.end();
                    model.add(Per91);
                    Per91n.add(Per91);
                    }
            Per91jn.add(Per91n);
    }
Per91ijn.add(Per91jn);
    }
//-------------------------------------------------------------------------------------------------------//

//----------------------------------Periorismos -Zjn*M <= DTijn   ------------------------- -------------//

    IloRangeMatrix3x3 Per92ijn(env,0);
    for (i=0;i<imax;i++){
            IloRangeMatrix2x2 Per92jn(env,0);
            for (j=0;j<jmax;j++){
        IloRangeArray Per92n(env,0);
              for (n=0;n<nmax;n++){
                IloExpr expr(env,0);
                    if(i!=j){
                            expr+=-(Zjn[j][n])*M - DTijn[i][j][n] ;
                    }
                    char P92[60];
                    sprintf(P92,"Per92ijn(i%d,j%d,n%d)",i,j,n);
                    float LB=-IloInfinity,UB=0;
```

```
                    IloRange Per92(env,LB,expr,UB,P92);
                    expr.end();
                    model.add(Per92);
                    Per92n.add(Per92);
                    }
            Per92jn.add(Per92n);
    }
            Per92ijn.add(Per92jn);
    }
//-------------------------------------------------------------------------------------------------------------//

//---------------------------------Periorismos DTijn <= Zjn*M---------------------------------//

    IloRangeMatrix3x3 Per93ijn(env,0);
    for (i=0;i<imax;i++){
     IloRangeMatrix2x2 Per93jn(env,0);
            for (j=0;j<jmax;j++){
            IloRangeArray Per93n(env,0);
            for (n=0;n<nmax;n++){
            IloExpr expr(env,0);
                    if(i!=j){
                            expr+= DTijn[i][j][n]- (Zjn[j][n])*M ;
                    }
                    char P93[60];
                    sprintf(P93,"Per93ijn(i%d,j%d,n%d)",i,j,n);
                    float LB=-IloInfinity,UB=0;
                    IloRange Per93(env,LB,expr,UB,P93);
                    expr.end();
                    model.add(Per93);
                    Per93n.add(Per93);
                    }
            Per93jn.add(Per93n);
    }
Per93ijn.add(Per93jn);
    }
//-------------------------------------------------------------------------------------------------------------//

//---------------------------------Periorismos DTkjz >= DTikn + tik ---------------------------------//

IloRangeMatrix5x5 Per122ikjnz(env,0);
    for (i=0;i<imax;i++){
IloRangeMatrix4x4 Per122kjnz(env,0);
    for (k=0;k<kmax;k++){
IloRangeMatrix3x3 Per122jnz(env,0);
    for (j=0;j<jmax;j++){
    IloRangeMatrix2x2 Per122nz(env,0);
```

```
        for (n=0;n<nmax;n++){
                IloRangeArray Per122z(env,0);
                for (z=0;z<zmax;z++){
                        IloExpr expr(env,0);
                        if(i!=j && i!=k && j!=k){
                                expr+= DTijn[i][k][n]+t[i][k]*Xij[i][k]- DTijn[k][j][z]-(4-Xij[i][k]-
Yin[i][n]-Xij[k][j]-Yin[k][z])*M;
                        }
                        char P122[60];
                        sprintf(P122,"Per122ikjnz(i%d,k%d,j%d,n%d,z%d)",i,k,j,n,z);
                        float LB=-IloInfinity,UB=0;
                        IloRange Per122(env,LB,expr,UB,P122);
                        expr.end();
                        model.add(Per122);
                        Per122z.add(Per122);
                        }
                Per122nz.add(Per122z);
        }
        Per122jnz.add(Per122nz);
        }
        Per122kjnz.add(Per122jnz);
        }
       Per122ikjnz.add(Per122kjnz);
        }
    //-------------------------------------------------------------------------------------------------------//

    //---------------------------------Periorismos 12-------------------------------------------------------//

    IloRangeMatrix3x3 Per12ijn(env,0);
        for (i=0;i<imax;i++){
        IloRangeMatrix2x2 Per12jn(env,0);
        for (j=0;j<jmax;j++){
                IloRangeArray Per12n(env,0);
                for (n=0;n<nmax;n++){
                        IloExpr expr(env,0);
                        if(i!=j){
                                expr+=      DTijn[i][j][n]+t[i][j]*Xij[i][j]-(2-Xij[i][j]-Zjn[j][n])*M-
ATjin[j][i][n];
                        }
                        char P12[60];
                        sprintf(P12,"Per12ijn(i%d,j%d,n%d)",i,j,n);
                        float LB=-IloInfinity,UB=0;
                        IloRange Per12(env,LB,expr,UB,P12);
                        expr.end();
                        model.add(Per12);
                        Per12n.add(Per12);
```

```
                    }
                Per12jn.add(Per12n);
            }
            Per12ijn.add(Per12jn);
            }
        //-------------------------------------------------------------------------------------------------------//

        //---------------------------------Periorismos 15-------------------------------------------------------//

        IloRangeMatrix3x3 Per15ijn(env,0);
            for (i=0;i<imax;i++){
            IloRangeMatrix2x2 Per15jn(env,0);
            for (j=0;j<jmax;j++){
                    IloRangeArray Per15n(env,0);
                    for (n=0;n<nmax;n++){
                            IloExpr expr(env,0);
                            if(i!=j){
                                    expr+=  ATjin[j][i][n]  -  (DTijn[i][j][n]+t[i][j]*Xij[i][j]+(2-Xij[i][j]-
Zjn[j][n])*M);
                            }
                            char P15[60];
                            sprintf(P15,"Per15ijn(i%d,j%d,n%d)",i,j,n);
                            float LB=-IloInfinity,UB=0;
                            IloRange Per15(env,LB,expr,UB,P15);
                            expr.end();
                            model.add(Per15);
                            Per15n.add(Per15);
                            }
                    Per15jn.add(Per15n);
            }
            Per15ijn.add(Per15jn);
            }
        //-------------------------------------------------------------------------------------------------------//


        //---------------------------------Periorismos 16-------------------------------------------------------//

             IloRangeMatrix3x3 Per16ijn(env,0);
                    for (i=0;i<imax;i++){
                    IloRangeMatrix2x2 Per16jn(env,0);
                    for (j=0;j<jmax;j++){
                        IloRangeArray Per16n(env,0);
                          for (n=0;n<nmax;n++){
                            IloExpr expr(env,0);
                            if(i!=j){
                                    expr+=-(Xij[i][j])*M - ATjin[j][i][n] ;
```

```
                    }
                    char P16[60];
                    sprintf(P16,"Per16ijn(i%d,j%d,n%d)",i,j,n);
                    float LB=-IloInfinity,UB=0;
                    IloRange Per16(env,LB,expr,UB,P16);
                    expr.end();
                    model.add(Per16);
                    Per16n.add(Per16);
                    }
            Per16jn.add(Per16n);
    }
            Per16ijn.add(Per16jn);
            }
//-------------------------------------------------------------------------------------------------------------//


//---------------------------------Periorismos 17-----------------------------------------------------//

     IloRangeMatrix3x3 Per17ijn(env,0);
            for (i=0;i<imax;i++){
            IloRangeMatrix2x2 Per17jn(env,0);
            for (j=0;j<jmax;j++){
        IloRangeArray Per17n(env,0);
             for (n=0;n<nmax;n++){
              IloExpr expr(env,0);
                    if(i!=j){
                            expr+= ATjin[j][i][n] - (Xij[i][j])*M ;
                    }
                    char P17[60];
                    sprintf(P17,"Per17ijn(i%d,j%d,n%d)",i,j,n);
                    float LB=-IloInfinity,UB=0;
                    IloRange Per17(env,LB,expr,UB,P17);
                    expr.end();
                    model.add(Per17);
                    Per17n.add(Per17);
                    }
            Per17jn.add(Per17jn);
    }
            Per17ijn.add(Per17jn);
    }
//-------------------------------------------------------------------------------------------------------------//



//---------------------------------Periorismos 156----------------------------------------------------//
```

```
IloRangeMatrix3x3 Per156ijn(env,0);
        for (i=0;i<imax;i++){
        IloRangeMatrix2x2 Per156jn(env,0);
        for (j=0;j<jmax;j++){
    IloRangeArray Per156n(env,0);
         for (n=0;n<nmax;n++){
          IloExpr expr(env,0);
                if(i!=j){
                        expr+=-(Zjn[j][n])*M - ATjin[j][i][n] ;
                }
                char P156[60];
                sprintf(P156,"Per156ijn(i%d,j%d,n%d)",i,j,n);
                float LB=-IloInfinity,UB=0;
                IloRange Per156(env,LB,expr,UB,P156);
                expr.end();
                model.add(Per156);
                Per156n.add(Per156);
                }
        Per156jn.add(Per156n);
  }
        Per156ijn.add(Per156jn);
        }
//-------------------------------------------------------------------------------------------------------------------//


//---------------------------------Periorismos 157-----------------------------------------------------------//

IloRangeMatrix3x3 Per157ijn(env,0);
        for (i=0;i<imax;i++){
        IloRangeMatrix2x2 Per157jn(env,0);
        for (j=0;j<jmax;j++){
    IloRangeArray Per157n(env,0);
         for (n=0;n<nmax;n++){
          IloExpr expr(env,0);
                if(i!=j){
                        expr+= ATjin[j][i][n] - (Zjn[j][n])*M ;
                }
                char P157[60];
                sprintf(P157,"Per157ijn(i%d,j%d,n%d)",i,j,n);
                float LB=-IloInfinity,UB=0;
                IloRange Per157(env,LB,expr,UB,P157);
                expr.end();
                model.add(Per157);
                Per157n.add(Per157);
                }
        Per157jn.add(Per157jn);
```

```
        }
                Per157ijn.add(Per157jn);
        }
    //-------------------------------------------------------------------------------------------------------------//

    //----------------------------------------  Periorismoi PI  ---------------------------------------------//

    //---------------------------------Periorismos Afiksis PI-----------------------------------------------//


                        IloRangeArray AFPIa(env,0);
                            for (a=0;a<amax;a++){
                                IloExpr expr(env,0);
                                for (i=0;i<imax;i++){
                                    if(i!=P){
                                    expr+=Xij[i][P];
                                    }
                                }
                                char api[60];
                                sprintf(api,"AFPIi(a%d)",a);
                                float LB=1,UB=1;
                                IloRange AFPI(env,LB,expr,UB,api);
                                expr.end();
                                model.add(AFPI);
                                AFPIa.add(AFPI);
                            }
    //-------------------------------------------------------------------------------------------------------------//


    //-----------------------------------------Periorismos paramonhs sto PI ----------------------//

                IloRangeMatrix3x3 Per84ijn(env,0);
                for (i=0;i<imax;i++){
                IloRangeMatrix2x2 Per84jn(env,0);
                for (j=0;j<jmax;j++){
            IloRangeArray Per84n(env,0);
                 for (n=0;n<nmax;n++){
                  IloExpr expr(env,0);
                        if(i!=j){
                                expr+=  WTP-DTijn[P][j][n]+ATjin[P][i][n]-(3-Xij[P][j]-Yin[P][n]-
Zjn[P][n])*M ;
                        }
                        char P84[60];
                        sprintf(P84,"Per84ijn(i%d,j%d,n%d)",j,n);
                        float LB=-IloInfinity,UB=0;
                        IloRange Per84(env,LB,expr,UB,P84);
```

```
                    expr.end();
                    model.add(Per84);
                    Per84n.add(Per84);
                    }
            Per84jn.add(Per84n);
    }
                        Per84ijn.add(Per84jn);
    }
//-------------------------------------------------------------------------------------------------------------------//



//----------------------------------Periorismos AT[D] = ToA ----------------------------------------//
IloRangeArray Per288a(env,0);
        for (a=0;a<amax;a++){
            IloExpr expr(env,0);
                    for (i=0;a<amax;a++){
                        for (n=0;n<nmax;n++){
                      expr+= ATjin[D][i][n]-ToA ;
                        }
                    }
            char P288[60];
            sprintf(P288,"Per11a(a%d)",a);
            int LB=-IloInfinity, UB=0;
            IloRange Per288(env,LB,expr,UB,P288);
                        expr.end();
            model.add(Per288);
            Per288a.add(Per288);

                    }
//-------------------------------------------------------------------------------------------------------------------//



//-----------------------------Sub-tour Elimination-----------------------------------------------//

  IloRangeMatrix2x2 Periorismos_Subrouteij(env,0);
      for (i=1;i<imax;i++){
        IloRangeArray Periorismos_Subroutej(env,0);
        for (j=1;j<jmax;j++){
          if (i!=j){
            IloExpr expr(env,0);
              if (i!=j) expr= Ui[i] - Ui[j] + N*Xij[i][j] - N ;
              char subroute[60];
              sprintf(subroute,"Periorismos_Subrouteij(i%d,j%d)",i,j);
              int LB=-IloInfinity, UB=-1;
```

```
                IloRange Periorismos_Subroute(env,LB,expr,UB,subroute);
                model.add(Periorismos_Subroute);
                Periorismos_Subroutej.add(Periorismos_Subroute);
                expr.end();
            }
        }
     Periorismos_Subrouteij.add(Periorismos_Subroutej);
    }
//-----------------------------------------------------------------------------------------------------------//


//---------------Antikeimeniki Sinartisi-----------------------------------------------------------------//

IloExpr expr1(env);

for (i=0;i<imax;i++){
    for (j=0;j<jmax;j++){
            if(i!=j){
            expr1+=Xij[i][j]*C[i][j];
            }
    }
    }


model.add(IloMinimize(env, expr1));
expr1.end();

cplex.extract(model);
cplex.exportModel("otinanai.lp");


cplex.solve();

if (!cplex.solve ()){
                    env.error()<<"Faild to optimize LP."<<endl;
                    throw(-1);
            }


            env.out()<<"Solution status = " <<cplex.getStatus()<<endl;
            env.out()<<"Solution value = " <<cplex.getObjValue()<<endl;



for(i=0;i<imax;i++){
    for(j=0;j<jmax;j++){
```

```
   if(i!=j){
     int g = cplex.getValue(Xij[i][j]);
     if(g!=0) cout<<"Xij"<<"("<<i<<","<<j<<")"<<"="<<g<<endl;
                              }

     }
}


for(i=0;i<imax;i++){
     for(n=0;n<nmax;n++){
     int g = cplex.getValue(Yin[i][n]);
     if(g!=0) cout<<"Yin"<<"("<<i<<","<<n<<")"<<"="<<g<<endl;

     }
}


for(j=0;j<jmax;j++){
     for(n=0;n<nmax;n++){
     int g = cplex.getValue(Zjn[j][n]);
     if(g!=0) cout<<"Zjn"<<"("<<j<<","<<n<<")"<<"="<<g<<endl;

     }
}


 for (j=0;j<jmax;j++){
      for(i=0;i<imax;i++){
                for(n=0;n<nmax;n++){
                          if(i!=j){
      float g = cplex.getValue(ATjin[j][i][n]);
      if(g!=0) cout<<"ATjin"<<"("<<j<<","<<i<<","<<n<<")"<<"="<<g<<endl;
                           }
                }
      }
 }



 for (i=0;i<imax;i++){
      for(j=0;j<jmax;j++){
                for(n=0;n<nmax;n++){
                          if(i!=j){
      float g = cplex.getValue(DTijn[i][j][n]);
      if(g!=0) cout<<"DTijn"<<"("<<i<<","<<j<<","<<n<<")"<<"="<<g<<endl;
```

```
                        }
                    }
                }
            }


        }
        catch ( IloException& e){
                cerr << "concert exception caught:"<<e<<endl;
        }
        catch (...){
                cerr<<"Unknown exception caught" <<endl;
        }
        env.end();
        return 0;
    } //End main
```

ΠΑΡΑΡΤΗΜΑ Β

ΠΙΝΑΚΕΣ ΚΟΣΤΟΥΣ & ΧΡΟΝΟΥ

	j	ΑΘΗΝΑ			ΛΑΡΙΣΑ			ΒΟΛΟΣ			ΘΕΣ/ΝΙΚΗ			ΦΑΡΣΑΛΑ			ΚΑΡΔΙΤΣΑ			ΙΩΑΝΝΙΝΑ		
i		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ΑΘΗΝΑ	0	200	1	1	4	200	200	200	200	200	5.5	200	200	3.65	200	200	200	200	200	200	200	200
	1	1	200	1	200	3.7	200	200	3.8	200	200	6	200	200	4	200	200	5	200	200	8	200
	2	1	1	200	200	200	3.5	200	200	3.4	200	200	5.7	200	200	3.5	200	200	4.25	200	200	6,5
ΛΑΡΙΣΑ	3	4	200	200	200	0.4	200	0.8	200	200	1.5	200	200	0.4	200	200	200	200	200	200	200	200
	4	200	3.7	200	0.4	200	0.4	200	1	200	200	2.5	200	200	1.4	200	200	1.3	200	200	4,3	200
	5	200	200	3.5	0.4	0.4	200	200	200	0.9	200	200	2	200	200	1.1	200	200	1	200	200	3,5
ΒΟΛΟΣ	6	200	200	200	0.8	200	200	200	0.3	0.3	200	200	200	200	200	200	200	200	200	200	200	200
	7	200	3.8	200	200	1	200	0.3	200	0.3	200	2.9	200	200	1.6	200	200	200	200	200	4,8	200
	8	200	200	3.4	200	200	0.9	0.3	0.3	200	200	200	2.6	200	200	1.3	200	200	2.3	200	200	4
ΘΕΣ/ΝΙΚΗ	9	5.5	200	200	1.5	200	200	200	200	200	200	0.5	0.5	1.65	200	200	200	200	200	200	200	200
	10	200	6	200	200	2.5	200	200	2.9	200	0.5	200	0.5	200	3.5	200	200	3.5	200	200	4	200
	11	200	200	5.7	200	200	2	200	200	2.6	0.5	0.5	200	200	200	3	200	200	3	200	200	3
ΦΑΡΣΑΛΑ	12	3.65	200	200	0.4	200	200	200	200	200	1.65	200	200	200	0.2	0.2	0.3	200	200	200	200	200
	13	200	4	200	200	1.4	200	200	1.6	200	200	3.5	200	0.2	200	0.2	200	1	200	200	200	200
	14	200	200	3.5	200	200	1.1	200	200	1.3	200	200	3	0.2	0.2	200	200	200	0.8	200	200	4
ΚΑΡΔΙΤΣΑ	15	200	200	200	200	200	200	200	200	200	200	200	200	0.3	200	200	200	0.3	0.3	200	200	200
	16	200	5	200	200	1.3	200	200	200	200	200	3.5	200	200	1	200	0.3	200	0.3	200	3,8	200
	17	200	200	4.25	200	200	1	200	200	2.3	200	200	3	200	200	0.8	0.3	0.3	200	200	200	3
ΙΩΑΝΝΙΝΑ	18	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	0,3	0,3
	19	200	8	200	200	4,3	200	200	4,8	200	200	4	200	200	200	200	200	3,8	200	0,3	200	0,3
	20	200	200	6,5	200	200	3,5	200	200	4	200	200	3	200	200	4	200	200	3	0,3	0,3	200
ΑΛΕΞ/ΠΟΛΗ	21	200	200	200	200	200	200	200	200	200	5,8	200	200	200	200	200	200	200	200	200	200	200
	22	200	10	200	200	200	200	200	200	200	200	4	200	200	200	200	200	200	200	200	200	200
	23	200	200	8,5	200	200	5	200	200	5,5	200	200	3	200	200	6	200	200	6	200	200	6
ΚΑΤΕΡΙΝΗ	24	4,5	200	200	0,8	200	200	200	200	200	0,8	200	200	1	200	200	200	200	200	200	200	200
	25	200	6	200	200	1,8	200	200	200	200	200	1,4	200	200	200	200	200	200	200	200	200	200
	26	200	200	5	200	200	1,3	200	200	1,9	200	200	1	200	200	2	200	200	2,5	200	200	3
ΤΡΙΚΑΛΑ	27	200	200	200	200	200	200	200	200	200	200	200	200	0,7	200	200	0,4	200	200	200	200	200
	28	200	5,5	200	200	1,4	200	200	2,5	200	200	3,9	200	200	2	200	200	1	200	200	2,9	200
	29	200	200	4,9	200	200	1	200	200	2	200	200	3	200	200	1,6	200	200	0,7	200	200	2
ΠΑΤΡΑ	30	1,5	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	31	200	3,4	200	200	6	200	200	6,2	200	200	7,5	200	200	200	200	200	6,5	200	200	5	200
	32	200	200	2,9	200	200	5	200	200	5	200	200	6,5	200	200	5	200	200	5	200	200	4
ΗΓΟΥΜΕΝ.	33	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	34	200	8	200	200	200	200	200	200	200	200	4,5	200	200	200	200	200	200	200	200	1,4	200
	35	200	200	7	200	200	4	200	200	5	200	200	3,5	200	200	4,5	200	200	3,5	200	200	1
ΚΑΒΑΛΑ	36	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	37	200	8,5	200	200	200	200	200	200	200	200	2,4	200	200	200	200	200	200	200	200	200	200
	38	200	200	7	200	200	3,5	200	200	4	200	200	1,9	200	200	4,5	200	200	4,9	200	200	4,5
ΞΑΝΘΗ	39	200	200	200	200	200	200	200	200	200	5	200	200	200	200	200	200	200	200	200	200	200
	40	200	9,5	200	200	200	200	200	200	200	200	3,4	200	200	200	200	200	200	200	200	200	200
	41	200	200	8	200	200	4	200	200	4,5	200	200	2,5	200	200	5	200	200	5,5	200	200	5
ΠΕΙΡΑΙΑΣ	42	1	1	1	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
ΗΡΑΚΛΕΙΟ	43	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200

	j	ΑΛΕΞ/ΠΟΛΗ			ΚΑΤΕΡΙΝΗ			ΤΡΙΚΑΛΑ			ΠΑΤΡΑ			ΗΓΟΥΜΕΝ			ΚΑΒΑΛΑ			ΞΑΝΘΗ			ΠΕΙΡΑΙΑΣ	ΗΡΑΚΛΕΙΟ
i		21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
ΑΘΗΝΑ	0	200	200	200	4,5	200	200	200	200	200	1,5	200	200	200	200	200	200	200	200	200	200	200	1	200
	1	200	10	200	200	6	200	200	5,5	200	200	3,4	200	200	8	200	200	8,5	200	200	9,5	200	1	200
	2	200	200	8,5	200	200	5	200	200	4,9	200	200	2,9	200	200	7	200	200	7	200	200	8	1	200
ΛΑΡΙΣΑ	3	200	200	200	0,8	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	4	200	5,8	200	200	1,8	200	200	1,4	200	200	6	200	200	200	200	200	200	200	200	200	200	200	200
	5	200	200	5	200	200	1,3	200	200	1	200	200	5	200	200	4	200	200	3,5	200	200	4	200	200
ΒΟΛΟΣ	6	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	7	200	200	200	200	2,4	200	200	2,5	200	200	6,2	200	200	200	200	200	200	200	200	200	200	200	200
	8	200	200	5,5	200	200	1,9	200	200	2	200	200	5	200	200	5	200	200	4	200	200	4,5	200	200
ΘΕΣ/ΝΙΚΗ	9	5,8	200	200	0,8	200	200	200	200	200	200	200	200	200	200	200	200	200	200	5	200	200	200	200
	10	200	4	200	200	1,4	200	200	3,9	200	200	7,5	200	200	4,5	200	200	2,4	200	200	3,4	200	200	200
	11	200	200	3	200	200	1	200	200	3	200	200	6,5	200	200	3,5	200	200	1,9	200	200	2,5	200	200
ΦΑΡΣΑΛΑ	12	200	200	200	1	200	200	0,7	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	13	200	200	200	200	200	200	200	2	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	14	200	200	6	200	200	2	200	200	1,6	200	200	5	200	200	4,5	200	200	4,5	200	200	5	200	200
ΚΑΡΔΙΤΣΑ	15	200	200	200	200	200	200	0,4	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	16	200	200	200	200	200	200	200	1	200	200	6,5	200	200	200	200	200	200	200	200	200	200	200	200
	17	200	200	6	200	200	2,5	200	200	0,7	200	200	5	200	200	3,5	200	200	4,9	200	200	5,5	200	200
ΙΩΑΝΝΙΝΑ	18	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	19	200	200	200	200	200	200	200	3	200	200	5	200	200	1,4	200	200	200	200	200	200	200	200	200
	20	200	200	6	200	200	3	200	200	2	200	200	4	200	200	1	200	200	4,5	200	200	5	200	200
ΑΛΕΞ/ΠΟΛΗ	21	200	0.8	0.8	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	1,9	200	200	200	200
	22	0.8	200	0.8	200	200	200	200	200	200	200	200	200	200	200	200	200	2,4	200	200	1,8	200	200	200
	23	0.8	0.8	200	200	200	3,9	200	200	6	200	200	9,5	200	200	6,5	200	200	1,7	200	200	1,2	200	200
ΚΑΤΕΡΙΝΗ	24	200	200	200	200	0,4	0,4	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	25	200	200	200	0,4	200	0,4	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	26	200	200	3,9	0,4	0,4	200	200	200	2,5	200	200	6	200	200	3,5	200	200	2,4	200	200	3	200	200
ΤΡΙΚΑΛΑ	27	200	200	200	200	200	200	200	0,4	0,4	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	28	200	200	200	200	200	200	0,4	200	0,4	200	7,5	200	200	200	200	200	200	200	200	200	200	200	200
	29	200	200	6	200	200	2,5	0,4	0,4	200	200	200	6	200	200	2,9	200	200	4,7	200	200	5	200	200
ΠΑΤΡΑ	30	200	200	200	200	200	200	200	200	200	200	0,5	0,5	200	200	200	200	200	200	200	200	200	200	200
	31	200	200	200	200	200	200	200	7,5	200	0,5	200	0,5	200	200	200	200	200	200	200	8,7	200	200	200
	32	200	200	9,5	200	200	6	200	200	6	0,5	0,5	200	200	200	4,5	200	200	8	200	200	9,5	200	200
ΗΓΟΥΜΕΝ.	33	200	200	200	200	200	200	200	200	200	200	200	200	200	0,3	0,3	200	200	200	200	200	200	200	200
	34	200	200	200	200	200	200	200	200	200	200	200	200	0,3	200	0,3	200	200	200	200	200	200	200	200
	35	200	200	6,5	200	200	3,5	200	200	2,9	200	200	4,5	0,3	0,3	200	200	200	5	200	200	5,8	200	200
ΚΑΒΑΛΑ	36	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	0,4	0,4	200	200	200	200	200
	37	200	2,4	200	200	200	200	200	200	200	200	200	200	200	200	200	0,4	200	0,4	200	1,2	200	200	200
	38	200	200	1,7	200	200	2,4	200	200	4,7	200	200	8	200	200	5	0,4	0,4	200	200	200	0,9	200	200
ΞΑΝΘΗ	39	1,9	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	0,4	0,4	200	200
	40	200	1,8	200	200	200	200	200	200	200	200	200	200	200	200	200	200	1,2	200	0,4	200	0,4	200	200
	41	200	200	1,2	200	200	3	200	200	5	200	200	8,7	200	200	5,8	200	200	0,9	0,4	0,4	200	200	200
ΠΕΙΡΑΙΑΣ	42	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	6
ΗΡΑΚΛΕΙΟ	43	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	6	200

Πίνακας 0.1 : Συγκεντρωτικός πίνακας[Ι] χρόνου μετακινήσεων απο κόμβο σε κόμβο

Πηγή : Βιβλιογραφία [20,21,22,23,24,25,26,27,28,29,30,31,34]

	j	ΑΘΗΝΑ			ΛΑΡΙΣΑ			ΒΟΛΟΣ			ΘΕΣ/ΝΙΚΗ			ΦΑΡΣΑΛΑ			ΚΑΡΔΙΤΣΑ			ΙΩΑΝΝΙΝΑ			ΑΛΕΞ/ΠΟΛΗ			ΚΑΤΕΡΙΝΗ			ΤΡΙΚΑΛΑ			ΠΑΤΡΑ			ΗΓΟΥΜΕΝ.			ΚΑΒΑΛΑ			ΞΑΝΘΗ			ΠΕΙΡΑΙΑΣ	ΗΡΑΚΛΕΙΟ
i		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
ΑΘΗΝΑ	0	200	1,4	1,4	30	200	200	200	200	200	45	200	200	31	200	200	14	200	200	200	200	200	200	200	200	40	200	200	200	200	200	10	200	200	200	200	200	200	200	200	200	200	200	1,4	200
	1	1,4	200	1,4	200	25	200	200	40	200	200	40	200	200	32	200	200	36	200	200	50	200	200	75	200	200	30	200	200	28	200	200	35	200	200	70	200	200	65	200	200	70	200	1,4	200
	2	1,4	1,4	200	200	200	50	200	200	60	200	200	78	200	200	40	200	200	42,5	200	200	79,5	200	200	117	200	200	68	200	200	46	200	200	42	200	200	85	200	200	98	200	200	105	1,4	200
ΛΑΡΙΣΑ	3	30	200	200	200	1	1	5	200	200	12	200	200	9	200	200	200	200	200	200	200	200	200	200	200	12	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	4	200	25	200	1	200	1	200	7	200	200	20	200	200	5,5	200	200	7	200	200	18	200	200	55	200	200	13.5	200	200	6	200	200	46	200	200	200	200	200	200	200	200	200	200	200	200
	5	200	200	50	1	1	200	200	200	10	200	200	27	200	200	6	200	200	6,5	200	200	23	200	200	65	200	200	17	200	200	7	200	200	58	200	200	33	200	200	46	200	200	52	200	200
ΒΟΛΟΣ	6	200	200	200	5	200	200	200	0,8	0,8	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	7	200	40	200	200	7	200	0,8	200	0,8	200	26	200	200	6,5	200	200	200	200	200	25	200	200	200	200	200	200	200	200	14	200	200	41	200	200	200	200	200	200	200	200	200	200	200	200
	8	200	200	60	200	200	10	0,8	0,8	200	200	200	35,5	200	200	8	200	200	16,5	200	200	33	200	200	75	200	200	25,5	200	200	17,2	200	200	52	200	200	43	200	200	55	200	200	60	200	200
ΘΕΣ/ΝΙΚΗ	9	45	200	200	12	200	200	200	200	200	200	1	1	22	200	200	200	200	200	200	200	200	23	200	200	5	200	200	200	200	200	200	200	200	200	200	200	200	200	200	14	200	200	200	200
	10	200	40	200	200	20	200	200	26	200	1	200	1	200	23	200	200	18	200	200	28	200	200	30	200	200	8	200	200	20	200	200	60	200	200	38	200	200	14	200	200	18	200	200	200
	11	200	200	78	200	200	27	200	200	35,5	1	1	200	200	200	33	200	200	33	200	200	39	200	200	40	200	200	11	200	200	34	200	200	75	200	200	48,5	200	200	20,5	200	200	26,5	200	200
ΦΑΡΣΑΛΑ	12	31	200	200	9	200	200	200	200	200	22	200	200	200	0,6	0,6	3	200	200	200	200	200	200	200	200	13	200	200	3,5	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	13	200	32	200	200	5,5	200	200	6,5	200	200	23	200	0,6	200	0,6	200	5	200	200	200	200	200	200	200	200	200	200	200	7	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	14	200	200	40	200	200	6	200	200	8	200	200	33	0,6	0,6	200	200	200	6	200	200	26	200	200	70	200	200	22,5	200	200	9,8	200	200	45	200	200	35,5	200	200	52	200	200	58	200	200
ΚΑΡΔΙΤΣΑ	15	14	200	200	200	200	200	200	200	200	200	200	200	3	200	200	200	0,6	0,6	200	200	200	200	200	200	200	200	200	2,5	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	16	200	36	200	200	7	200	200	200	200	200	18	200	200	5	200	0,6	200	0,6	200	20	200	200	200	200	200	200	200	200	3,5	200	200	38	200	200	200	200	200	200	200	200	200	200	200	200
	17	200	200	42,5	200	200	6,5	200	200	16,5	200	200	33	200	200	6	0,6	0,6	200	200	200	25	200	200	72	200	200	23	200	200	5	200	200	47	200	200	30	200	200	52	200	200	58,5	200	200
ΙΩΑΝΝΙΝΑ	18	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	0,7	0,7	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	19	200	50	200	200	18	200	200	25	200	200	28	200	200	200	200	200	20	200	0,7	200	0,7	200	200	200	200	200	200	200	14	200	200	34	200	200	8	200	200	200	200	200	200	200	200	200
	20	200	200	79,5	200	200	23	200	200	33	200	200	39	200	200	26	200	200	25	0,7	0,7	200	200	200	76	200	200	36,5	200	200	16,6	200	200	40	200	200	13	200	200	58	200	200	64,4	200	200
ΑΛΕΞ/ΠΟΛΗ	21	200	200	200	200	200	200	200	200	200	23	200	200	200	200	200	200	200	200	200	200	200	200	1	1	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	4	200	200	200	200
	22	200	65	200	200	200	200	200	200	200	200	30	200	200	200	200	200	200	200	200	200	200	1	200	1	200	200	200	200	200	200	200	200	200	200	200	200	200	14	200	200	9,5	200	200	200
	23	200	200	117	200	200	65	200	200	75	200	200	40	200	200	70	200	200	72	200	200	76	1	1	200	200	200	49	200	200	72,5	200	200	114	200	200	87	200	200	19,5	200	200	14,5	200	200
ΚΑΤΕΡΙΝΗ	24	40	200	200	12	200	200	200	200	200	5	200	200	13	200	200	200	200	200	200	200	200	200	200	200	200	1	1	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	25	200	30	200	200	13.5	200	200	200	200	200	8	200	200	200	200	200	200	200	200	200	200	200	200	200	1	200	1	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	26	200	200	68	200	200	17	200	200	25,5	200	200	11	200	200	22,5	200	200	23	200	200	36,5	200	200	49	1	1	200	200	200	23	200	200	73	200	200	46	200	200	30	200	200	36,3	200	200
ΤΡΙΚΑΛΑ	27	200	200	200	200	200	200	200	200	200	200	200	200	3,5	200	200	2,5	200	200	200	200	200	200	200	200	200	200	200	200	0,8	0,8	200	200	200	200	200	200	200	200	200	200	200	200	200	200
	28	200	28	200	200	6	200	200	14	200	200	20	200	200	7	200	200	3,5	200	200	14	200	200	200	200	200	200	200	0,8	200	0,8	200	38	200	200	200	200	200	200	200	200	200	200	200	200
	29	200	200	46	200	200	7	200	200	17,2	200	200	34	200	200	9,8	200	200	4	200	200	16,6	200	200	72,5	200	200	23	0,8	0,8	200	200	200	50	200	200	26	200	200	53	200	200	59,3	200	200
ΠΑΤΡΑ	30	10	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	1	1	200	200	200	200	200	200	200	200	200	200	200
	31	200	28	200	200	38	200	200	40	200	200	50	200	200	200	200	200	35	200	200	27	200	200	200	200	200	200	200	200	30	200	1	200	1	200	200	200	200	200	200	200	75	200	200	200
	32	200	200	42	200	200	58	200	200	52	200	200	75	200	200	45	200	200	47	200	200	40	200	200	114	200	200	73	200	200	50	1	1	200	200	200	46	200	200	95	200	200	100	200	200
ΗΓΟΥΜΕΝ.	33	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	0,7	0,7	200	200	200	200	200	200	200	200
	34	200	50	200	200	200	200	200	200	200	200	31	200	200	200	200	200	200	200	200	8	200	200	200	200	200	200	200	200	200	200	200	200	200	0,7	200	0,7	200	200	200	200	200	200	200	200
	35	200	200	85	200	200	33	200	200	43	200	200	48,5	200	200	35,5	200	200	30	200	200	13	200	200	87	200	200	46	200	200	26	200	200	46	0,7	0,7	200	200	200	68	200	200	74,5	200	200
ΚΑΒΑΛΑ	36	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	1	1	200	200	200	200	200
	37	200	50	200	200	200	200	200	200	200	200	14	200	200	200	200	200	200	200	200	200	200	200	14	200	200	200	200	200	200	200	200	200	200	200	200	200	1	200	1	200	4	200	200	200
	38	200	200	98	200	200	46	200	200	55	200	200	20,5	200	200	52	200	200	52	200	200	58	200	200	19,5	200	200	30	200	200	53	200	200	95	200	200	68	1	1	200	200	200	6,4	200	200
ΞΑΝΘΗ	39	200	200	200	200	200	200	200	200	200	14	200	200	200	200	200	200	200	200	200	200	200	4	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	0,8	0,8	200	200
	40	200	55	200	200	200	200	200	200	200	200	18	200	200	200	200	200	200	200	200	200	200	200	9,5	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	4	200	0,8	200	200
	41	200	200	105	200	200	52	200	200	60	200	200	26,5	200	200	58	200	200	58,5	200	200	64,4	200	200	14,5	200	200	36,3	200	200	59,3	200	200	100	200	200	74,5	200	200	6,4	0,8	0,8	200	200	200	200
ΠΕΙΡΑΙΑΣ	42	1,4	1,4	1,4	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	49
ΗΡΑΚΛΕΙΟ	43	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200	49	200	200

Πίνακας 0.2 : Συγκεντρωτικός πίνακας κόστους μετακίνησης απο κόμβο σε κόμβο

Πηγή : Βιβλιογραφία [20,21,21,22,23,24,25,26,27,28,29,30,31,34]