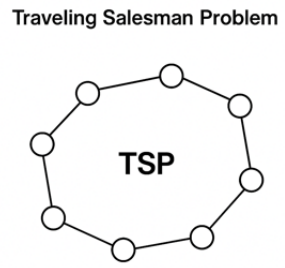
ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Γραφική Επίλυση του Προβλήματος του Περιοδεύοντος Πωλητή με τη χρήση Νευρωνικών Δικτύων όλων των τύπων και Αλγορίθμων Βαθιάς Μάθησης-Deep Learning»



Του φοιτητή
Ζαχαρέγκα Αλέξανδρου Κωνσταντίνου
Αρ. Μητρώου: 175075

Επιβλέπων
Κωνσταντίνος Γουλιάνας
Βαθμίδα: Καθηγητής

31 Μαΐου 2025

Τίτλος Π.Ε. Γραφική Επίλυση του Προβλήματος του Περιοδεύοντος Πωλητή
με τη χρήση Νευρωνικών Δικτύων όλων των τύπων και Αλγορίθμων Βαθιάς Μάθησης-Deep Learning
Κωδικός Π.Ε. 22170
Ονοματεπώνυμο φοιτητή Ζαχαρέγκας Αλέξανδρος Κωνσταντίνος
Ονοματεπώνυμο εισηγητή Κωνσταντίνος Γουλιάνας
Ημερομηνία ανάληψης Δ.Ε. 04-11-2023
Ημερομηνία περάτωσης Δ.Ε. 28-05-2025

Αφιερώνεται

Στη μητέρα μου, που υπήρξε πάντα το στήριγμά μου, με πίστεψε όταν ακόμη δεν πίστευα εγώ στον εαυτό μου.

Στα αδέρφια μου, που με τον δικό τους τρόπο με βοήθησαν να ανακαλύψω τον δρόμο που θέλω να ακολουθήσω.

Και στη Σοφία, που μπήκε στη ζωή μου σαν ηρεμία μέσα στο χάος, με έκανε να χαμογελώ στα δύσκολα και με αγάπησε απλά, αληθινά, όπως κανείς άλλος. Και στον καθηγητή μου, Κωνσταντίνο Γουλιάνα, για τη στήριξή του σε όλη την ακαδημαϊκή μου πορεία και για την πολύτιμη βοήθεια και καθοδήγηση που μου προσέφερε.

Πρόλογος

Το Πρόβλημα του Περιοδεύοντος Πωλητή (TSP) αποτελεί ένα από τα πιο διάσημα και θεμελιώδη προβλήματα βελτιστοποίησης στην επιστήμη των υπολογιστών, με εφαρμογές που εκτείνονται από τη διαχείριση εφοδιαστικής αλυσίδας και τη δρομολόγηση οχημάτων μέχρι τον σχεδιασμό μικροτσίπ και την αστρονομία. Η υπολογιστική του πολυπλοκότητα (NP-complete) το καθιστά εξαιρετικά δύσκολο να επιλυθεί με ακρίβεια για μεγάλα δεδομένα, ενώ οι παραδοσιακές μέθοδοι επίλυσης συχνά αποτυγχάνουν να παρέχουν αποδοτικές λύσεις σε πραγματικό χρόνο. Τα νευρωνικά δίκτυα και οι αλγόριθμοι βαθιάς μάθησης προσφέρουν μια καινοτόμο προσέγγιση, καθώς μπορούν να ”μάθουν” μοτίβα και να παρέχουν υψηλής ποιότητας προσεγγιστικές λύσεις σε σύνθετα προβλήματα. Ο συνδυασμός της θεωρητικής επιστήμης των υπολογιστών με τις σύγχρονες τεχνικές τεχνητής νοημοσύνης αποτελεί για εμένα ένα συναρπαστικό πεδίο έρευνας, και είναι ο βασικός λόγος που επέλεξα αυτό το θέμα. Η εκπόνηση αυτής της εργασίας μου δίνει την ευκαιρία να εμβαθύνω στις τεχνικές βελτιστοποίησης και να αποκτήσω πρακτική εμπειρία στην εφαρμογή προηγμένων αλγορίθμων μηχανικής μάθησης σε κλασικά υπολογιστικά προβλήματα.

Περίληψη

Η παρούσα πτυχιακή εργασία με τίτλο ”Γραφική Επίλυση του Προβλήματος του Περιοδεύοντος Πωλητή με τη χρήση Νευρωνικών Δικτύων όλων των τύπων και Αλγορίθμων Βαθιάς Μάθησης” πραγματεύεται την εφαρμογή τεχνικών τεχνητής νοημοσύνης στην επίλυση ενός από τα πιο γνωστά προβλήματα βελτιστοποίησης στην επιστήμη των υπολογιστών. Αρχικά παρουσιάζεται το Πρόβλημα του Περιοδεύοντος Πωλητή (TSP), η μαθηματική του διατύπωση και οι κλασικές μέθοδοι επίλυσής του, ενώ αναλύεται η υπολογιστική του πολυπλοκότητα και η σημασία του για πρακτικές εφαρμογές. Ακολουθεί μια εισαγωγή στα νευρωνικά δίκτυα, με ανάλυση των διαφόρων τύπων τους (βαθιά νευρωνικά δίκτυα, συνελικτικά νευρωνικά δίκτυα, αναδρομικά νευρωνικά δίκτυα, δίκτυα βαθιάς πεποίθησης και βαθιά ενισχυτική μάθηση). Στη συνέχεια, πραγματοποιείται εκτενής βιβλιογραφική ανασκόπηση των μεθόδων που έχουν προταθεί για την επίλυση του TSP με τη χρήση νευρωνικών δικτύων και αλγορίθμων βαθιάς μάθησης, με κατηγοριοποίηση και συγκριτική ανάλυση των προσεγγίσεων. Το πρακτικό μέρος της εργασίας περιλαμβάνει την επιλογή και υλοποίηση τεσσάρων διαφορετικών αλγορίθμων από διαφορετικές κατηγορίες και την ανάπτυξη γραφικού περιβάλλοντος που επιτρέπει την οπτικοποίηση της επίλυσης του προβλήματος στο χάρτη της Ελλάδας. Ο χρήστης μπορεί να επιλέξει διαφορετικό αριθμό πόλεων και να παρακολουθήσει σε πραγματικό χρόνο την εξέλιξη της εύρεσης της βέλτιστης διαδρομής. Τέλος, πραγματοποιείται συγκριτική αξιολόγηση των τεσσάρων αλγορίθμων με βάση την ποιότητα της λύσης, την ταχύτητα σύγκλισης και την επεκτασιμότητα τους σε προβλήματα μεγαλύτερης κλίμακας.

«Graphical Solution of the Traveling Salesman Problem using Neural Networks of All Types and Deep Learning Algorithms»

«Alexandros Konstantinos Zacharegkas»

Abstract

This thesis entitled ”Graphical Solution of the Traveling Salesman Problem using Neural Networks of all types and Deep Learning Algorithms” investigates the application of artificial intelligence techniques to solve one of the most renowned optimization problems in computer science. Initially, the Traveling Salesman Problem (TSP) is presented, including its mathematical formulation and classical solution methods, while analyzing its computational complexity and importance for practical applications. An introduction to neural networks follows, with analysis of their various types (deep neural networks, convolutional neural networks, recurrent neural networks, deep belief networks, and deep reinforcement learning). Subsequently, an extensive literature review of methods proposed for solving the TSP using neural networks and deep learning algorithms is conducted, with categorization and comparative analysis of approaches. The practical part of the thesis includes the selection and implementation of three different algorithms from distinct categories and the development of a graphical interface that allows visualization of the problem-solving process on the map of Greece. The user can select different numbers of cities and observe in real-time the evolution of finding the optimal route. Finally, a comparative evaluation of the three algorithms is performed based on solution quality, convergence speed, and their scalability to larger-scale problems.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες σε όλους όσοι συνέβαλαν στην ολοκλήρωση αυτής της πτυχιακής εργασίας.
Ιδιαίτερη ευγνωμοσύνη οφείλω στον καθηγητή μου, Κωνσταντίνο Γουλιάνα, για την πολύτιμη καθοδήγηση, τη συνεχή στήριξη και τη διαρκή του διαθεσιμότητα καθ' όλη τη διάρκεια των σπουδών μου. Η συμβολή του υπήρξε καθοριστική, όχι μόνο στην πορεία της εργασίας αυτής, αλλά και στην ευρύτερη ακαδημαϊκή μου εξέλιξη.
Ευχαριστώ από καρδιάς τη μητέρα μου για την αδιάκοπη στήριξη και πίστη της σε εμένα. Τα αδέρφια μου για την έμπνευση και την ώθηση να ανακαλύψω το πάθος μου. Και τη Σοφία, που με τη δύναμη της αγάπης της με βοήθησε να παραμείνω δημιουργικός και δυνατός ακόμα και στις πιο απαιτητικές στιγμές.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Συντομογραφίες

TSP	Travelling Salesman Problem
DL	Deep Learning
FNN	Feedforward Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
SOM	Self-Organizing Map
DRL	Deep Reinforcement Learning
ANN	Artificial Neural Network
MLP	Multilayer Perceptron
ILP	Integer Linear Programming
API	Application Programming Interface
CSV	Comma-Separated Values
ACO	Ant Colony Optimization
OR-Tools	Operations Research Tools
GNN	Graph Neural Network
BMU	Best Matching Unit

Εισαγωγή

Η βελτιστοποίηση αποτελεί κεντρικό πυλώνα σε πλήθος επιστημονικών και τεχνολογικών πεδίων, με τα προβλήματα συνδυαστικής βελτιστοποίησης να κατέχουν εξέχουσα θέση λόγω της υπολογιστικής τους πολυπλοκότητας και της ευρείας πρακτικής τους σημασίας. Ανάμεσα σε αυτά, το **Πρόβλημα του Περιοδεύοντος Πωλητή (Travelling Salesman Problem - TSP)** ξεχωρίζει ως ένα από τα πιο θεμελιώδη και πολυμελετημένα. Η πρόκληση της εύρεσης της συντομότερης δυνατής διαδρομής που επισκέπτεται ένα σύνολο πόλεων ακριβώς μία φορά και επιστρέφει στην αρχική, παρότι διατυπώνεται απλά, κρύβει τεράστια υπολογιστική δυσκολία.

Η NP-πλήρης φύση του TSP καθιστά την εύρεση της βέλτιστης λύσης με ακριβείς μεθόδους πρακτικά ανέφικτη για προβλήματα μεγάλης κλίμακας, ωθώντας την έρευνα προς την ανάπτυξη ευρετικών και προσεγγιστικών αλγορίθμων. Τα τελευταία χρόνια, η εντυπωσιακή άνοδος της **Τεχνητής Νοημοσύνης**, και ειδικότερα των **Νευρωνικών Δικτύων (ΝΔ)** και της **Βαθιάς Μάθησης (Deep Learning - DL)**, έχει φέρει επανάσταση στον τρόπο προσέγγισης τέτοιων δύσκολων προβλημάτων. Τα μοντέλα αυτά διαθέτουν την ικανότητα να ”μαθαίνουν” πολύπλοκα μοτίβα και ευρετικές στρατηγικές απευθείας από τα δεδομένα, προσφέροντας νέες, ισχυρές λύσεις για τη συνδυαστική βελτιστοποίηση.

Η παρούσα πτυχιακή εργασία επικεντρώνεται στη διερεύνηση της εφαρμογής σύγχρονων τεχνικών Νευρωνικών Δικτύων και Βαθιάς Μάθησης για την επίλυση του Προβλήματος του Περιοδεύοντος Πωλητή. Ο κύριος στόχος είναι διττός:

1. Να πραγματοποιηθεί μια **συστηματική θεωρητική μελέτη** του προβλήματος, των θεμελιωδών αρχών των ΝΔ και DL, και μια **εκτενής βιβλιογραφική ανασκόπηση** των κυριότερων προσεγγίσεων που έχουν προταθεί στη βιβλιογραφία για την επίλυση του TSP με αυτές τις τεχνικές.
2. Να προχωρήσει στην **πρακτική υλοποίηση και αξιολόγηση τριών διακριτών αλγορίθμων** που βασίζονται σε διαφορετικές φιλοσοφίες ΝΔ/DL (π.χ., μη επιβλεπόμενη μάθηση, ακολουθιακά μοντέλα με προσοχή, ενισχυτική μάθηση). Η απόδοση αυτών των αλγορίθμων θα εξεταστεί μέσω **γραφικής προσομοίωσης**, οπτικοποιώντας τη διαδικασία εύρεσης λύσης πάνω σε έναν **χάρτη της Ελλάδας**, με δυνατότητα επιλογής του αριθμού των πόλεων από τον χρήστη.

Η δομή της εργασίας ακολουθεί τους παραπάνω στόχους. Το Κεφάλαιο 1 παρουσιάζει αναλυτικά το Πρόβλημα του Περιοδεύοντος Πωλητή. Το Κεφάλαιο 2 εισάγει τις βασικές έννοιες των Νευρωνικών Δικτύων και της Βαθιάς Μάθησης. Το Κεφάλαιο 3 προσφέρει μια λεπτομερή βιβλιογραφική ανασκόπηση των σχετικών μεθόδων ΝΔ/DL για το TSP. Στη συνέχεια, παρουσιάζεται η υλοποίηση των επιλεγμένων αλγορίθμων και του συστήματος προσομοίωσης, ακολουθούμενη από τη συγκριτική αξιολόγηση των αποτελεσμάτων. Η εργασία ολοκληρώνεται με την εξαγωγή συμπερασμάτων.

Κεφάλαιο 1ο: Το Πρόβλημα Του Περιπλανώμενου Πωλητή

Η ραγδαία ανάπτυξη των υπολογιστικών μεθόδων και της Τεχνητής Νοημοσύνης έχει μετασχηματίσει τον τρόπο με τον οποίο προσεγγίζουμε σύνθετα μαθηματικά και βελτιστοποιητικά προβλήματα. Ένα

από τα πιο γνωστά και διαχρονικά παραδείγματα είναι το **Πρόβλημα του Περιοδεύοντος Πωλητή (Travelling Salesman Problem - TSP)**, το οποίο αποτελεί βασικό ζήτημα στον τομέα της συνδυαστικής βελτιστοποίησης και βρίσκει εφαρμογή σε πλήθος πρακτικών πεδίων: από τη δρομολόγηση οχημάτων και τη ρομποτική, μέχρι τη βιοπληροφορική και τα logistics.

Το TSP είναι ένα *NP-πλήρες πρόβλημα*, γεγονός που σημαίνει ότι δεν υπάρχει γνωστός πολυωνυμικός αλγόριθμος που να λύνει κάθε περίπτωση σε λογικό χρόνο. Πρακτικά, αυτό σημαίνει ότι, παρόλο που η εύρεση μιας οποιασδήποτε διαδρομής είναι εύκολη και ο έλεγχος του κόστους μιας δεδομένης διαδρομής είναι γρήγορος, η εύρεση της βέλτιστης διαδρομής απαιτεί, στη γενική περίπτωση, υπολογιστικό χρόνο που αυξάνεται εκθετικά με τον αριθμό των πόλεων, καθιστώντας το πρακτικά αδύνατο για μεγάλα σύνολα δεδομένων με ακριβείς μεθόδους. Ο αριθμός των πιθανών λύσεων αυξάνεται με ρυθμό $(n-1)!/2$ για συμμετρικές περιπτώσεις, καθιστώντας το υπολογιστικά αδύνατο για μεγάλα n.

Παρά τη φαινομενικά απλή διατύπωσή του, το TSP παραμένει σημείο αναφοράς για την αξιολόγηση πλήθους αλγορίθμων, από κλασικές ευρετικές μέχρι σύγχρονες προσεγγίσεις βαθιάς μάθησης [17, 49].

	A	B	C	D
A	–	10	15	20
B	10	–	35	25
C	15	35	–	30
D	20	25	30	–

Στο παραπάνω παράδειγμα, ο στόχος είναι να βρεθεί η διαδρομή με το ελάχιστο συνολικό κόστος, όπως A $\to$ B $\to$ D $\to$ C $\to$ A. Μία από τις βέλτιστες λύσεις είναι η A $\to$ B $\to$ D $\to$ C $\to$ A με συνολικό κόστος 10 + 25 + 30 + 15 = 80.

1.1 Μαθηματική Διατύπωση

Το TSP μπορεί να διατυπωθεί ως πρόβλημα Ακέραιου Γραμμικού Προγραμματισμού (ILP), όπου ελαχιστοποιείται η συνολική απόσταση μεταξύ n πόλεων:

$$G = (V, E), \quad V = \{v_1, \ldots, v_n\}, \quad c_{ij} : \ i \to j$$

$$x_{ij} = \begin{cases} 1, & \text{αν επιλέγεται η μετάβαση } i \to j \\ 0, & \text{διαφορετικά} \end{cases}$$

Αντικειμενική συνάρτηση:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Περιορισμοί:

- $\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i$ (κάθε πόλη φεύγει μία φορά)
- $\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j$ (κάθε πόλη επισκέπτεται μία φορά)
- Υποπεριορισμοί Miller–Tucker–Zemlin:

$$u_i - u_j + n \cdot x_{ij} \leq n - 1 \quad \forall i \neq j,\ i, j \in \{2, \ldots, n\}$$

Μια εναλλακτική διατύπωση είναι μέσω περιτάξεων:

$$\min_{\pi \in S_n} \sum_{i=1}^{n} d_{\pi(i),\pi(i+1)} \quad \text{όπου } \pi(n+1) = \pi(1)$$

Αυτή η διατύπωση αναδεικνύει τον συνδυαστικό χαρακτήρα του προβλήματος, καθώς ο χώρος αναζήτησης είναι το σύνολο S_n όλων των δυνατών περιτάξεων των n πόλεων.

1.2 Ιστορική Εξέλιξη και Θεμελιώδη Έργα

Η συστηματική μελέτη του TSP ξεκίνησε τη δεκαετία του 1950, με θεμελιώδη έργα όπως των Dantzig-Fulkerson-Johnson (1954) οι οποίοι παρουσίασαν μια πρωτοποριακή προσέγγιση βασισμένη στην κοπή επιπέδων (cutting planes) για την επίλυση ενός προβλήματος 49 πόλεων, και Held-Karp (1962) οι οποίοι ανέπτυξαν έναν αλγόριθμο δυναμικού προγραμματισμού [12]. Το 1960, οι Miller, Tucker και Zemlin εισήγαγαν τον πρώτο περιοριστικό ILP μοντέλο για εξάλειψη υποπεριοδειών [31].

Το έργο των Applegate et al. (2006) [2] αποτέλεσε την πιο πλήρη υπολογιστική μελέτη με χρήση του Concorde TSP Solver.

1.3 Περιοχές Εφαρμογής

Το TSP έχει πρακτικές εφαρμογές σε δεκάδες τομείς:

- **Logistics & μεταφορές** – σχεδιασμός διαδρομών οχημάτων και drones
- **Ηλεκτρονικά κυκλώματα** – VLSI layout drilling
- **Βιοπληροφορική** – ευθυγράμμιση γονιδιακών αλληλουχιών ή χαρτογράφηση γονιδιωμάτων
- **Ρομποτική** – βελτιστοποίηση κινήσεων ρομποτικών βραχιόνων σε γραμμές συναρμολόγησης ή κάλυψη περιοχών από αυτόνομα οχήματα

1.4 Κατηγορίες Μεθόδων Επίλυσης

Οι κύριες κατηγορίες περιλαμβάνουν:

- **Ακριβείς μέθοδοι:** ILP, δυναμικός προγραμματισμός [12]

- **Προσεγγιστικοί αλγόριθμοι:** Nearest Neighbor, MST, Christofides [11]
- **Μεταευρετικοί:** Simulated Annealing, Genetic Algorithms, Ant Colony [23]
- **Νευρωνικά δίκτυα:** Hopfield Networks [15], Self-Organizing Maps [8]
- **Deep Learning προσεγγίσεις:** Pointer Networks [49], Deep Reinforcement Learning [28], Graph Neural Networks [17], Attention-based DRL [36]

1.5 Γιατί το TSP είναι ιδανικό για Deep Learning

Το TSP είναι αντιπροσωπευτικό πρόβλημα *combinatorial optimization*, που απαιτεί έξυπνες στρατηγικές για εύρεση καλών λύσεων. Τα deep learning μοντέλα:

- Μαθαίνουν δομές γραφήματος (GNNs, BiGNN [27])
- Γενικεύουν από δεδομένα, αντί για brute force επίλυση
- Υποκαθιστούν την αναζήτηση με ”προβλεπτικό σχεδιασμό” (policy networks)

Σε αντίθεση με τους κλασικούς ευρετικούς αλγορίθμους που βασίζονται σε προκαθορισμένους κανόνες (π.χ., ’πήγαινε στην πλησιέστερη πόλη’), τα μοντέλα βαθιάς μάθησης μπορούν να μάθουν πολύ πιο σύνθετες και προσαρμοστικές στρατηγικές επίλυσης απευθείας από παραδείγματα προβλημάτων και λύσεων, ή ακόμα και μέσω ενισχυτικής μάθησης χωρίς να απαιτούνται βέλτιστες λύσεις ως δεδομένα εκπαίδευσης.

Ένα παράδειγμα είναι τα Pointer Networks που αναπαριστούν πόλεις ως ακολουθίες και μαθαίνουν να εξάγουν την σωστή σειρά διαδρομής [49], ενώ άλλα μοντέλα όπως το Attention Model του Kool et al. [21] συνδυάζουν attention με reinforcement learning για μεγαλύτερη ευελιξία.

Κεφάλαιο 2ο: Νευρωνικά Δίκτυα και Βαθιά Μάθηση

2.1 Ιστορική Αναδρομή των Νευρωνικών Δικτύων

Η ιδέα των τεχνητών νευρωνικών δικτύων αντλεί έμπνευση από τη δομή και λειτουργία του ανθρώπινου εγκεφάλου, όπως αυτή περιγράφεται από τη νευροεπιστήμη. Η πρώτη απόπειρα μοντελοποίησης ενός τεχνητού νευρώνα έγινε το 1943 από τους Warren McCulloch και Walter Pitts, οι οποίοι παρουσίασαν ένα μοντέλο με δυαδική είσοδο και έξοδο βασισμένο στη λογική άλγεβρα [29].

Η πιο ουσιαστική πρόοδος ήρθε το 1958, όταν ο Frank Rosenblatt παρουσίασε το **Perceptron**, μια απλή αρχιτεκτονική νευρωνικού δικτύου ικανή να πραγματοποιήσει γραμμική ταξινόμηση [37]. Το Perceptron θεωρήθηκε επαναστατικό για την εποχή του, όμως σύντομα φάνηκαν τα όριά του, κυρίως στην αδυναμία επίλυσης του προβλήματος XOR, όπως υπογράμμισε το 1969 το βιβλίο των Minsky και Papert [32], οδηγώντας σε μια περίοδο γνωστή ως “AI winter”.

Το ενδιαφέρον για τα νευρωνικά δίκτυα αναθερμάνθηκε τη δεκαετία του 1980 με την εισαγωγή του **αλγορίθμου backpropagation**, που επιτρέπει την εκπαίδευση πολυεπίπεδων δικτύων (MLPs) [38]. Το γεγονός αυτό άνοιξε τον δρόμο για πιο σύνθετα μοντέλα και εφαρμογές.

Η επόμενη επανάσταση ήρθε τη δεκαετία του 2000 με την εξέλιξη του **Deep Learning**. Ο Geoffrey Hinton παρουσίασε το 2006 τις έννοιες των *Deep Belief Networks*, δείχνοντας ότι τα βαθιά δίκτυα μπορούν να εκπαιδεύονται με επιτυχία [13]. Το 2012, το CNN μοντέλο **AlexNet** από τους Krizhevsky, Sutskever και Hinton κατάφερε να μειώσει δραστικά το σφάλμα στον διαγωνισμό ImageNet, σηματοδοτώντας την αρχή της εποχής του deep learning [22].

Από τότε, έχουν αναπτυχθεί νέες αρχιτεκτονικές όπως:

- **Convolutional Neural Networks (CNNs)**, ιδανικά για επεξεργασία εικόνας [25].
- **Recurrent Neural Networks (RNNs)**, για επεξεργασία χρονικών σειρών και φυσικής γλώσσας [10].
- **Generative Adversarial Networks (GANs)** [9].
- **Transformers**, αρχικά για NLP και πλέον παντού [48].

Η τεράστια πρόοδος στον τομέα οφείλεται σε μεγάλο βαθμό στην εκθετική αύξηση των διαθέσιμων υπολογιστικών πόρων, ιδίως με την ευρεία χρήση των μονάδων επεξεργασίας γραφικών (GPUs) για παράλληλους υπολογισμούς, στη δραματική αύξηση της διαθεσιμότητας μεγάλων συνόλων δεδομένων (Big Data), καθώς και στη συνεχή βελτίωση των αλγορίθμων εκπαίδευσης και των τεχνικών κανονικοποίησης. Σημαντική συμβολή είχαν επίσης πρωτοπόροι όπως ο Yann LeCun, κυρίως στην ανάπτυξη των CNNs, και ο Yoshua Bengio, σε διάφορους τομείς της βαθιάς μάθησης.

2.2 Βασικές Έννοιες και Αρχιτεκτονική Τεχνητών Νευρωνικών Δικτύων

Τα Τεχνητά Νευρωνικά Δίκτυα (*Artificial Neural Networks* - ANN) αποτελούν υπολογιστικά μοντέλα εμπνευσμένα από τον ανθρώπινο εγκέφαλο. Αποτελούνται από διασυνδεδεμένες μονάδες επεξεργασίας — τους *τεχνητούς νευρώνες* — οι οποίοι είναι οργανωμένοι σε στρώματα.

Δομή ενός ANN:

- **Είσοδος (Input Layer):** Λαμβάνει τα αρχικά δεδομένα.
- **Κρυφά στρώματα (Hidden Layers):** Εκτελούν μετασχηματισμούς και εξαγωγή χαρακτηριστικών.
- **Έξοδος (Output Layer):** Παράγει την τελική πρόβλεψη ή απόφαση.

Κάθε σύνδεση μεταξύ δύο νευρώνων χαρακτηρίζεται από ένα **βάρος** (w_{ij}) που δηλώνει τη σημασία της σύνδεσης. Η έξοδος ενός νευρώνα j υπολογίζεται με βάση την είσοδο x_i ως:

$$z_j = \sum_i w_{ij} x_i + b_j$$

και περνά από μια **συνάρτηση ενεργοποίησης** $f(z)$, όπως:

- **Sigmoid:** $f(z) = \frac{1}{1+e^{-z}}$
- **ReLU:** $f(z) = \max(0, z)$
- **Tanh:** $f(z) = \tanh(z)$

Η χρήση μη-γραμμικών συναρτήσεων ενεργοποίησης είναι κρίσιμη, καθώς επιτρέπει στα δίκτυα να μοντελοποιούν πολύπλοκες σχέσεις στα δεδομένα που δεν θα ήταν δυνατές με καθαρά γραμμικούς μετασχηματισμούς. Χωρίς μη-γραμμικότητα, ένα πολυεπίπεδο δίκτυο θα ήταν μαθηματικά ισοδύναμο με ένα μονοεπίπεδο δίκτυο, περιορίζοντας δραστικά την εκφραστική του δύναμη.

Η διαδικασία εκπαίδευσης ενός ANN γίνεται με τη χρήση της μεθόδου **backpropagation**, όπου τα σφάλματα διαδίδονται προς τα πίσω και προσαρμόζονται τα βάρη ώστε να ελαχιστοποιείται μια *συνάρτηση κόστους* (π.χ. MSE ή cross-entropy) [26, 38].

Η χρήση περισσότερων κρυφών στρωμάτων χαρακτηρίζει τα *Deep Neural Networks (DNNs)*, τα οποία μπορούν να μάθουν σύνθετες μη γραμμικές σχέσεις από τα δεδομένα. Η εκπαίδευσή τους απαιτεί μεγάλες ποσότητες δεδομένων και υπολογιστική ισχύ, κάτι που τα καθιστά αποτελεσματικά μόνο τις τελευταίες δεκαετίες, με τη συνδρομή GPU/TPU.

Ο σχεδιασμός ενός ANN απαιτεί επιλογή:

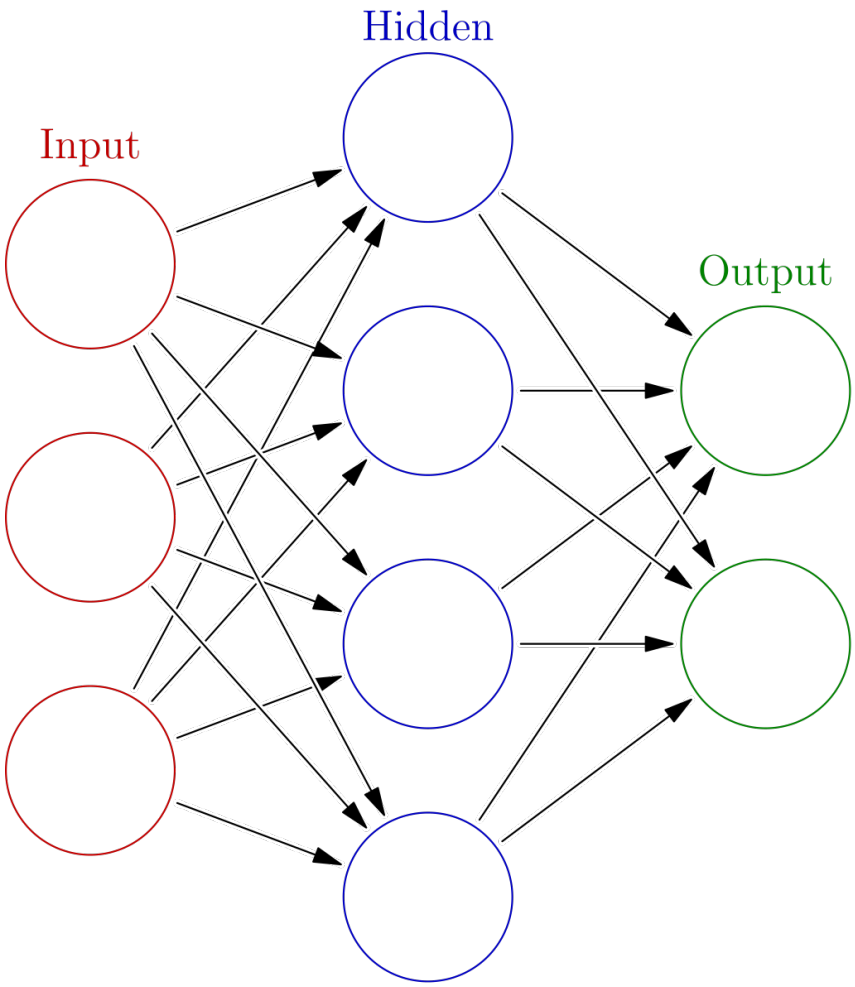
- Αριθμού και τύπων στρωμάτων,
- Συνάρτησης ενεργοποίησης,
- Τεχνικών regularization (dropout, batch norm),
- Αλγορίθμου βελτιστοποίησης (SGD, Adam, RMSprop).

Το ANN είναι το βασικό δομικό στοιχείο πάνω στο οποίο χτίζονται πιο εξειδικευμένα δίκτυα όπως τα CNN, RNN και DRL που θα παρουσιαστούν στη συνέχεια.

2.3 Βαθιά Μάθηση (Deep Learning): Ορισμός και Σημασία

Η **Βαθιά Μάθηση (Deep Learning)** αποτελεί έναν υποκλάδο της *Μηχανικής Μάθησης (Machine Learning)*, ο οποίος βασίζεται στη χρήση **πολυεπίπεδων τεχνητών νευρωνικών δικτύων** για την αυτόματη εκμάθηση αναπαραστάσεων από τα δεδομένα [26, 40].

Σε αντίθεση με τις παραδοσιακές τεχνικές μηχανικής μάθησης που βασίζονται στη χειροκίνητη εξαγωγή χαρακτηριστικών (feature engineering), τα deep networks μαθαίνουν τα σημαντικά χαρακτηριστικά



Σχήμα 2.1: Αρχιτεκτονική ενός Πολυστρωματικού Τεχνητού Νευρωνικού Δικτύου (MLP) [7]

κατευθείαν από τα δεδομένα μέσω σταδιακών επιπέδων αφαίρεσης. Κάθε στρώμα του δικτύου μαθαίνει να αναγνωρίζει όλο και πιο σύνθετα μοτίβα, βασιζόμενο στις εξόδους του προηγούμενου στρώματος. Αυτή η **ιεραρχική δομή μάθησης** (hierarchical feature learning) επιτρέπει στα βαθιά δίκτυα να κατανοούν πολύπλοκες δομές, από χαμηλού επιπέδου χαρακτηριστικά (όπως άκρα και υφές σε μια εικόνα) έως υψηλού επιπέδου έννοιες (όπως αντικείμενα ή πρόσωπα).

Κύρια χαρακτηριστικά του Deep Learning:

- Χρήση πολλαπλών κρυφών στρωμάτων (5 έως 100+ layers).
- Μεγάλος αριθμός παραμέτρων (συχνά εκατομμύρια).
- Ανάγκη για μεγάλους όγκους δεδομένων και ισχυρούς υπολογιστικούς πόρους (GPU/TPU).
- Αυτόματη εξαγωγή χαρακτηριστικών.

Η επιτυχία της βαθιάς μάθησης αποδίδεται κυρίως στους παρακάτω παράγοντες:

1. **Διαθεσιμότητα μεγάλων δεδομένων (Big Data):** από το διαδίκτυο, αισθητήρες, IoT.

2. **Υπολογιστική ισχύς:** GPU acceleration, Cloud training.
3. **Νέες τεχνικές εκπαίδευσης:** dropout, batch normalization, Adam optimizer.
4. **Βελτιωμένες αρχιτεκτονικές:** CNN, RNN, Attention.

Εφαρμογές της Βαθιάς Μάθησης:

- **Υπολογιστική Όραση (Computer Vision):** αναγνώριση αντικειμένων, προσώπων, ιατρική απεικόνιση.
- **Επεξεργασία Φυσικής Γλώσσας (NLP):** αυτόματη μετάφραση, chatbots, sentiment analysis.
- **Αυτόνομα Οχήματα:** λήψη αποφάσεων σε πραγματικό χρόνο.
- **Παιχνίδια και Ενίσχυση Μάθησης (DRL):** AlphaGo, OpenAI Five.
- **Βελτιστοποίηση και Routing:** συμπεριλαμβανομένης της επίλυσης του TSP.

Η βαθιά μάθηση έχει επιφέρει **επανάσταση** στον τρόπο με τον οποίο προσεγγίζονται προβλήματα που παλαιότερα θεωρούνταν άλυτα. Η εφαρμογή της σε συνδυαστικά προβλήματα όπως το TSP είναι ιδιαίτερα υποσχόμενη, καθώς συνδυάζει την ικανότητα γενίκευσης με την ανακάλυψη προτύπων σε πολύπλοκες δομές [1].

2.4 Τύποι Νευρωνικών Δικτύων

Κατά την τελευταία δεκαετία, η ερευνητική κοινότητα έχει αναπτύξει πολλαπλές παραλλαγές τεχνητών νευρωνικών δικτύων, η καθεμιά από τις οποίες είναι σχεδιασμένη για να εξυπηρετεί διαφορετικά είδη προβλημάτων. Σε αυτή την ενότητα παρουσιάζονται οι πιο σημαντικοί τύποι Νευρωνικών Δικτύων που σχετίζονται με τη βελτιστοποίηση, την ταξινόμηση, την πρόβλεψη και την αναπαράσταση δομών, με έμφαση στη χρησιμότητά τους για προβλήματα δρομολόγησης, όπως το TSP.

Οι βασικές κατηγορίες που θα παρουσιαστούν είναι:

1. Πλήρως Συνδεδεμένα Δίκτυα (Feedforward Neural Networks - FNN)
2. Συνελικτικά Δίκτυα (Convolutional Neural Networks - CNN)
3. Αναδρομικά Δίκτυα (Recurrent Neural Networks - RNN)
4. Δίκτυα Hopfield
5. Δίκτυα Αυτό-Οργάνωσης (Self-Organizing Maps - SOM)
6. Δίκτυα Δεικτών (Pointer Networks)
7. Βαθιά Ενισχυτική Μάθηση (Deep Reinforcement Learning - DRL)

2.4.1 Πλήρως Συνδεδεμένα Νευρωνικά Δίκτυα (Feedforward Neural Networks - FNN)

Τα Πλήρως Συνδεδεμένα ή Μπροστοτροφοδοτούμενα Νευρωνικά Δίκτυα (*Feedforward Neural Networks - FNN*) αποτελούν την πιο απλή και βασική μορφή νευρωνικού δικτύου. Η ροή πληροφορίας σε αυτά είναι μονοκατευθυντική: από την είσοδο προς την έξοδο, χωρίς ανατροφοδότηση.

Κάθε νευρώνας σε ένα στρώμα είναι συνδεδεμένος με όλους τους νευρώνες του επόμενου στρώματος, γεγονός που καθιστά τα δίκτυα αυτά ικανά να μαθαίνουν μη γραμμικούς μετασχηματισμούς των δεδομένων. Παρόλα αυτά, δεν είναι κατάλληλα για αλληλουχιακά ή χωρικά δεδομένα.

Τα FNN μπορούν να χρησιμοποιηθούν στο TSP για να μοντελοποιήσουν απλές εκτιμήσεις κόστους ή να αναγνωρίσουν μοτίβα στις συνδέσεις μεταξύ πόλεων. Ωστόσο, η απόδοσή τους είναι περιορισμένη σε σύγκριση με πιο εξειδικευμένες αρχιτεκτονικές όπως τα Pointer Networks ή τα Graph Neural Networks.

Πλεονεκτήματα:

- Απλή υλοποίηση και κατανόηση
- Κατάλληλο για βασικά προβλήματα ταξινόμησης ή παλινδρόμησης

Μειονεκτήματα:

- Δεν διαχειρίζεται χωρική ή χρονική συσχέτιση
- Υψηλό κόστος εκπαίδευσης για μεγάλα δίκτυα

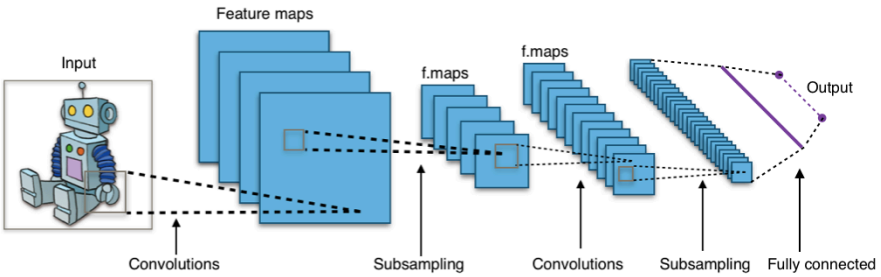
2.4.2 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks - CNN)

Τα Συνελικτικά Νευρωνικά Δίκτυα (*Convolutional Neural Networks - CNN*) αποτελούν μια ειδική κατηγορία νευρωνικών δικτύων που έχουν σχεδιαστεί για την επεξεργασία δεδομένων με δομή πλέγματος, όπως εικόνες ή χωρικά δεδομένα [25].

Το βασικό χαρακτηριστικό των CNN είναι η χρήση φίλτρων (kernels) τα οποία «σαρώνουν» την είσοδο (convolution) και εξάγουν τοπικά χαρακτηριστικά, όπως ακμές, καμπύλες ή μοτίβα. Αυτό επιτρέπει στο δίκτυο να διατηρεί τη χωρική πληροφορία και να μειώνει το πλήθος παραμέτρων.

Αρχιτεκτονικά στοιχεία ενός CNN:

- **Convolutional Layer:** εφαρμόζει φίλτρα $k \times k$ για εξαγωγή χαρακτηριστικών.
- **Activation Function:** συνήθως ReLU.
- **Pooling Layer:** μειώνει τη χωρική διάσταση (π.χ. με max pooling).
- **Fully Connected Layer:** στο τέλος, για τελική απόφαση/εκτίμηση.



Σχήμα 2.2: Τυπική αρχιτεκτονική ενός CNN με επίπεδα convolution και pooling [51]

Αν και σχεδιάστηκαν για εικόνες, τα CNN έχουν χρησιμοποιηθεί και στο TSP όταν τα δεδομένα διαμορφώνονται σε χωρικούς χάρτες ή πίνακες αποστάσεων. Επιπλέον, σε πρόσφατες μελέτες, CNN έχουν ενσωματωθεί σε υβριδικές αρχιτεκτονικές για να επεξεργάζονται «εικόνες» των μεταβολών της διαδρομής ή heatmaps σε προβλήματα βελτιστοποίησης [50].

Πλεονεκτήματα:

- Εξαιρετική απόδοση σε χωρικά δεδομένα
- Λιγότερες παράμετροι συγκριτικά με πλήρως συνδεδεμένα δίκτυα
- Μεγάλη ικανότητα γενίκευσης

Μειονεκτήματα:

- Μη ιδανικά για μη χωρικά ή αλληλουχιακά δεδομένα
- Απαιτεί σημαντικό χρόνο εκπαίδευσης και προσεκτικό σχεδιασμό αρχιτεκτονικής

2.4.3 Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks - RNN)

Τα **Αναδρομικά Νευρωνικά Δίκτυα** (*Recurrent Neural Networks - RNN*) αποτελούν αρχιτεκτονικές που έχουν σχεδιαστεί ειδικά για την επεξεργασία αλληλουχιακών δεδομένων, στα οποία η χρονική εξάρτηση μεταξύ των στοιχείων παίζει καθοριστικό ρόλο [10].

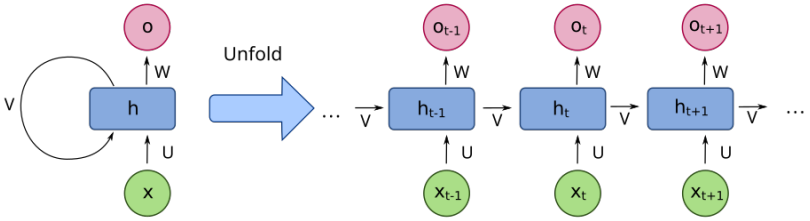
Σε αντίθεση με τα feedforward δίκτυα, τα RNN διαθέτουν **μηχανισμό μνήμης** που επιτρέπει στο δίκτυο να διατηρεί πληροφορία από προηγούμενες καταστάσεις. Αυτό επιτυγχάνεται μέσω της επανατροφοδότησης της εξόδου κάθε χρονικού βήματος πίσω στον ίδιο τον νευρώνα για το επόμενο βήμα:

$$h_t = f(W \cdot x_t + U \cdot h_{t-1} + b)$$

Όπου:

- x_t: είσοδος στο χρόνο t

- h_t: κατάσταση μνήμης
- W, U: πίνακες βαρών
- f: μη γραμμική συνάρτηση ενεργοποίησης



Σχήμα 2.3: Αναπαράσταση ενός RNN «ξεδιπλωμένου» στο χρόνο [55]

Ωστόσο, τα απλά RNN παρουσιάζουν δυσκολίες εκμάθησης μακροπρόθεσμων εξαρτήσεων, λόγω φαινομένων όπως το *vanishing gradient*. Για την επίλυση αυτών των προβλημάτων έχουν αναπτυχθεί πιο προηγμένες εκδόσεις:

Long Short-Term Memory (LSTM) [14]: Διαθέτει ειδικές μονάδες «πυλών» (gates) που ελέγχουν ποιες πληροφορίες αποθηκεύονται ή ξεχνιούνται.

Gated Recurrent Unit (GRU) [6]: Μια πιο απλοποιημένη εκδοχή του LSTM με λιγότερες πύλες αλλά παρόμοια αποτελεσματικότητα.

Τα RNN, LSTM και GRU βρίσκουν εφαρμογή στο TSP ως εργαλεία που μπορούν να διαχειριστούν την αλληλουχία επισκέψεων πόλεων. Ιδιαίτερα, χρησιμοποιούνται σε μοντέλα τύπου *Pointer Networks*, όπου η έξοδος του κάθε χρονικού βήματος δείχνει την επόμενη πόλη προς επίσκεψη.

Πλεονεκτήματα:

- Κατάλληλα για αλληλουχιακά ή χρονικά προβλήματα
- Κρατούν «μνήμη» προηγούμενων καταστάσεων
- Ισχυρή χρήση σε TSP σε συνδυασμό με attention μηχανισμούς

Μειονεκτήματα:

- Δυσκολίες εκπαίδευσης λόγω vanishing/exploding gradients
- Υψηλό υπολογιστικό κόστος για μεγάλες ακολουθίες

2.4.4 Δίκτυα Hopfield

Τα **Δίκτυα Hopfield** είναι μια μορφή αναδρομικού τεχνητού νευρωνικού δικτύου που εισήχθη το 1982 από τον John Hopfield [15]. Σχεδιάστηκαν για να λειτουργούν ως **δυναμικά συστήματα μνήμης**, ικανά να ανακαλούν μοτίβα από την αρχική τους κατάσταση μέσω ελαχιστοποίησης ενέργειας.

Το δίκτυο αποτελείται από ένα σύνολο νευρώνων x_i όπου κάθε νευρώνας είναι συνδεδεμένος με κάθε άλλον συμμετρικά. Οι συνδέσεις (βάρη) w_{ij} είναι σταθερές και δεν υπάρχουν αυτοσυνδέσεις ($w_{ii} = 0$). Το δίκτυο ακολουθεί έναν ενεργειακό κανόνα:

$$E = -\frac{1}{2} \sum_{i \neq j} w_{ij} x_i x_j + \sum_i \theta_i x_i$$

όπου E είναι η συνολική ενέργεια του συστήματος και θ_i είναι το κατώφλι (bias) κάθε νευρώνα.

Κατά την εκτέλεση, οι νευρώνες ενημερώνονται ασύγχρονα και το σύστημα σταδιακά συγκλίνει σε μια τοπικά ελάχιστη τιμή ενέργειας. Αυτή η ιδιότητα κάνει τα Hopfield Networks κατάλληλα για προβλήματα **συνδυαστικής βελτιστοποίησης**, όπως το TSP.

Εφαρμογή στο TSP: Η επίλυση του TSP με Hopfield δίκτυα προτάθηκε από τον Tank και τον Hopfield το 1985 [46], όπου η διαδρομή μοντελοποιείται ως πίνακας ενεργοποίησης $n \times n$, με n πόλεις και n χρονικά βήματα. Η ενέργεια του συστήματος περιλαμβάνει όρους για:

- κάθε πόλη να εμφανίζεται ακριβώς μία φορά
- κάθε χρονική θέση να καταλαμβάνεται από μία πόλη
- ελαχιστοποίηση συνολικού μήκους διαδρομής

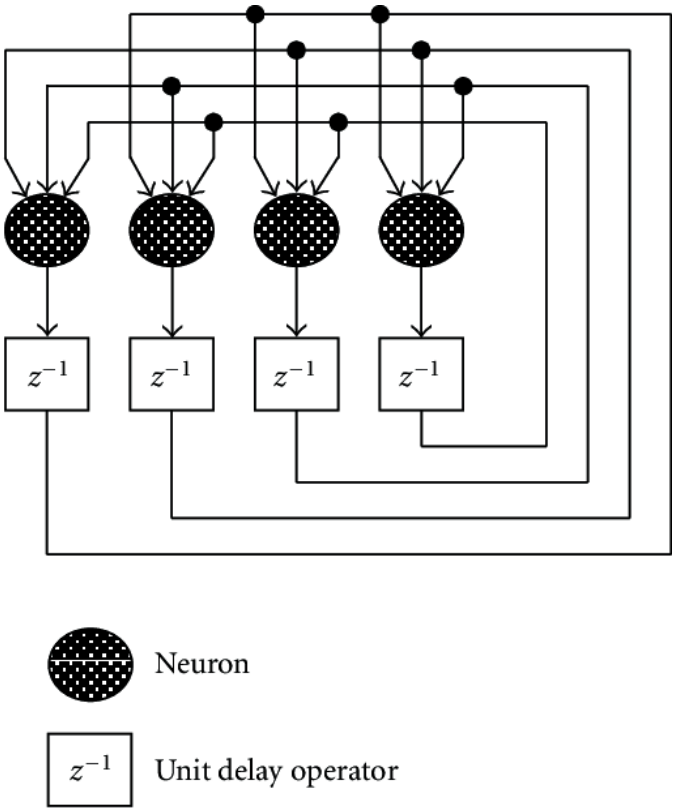
Αν και το Hopfield δίκτυο δεν εγγυάται την εύρεση της βέλτιστης λύσης, συχνά καταλήγει σε καλή προσέγγιση εντός λίγων επαναλήψεων. Επίσης, λειτουργεί γρήγορα για μικρό πλήθος πόλεων.

Πλεονεκτήματα:

- Έμφυτη δυνατότητα για συνδυαστική βελτιστοποίηση, αποτελώντας μια από τις πρώτες προσπάθειες σύνδεσης ΝΔ με τέτοια προβλήματα
- Απλή μαθηματική αναπαράσταση βασισμένη στην έννοια της ενέργειας
- Πρώιμη και ιστορικά σημαντική εφαρμογή ΤΝ στο TSP

Μειονεκτήματα:

- Ευαισθησία σε τοπικά ελάχιστα
- Χαμηλή απόδοση σε μεγάλα προβλήματα (π.χ. >20 πόλεις)
- Χρήζει πολύ προσεκτικού σχεδιασμού ενεργειακής συνάρτησης



Σχήμα 2.4: Απεικόνιση Hopfield αρχιτεκτονικής για το TSP (Tank Hopfield, 1985) [56]

2.4.5 Δίκτυα Αυτό-Οργάνωσης (Self-Organizing Maps - SOM)

Οι Αυτο-οργανούμενοι Χάρτες (SOM), που προτάθηκαν από τον Teuvo Kohonen, είναι ένας τύπος νευρωνικού δικτύου που χρησιμοποιείται κυρίως για μείωση διαστάσεων και οπτικοποίηση δεδομένων [20]. Για το TSP, χρησιμοποιείται συνήθως μια **μονοδιάστατη (κυκλική) εκδοχή του SOM**, γνωστή και ως "δακτύλιος Kohonen".

Η ιδέα είναι να δημιουργηθεί ένας δακτύλιος νευρώνων, ο οποίος σταδιακά "εφαρμόζει" πάνω στον χάρτη των πόλεων. Κάθε νευρώνας j έχει ένα διάνυσμα βαρών $\mathbf{w}_j$ που αντιστοιχεί σε μια θέση στον 2D χώρο των πόλεων. Η εκπαίδευση γίνεται ως εξής:

item Επιλέγεται τυχαία μια πόλη **p**.
item Βρίσκεται ο νευρώνας j^* ("νικητής") που είναι πιο κοντά στην πόλη **p**: $j^* = arg min_j |mathbfp - mathbfw_j|$.
item Ενημερώνονται τα βάρη του νικητή και των γειτόνων του:

$$mathbfw_j(t+1) = mathbfw_j(t) + alpha(t)cdoth(j, j^*, t)cdot(mathbfp - mathbfw_j(t))$$

όπου
$alpha(t)$ είναι ο ρυθμός εκμάθησης και $h(j, j^*, t)$ είναι η **συνάρτηση γειτονίας**, συνήθως μια Gaussian συνάρτηση που μειώνεται με την απόσταση από τον νικητή και με τον χρόνο t. Αυτή η σταδιακή μείωση της γειτονίας (neighborhood shrinking) είναι κρίσιμη για τη σύγκλιση.

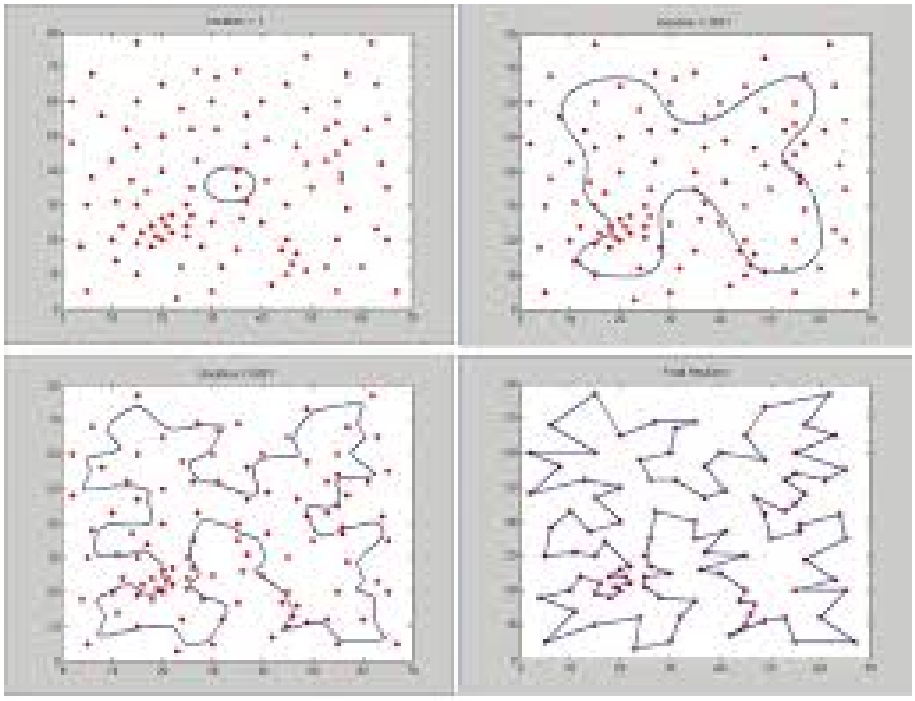
Εκπαίδευση SOM: Για κάθε δεδομένο x:

1. Επιλέγεται ο νευρώνας νικητής (Best Matching Unit - BMU), δηλαδή αυτός με τη μικρότερη απόσταση από το x.
2. Ο BMU και οι γείτονές του προσαρμόζονται ώστε να πλησιάσουν το x:

$$w_i(t+1) = w_i(t) + \alpha(t) \cdot h_{ci}(t) \cdot (x - w_i(t))$$

όπου $h_{ci}(t)$ είναι η γειτονική συνάρτηση (π.χ. Gaussian).

Εφαρμογή στο TSP: Το SOM έχει χρησιμοποιηθεί εκτενώς για την επίλυση του TSP, ιδίως μέσω γεωμετρικής προσέγγισης. Οι πόλεις τοποθετούνται ως σταθερά σημεία στο επίπεδο και το SOM προσαρμόζει ένα κυκλικό δίκτυο νευρώνων ώστε να «αγκαλιάσει» αυτές τις πόλεις. Ο τελικός κύκλος αποτελεί την προσεγγιστική λύση του TSP [8].



Σχήμα 2.5: Οπτικοποίηση του SOM για TSP: το SOM προσαρμόζεται ώστε να περάσει κοντά σε όλες τις πόλεις [44]

Πλεονεκτήματα:

- Κατάλληλο για προβλήματα clustering και οπτικής απεικόνισης

- Ικανότητα να διατηρεί την τοπολογική εγγύτητα των δεδομένων
- Αρκετά γρήγορο σε μικρούς αριθμούς πόλεων

Μειονεκτήματα:

- Χρειάζεται προσεκτική ρύθμιση παραμέτρων (learning rate, neighborhood)
- Δεν παρέχει εγγυημένα βέλτιστη λύση
- Απόδοση μειώνεται σε πολύπλοκα ή μεγάλης κλίμακας προβλήματα

2.4.6 Δίκτυα Δεικτών (Pointer Networks)

Τα **Pointer Networks** είναι μια εξελιγμένη αρχιτεκτονική νευρωνικών δικτύων που παρουσιάστηκε από τους Vinyals et al. (2015) και σχεδιάστηκε για να αντιμετωπίσει προβλήματα όπου η έξοδος δεν είναι από προκαθορισμένο λεξιλόγιο, αλλά από δυναμικά εισερχόμενα δεδομένα [49].

Αποτελούν παραλλαγή των μοντέλων **Encoder–Decoder με Attention**, όπως χρησιμοποιούνται στη Μηχανική Μετάφραση, με βασική διαφορά ότι η έξοδος αποτελεί μια αλληλουχία από δείκτες (δείχνει σε στοιχεία της εισόδου).

Στο TSP, οι πόλεις δίνονται ως είσοδος στο δίκτυο, και η έξοδος είναι η σειρά με την οποία πρέπει να επισκεφθούν οι πόλεις ώστε να ελαχιστοποιηθεί το μήκος της διαδρομής.

Αρχιτεκτονική:

- **Encoder:** Συνήθως LSTM ή GRU, κωδικοποιεί τις εισόδους (πόλεις) σε context vectors.
- **Attention:** Υπολογίζει δυναμικά την πιθανότητα επιλογής της επόμενης πόλης.
- **Decoder:** Παράγει δείκτες προς τις εισόδους, όχι λέξεις.

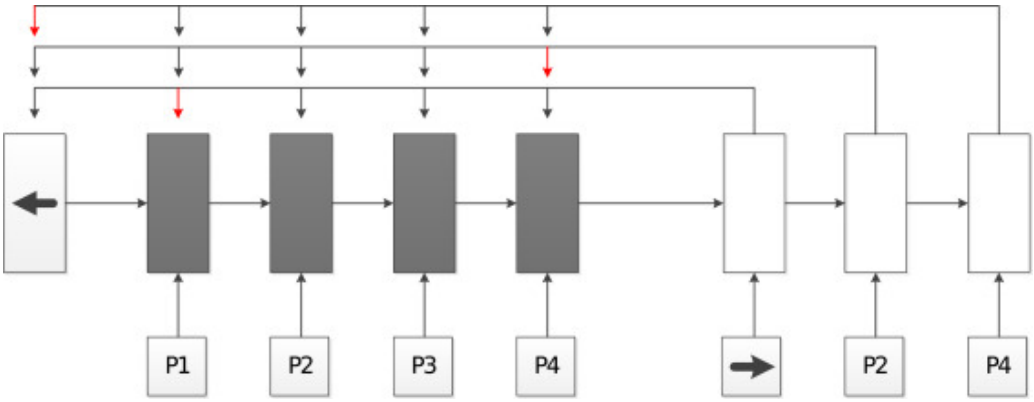
Η προσοχή (attention mechanism) καθορίζει ποια πόλη πρέπει να επιλεγεί επόμενη με βάση την τρέχουσα κατάσταση του αποκωδικοποιητή και τα embeddings των πόλεων:

$$\text{AttentionScore}(i) = \tanh(W_1 h_i + W_2 s_t)$$

όπου h_i είναι το embedding της πόλης i, και s_t είναι η κατάσταση του αποκωδικοποιητή στο χρονικό βήμα t.

Πλεονεκτήματα:

- Μαθαίνει από δεδομένα χωρίς την ανάγκη αλγοριθμικής αναπαράστασης
- Γενικεύει σε διαφορετικά μεγέθη προβλημάτων



Σχήμα 2.6: Αρχιτεκτονική Pointer Network για το TSP (Vinyals et al., 2015) [41]

- Ικανό να παράγει σχεδόν βέλτιστες λύσεις για TSP σε πραγματικό χρόνο

Μειονεκτήματα:

- Απαιτεί μεγάλο όγκο δεδομένων για εκπαίδευση
- Η εκπαίδευση μπορεί να είναι ασταθής χωρίς κατάλληλη ενίσχυση (reinforcement learning)

2.4.7 Βαθιά Ενισχυτική Μάθηση (Deep Reinforcement Learning - DRL)

Η **Βαθιά Ενισχυτική Μάθηση** (*Deep Reinforcement Learning - DRL*) αποτελεί τη σύζευξη δύο ισχυρών πεδίων της Τεχνητής Νοημοσύνης: της Ενισχυτικής Μάθησης (Reinforcement Learning - RL) και της Βαθιάς Μάθησης (Deep Learning). Το DRL χρησιμοποιεί βαθιά νευρωνικά δίκτυα για να προσεγγίσει πολιτικές, συναρτήσεις αξίας και άλλα στοιχεία ενός περιβάλλοντος λήψης αποφάσεων [33].

Σε αντίθεση με τα επιβλεπόμενα μοντέλα, τα DRL συστήματα δεν εκπαιδεύονται σε έτοιμες ετικέτες, αλλά μαθαίνουν μέσω αλληλεπίδρασης με ένα περιβάλλον, με στόχο τη μεγιστοποίηση μιας συνάρτησης ανταμοιβής (reward).

Τυπικό περιβάλλον ενισχυτικής μάθησης (Markov Decision Process - MDP):

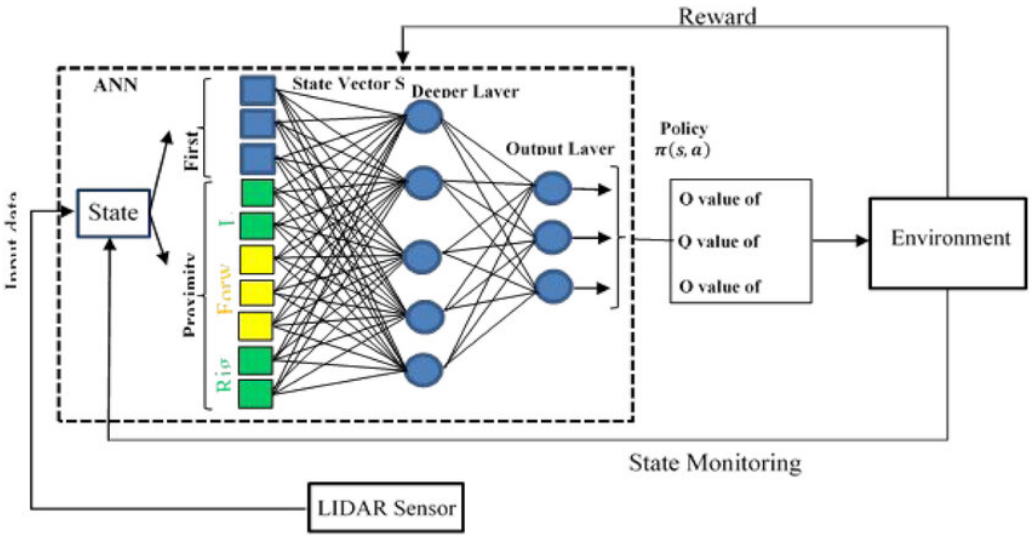
- **Καταστάσεις (States)** S
- **Ενέργειες (Actions)** A
- **Μεταβάσεις** $P(s'|s,a)$
- **Ανταμοιβές** $R(s,a)$
- **Πολιτική** $\pi(a|s)$: στρατηγική του πράκτορα

Στο πλαίσιο του TSP, κάθε κατάσταση αντιστοιχεί σε μια μερικώς ολοκληρωμένη διαδρομή, και οι ενέργειες είναι οι πιθανές επόμενες πόλεις. Ο πράκτορας επιλέγει την πόλη που θα επισκεφθεί επόμενη ώστε

να ελαχιστοποιηθεί το συνολικό μήκος της διαδρομής. Η τελική ανταμοιβή μπορεί να είναι αρνητική της συνολικής απόστασης (δηλ. όσο μικρότερη, τόσο καλύτερα).

Δημοφιλείς αλγόριθμοι DRL για TSP:

- **Policy Gradient:** Προσαρμόζει απευθείας την πολιτική $\pi_\theta(a|s)$
- **REINFORCE:** Monte Carlo μέθοδος με ενημέρωση πολιτικής
- **Actor–Critic:** Δύο δίκτυα – ένα για πολιτική (actor) και ένα για εκτίμηση αξίας (critic)
- **Proximal Policy Optimization (PPO):** Σταθεροποιημένος αλγόριθμος gradient-based



Σχήμα 2.7: Αρχιτεκτονική πράκτορα DRL για το TSP: ο πράκτορας επιλέγει την επόμενη πόλη με βάση την πολιτική του [30]

Πλεονεκτήματα:

- Ικανότητα μάθησης πολύπλοκων στρατηγικών από το μηδέν, μέσω αλληλεπίδρασης και ανταμοιβής, χωρίς την ανάγκη για ετικετοποιημένα δεδομένα (βέλτιστες λύσεις)
- Καλή γενίκευση σε διαφορετικά μεγέθη ή παραλλαγές του προβλήματος που δεν έχουν συναντηθεί κατά την εκπαίδευση
- Επεκτάσιμο σε δυναμικά ή στοχαστικά περιβάλλοντα TSP, όπου οι συνθήκες μπορεί να αλλάζουν

Μειονεκτήματα:

- Υψηλό υπολογιστικό κόστος και χρόνος εκπαίδευσης
- Αστάθεια κατά την εκπαίδευση (variance)
- Απαιτεί προσεκτικό σχεδιασμό ανταμοιβής και παραμέτρων

2.5 Εκπαίδευση Νευρωνικών Δικτύων

Η εκπαίδευση ενός νευρωνικού δικτύου συνίσταται στη βελτιστοποίηση των βαρών του δικτύου με στόχο τη βελτίωση της ακρίβειας πρόβλεψης. Η διαδικασία βασίζεται στην αρχή της **ελαχιστοποίησης μιας συνάρτησης κόστους**, η οποία εκφράζει τη διαφορά μεταξύ της πραγματικής και της προβλεπόμενης τιμής.

1. Συνάρτηση Κόστους (Loss Function)

Η συνάρτηση κόστους μετρά το «λάθος» του δικτύου. Συνήθεις επιλογές είναι:

- **Mean Squared Error (MSE):**

$$\mathcal{L}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- **Cross-Entropy Loss (για κατηγοριοποίηση):**

$$\mathcal{L}(y, \hat{y}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i)$$

2. Backpropagation

Η **αντίστροφη διάδοση σφάλματος (backpropagation)** είναι η διαδικασία με την οποία υπολογίζονται οι παράγωγοι του σφάλματος ως προς τα βάρη και ενημερώνονται τα βάρη μέσω gradient descent [38].

$$w_{ij} \leftarrow w_{ij} - \eta \cdot \frac{\partial \mathcal{L}}{\partial w_{ij}}$$

Όπου η είναι ο ρυθμός εκμάθησης (learning rate), μια κρίσιμη υπερπαράμετρος που καθορίζει το μέγεθος των βημάτων προσαρμογής. Η διαδικασία εφαρμόζεται επαναληπτικά για πολλές **εποχές (epochs)**, δηλαδή πλήρη περάσματα μέσα από ολόκληρο το σύνολο δεδομένων εκπαίδευσης. Για λόγους υπολογιστικής αποδοτικότητας και σταθερότητας, η ενημέρωση των βαρών συνήθως γίνεται ανά **παρτίδες (batches)** ή **μικρο-παρτίδες (mini-batches)** δεδομένων, αντί για ένα δείγμα τη φορά (online learning) ή ολόκληρο το dataset (batch learning).

3. Αλγόριθμοι Βελτιστοποίησης

Οι πιο διαδεδομένοι αλγόριθμοι βελτιστοποίησης είναι:

- **Stochastic Gradient Descent (SGD)**
- **Momentum:** προσθέτει αδράνεια στην κατεύθυνση βελτιστοποίησης

- **Adam (Adaptive Moment Estimation):** συνδυάζει RMSprop + Momentum
- **RMSprop:** προσαρμόζει το learning rate ανά βάρος

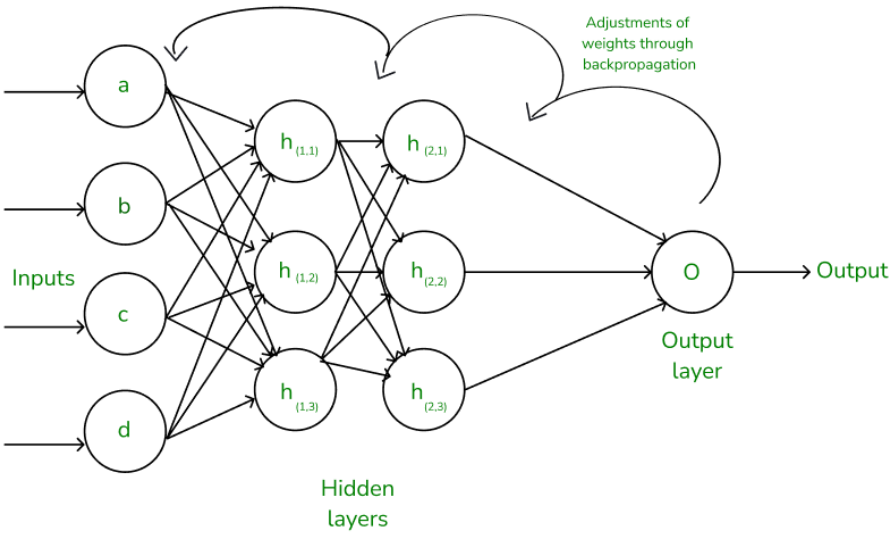
4. Τεχνικές Κανονικοποίησης (Regularization)

Για την αποφυγή υπερπροσαρμογής (overfitting), εφαρμόζονται τεχνικές όπως:

- **Dropout:** τυχαία μη ενεργοποίηση νευρώνων κατά την εκπαίδευση
- **Batch Normalization:** ομαλοποίηση ενεργοποιήσεων ανά batch
- L_1, L_2 **Regularization:** προσθήκη ποινής στο loss για μεγάλα βάρη

5. Αρχικοποίηση Βαρών (Weight Initialization)

Η αρχική τιμή των βαρών πριν την έναρξη της εκπαίδευσης μπορεί να επηρεάσει σημαντικά την ταχύτητα σύγκλισης και την τελική απόδοση του μοντέλου. Απλή αρχικοποίηση σε μηδέν ή τυχαίες τιμές από ομοιόμορφη κατανομή μπορεί να οδηγήσει σε προβλήματα (π.χ., συμμετρική ενημέρωση νευρώνων, vanishing/exploding gradients). Χρησιμοποιούνται πιο εξελιγμένες τεχνικές όπως η αρχικοποίηση Xavier/Glorot ή He, οι οποίες λαμβάνουν υπόψη τον αριθμό των εισερχόμενων και εξερχόμενων συνδέσεων κάθε νευρώνα για να διατηρήσουν τη διασπορά των ενεργοποιήσεων σταθερή κατά την προπαγάνωση μέσα στο δίκτυο.



Σχήμα 2.8: Απλοποιημένη απεικόνιση της διαδικασίας backpropagation [5]

Η επιλογή των παραμέτρων (optimizer, learning rate, regularization) είναι κρίσιμη για την επιτυχία του μοντέλου και καθορίζει τη δυνατότητα γενίκευσης σε μη ορατά δεδομένα.

2.6 Προκλήσεις και Περιορισμοί των Νευρωνικών Δικτύων

Παρόλο που τα νευρωνικά δίκτυα έχουν οδηγήσει σε θεαματικά αποτελέσματα σε πολλούς τομείς της τεχνητής νοημοσύνης, δεν είναι πανάκεια. Υπάρχουν αρκετές προκλήσεις και περιορισμοί που πρέπει να λαμβάνονται υπόψη κατά τον σχεδιασμό και την εφαρμογή τους.

1. Υπερπροσαρμογή (Overfitting)

Ένα από τα συχνότερα προβλήματα είναι η υπερπροσαρμογή, κατά την οποία το δίκτυο ”μαθαίνει” υπερβολικά καλά τα δεδομένα εκπαίδευσης, αλλά αποτυγχάνει να γενικεύσει σε νέα, άγνωστα δεδομένα. Το πρόβλημα αυτό επιδεινώνεται όταν τα δεδομένα εκπαίδευσης είναι λίγα ή περιέχουν θόρυβο.

Αντιμετώπιση:

- Dropout, early stopping, data augmentation, regularization

2. Υπολογιστική Πολυπλοκότητα

Η εκπαίδευση βαθιών νευρωνικών δικτύων απαιτεί σημαντική υπολογιστική ισχύ, ειδικά για μεγάλα σύνολα δεδομένων. Η ανάγκη για GPUs ή TPUs αποτελεί σημαντικό εμπόδιο για πολλούς ερευνητές ή οργανισμούς με περιορισμένους πόρους.

3. Έλλειψη Διαφάνειας – Το «Μαύρο Κουτί»

Τα νευρωνικά δίκτυα συχνά θεωρούνται «μαύρα κουτιά» (black boxes), καθώς είναι δύσκολο να εξηγηθεί με σαφήνεια πώς καταλήγουν σε μια απόφαση. Αυτό δημιουργεί προβλήματα σε ευαίσθητες εφαρμογές (π.χ. υγεία, χρηματοοικονομικά).

4. Ανάγκη για Μεγάλα Σύνολα Δεδομένων

Η επιτυχία των περισσότερων deep learning μοντέλων βασίζεται στην ύπαρξη μεγάλου όγκου δεδομένων. Σε περιβάλλοντα όπου τα δεδομένα είναι περιορισμένα ή μη προσβάσιμα, η απόδοση των δικτύων μπορεί να υποβαθμιστεί σημαντικά.

5. Ευαισθησία σε Αστοχίες και Εξωτερικούς Παράγοντες

Τα δίκτυα είναι συχνά ευάλωτα σε λεγόμενες *adversarial attacks*, δηλαδή μικρές τροποποιήσεις στα δεδομένα εισόδου που οδηγούν σε εντελώς λανθασμένες προβλέψεις [45].

6. Ηθικά και Κοινωνικά Ζητήματα

Η χρήση νευρωνικών δικτύων εγείρει ηθικά ζητήματα, όπως προκαταλήψεις στα δεδομένα (bias) που μπορούν να οδηγήσουν σε αλγοριθμική διάκριση, ζητήματα ιδιωτικότητας, και πιθανές επιπτώσεις στην

απασχόληση λόγω αυτοματοποίησης. Η ανάγκη για **εξηγήσιμη τεχνητή νοημοσύνη (Explainable AI - XAI)** είναι σήμερα πιο έντονη από ποτέ, με την έρευνα να εστιάζει σε τεχνικές (όπως LIME, SHAP) που μπορούν να φωτίσουν τη διαδικασία λήψης αποφάσεων των ’μαύρων κουτιών’, αυξάνοντας τη διαφάνεια και την εμπιστοσύνη σε κρίσιμες εφαρμογές.

Κεφάλαιο 3ο: Βιβλιογραφική Ανασκόπηση

3.1 Πρώιμες Προσεγγίσεις (1980–1999)

Κατά τις δεκαετίες του 1980 και 1990, η χρήση νευρωνικών δικτύων για την επίλυση του Προβλήματος του Περιοδεύοντος Πωλητή (TSP) ξεκίνησε με έμφαση σε βιολογικά εμπνευσμένες αρχιτεκτονικές. Οι σημαντικότερες από αυτές ήταν τα **Hopfield Networks** και τα **Self-Organizing Maps (SOM)**, οι οποίες επιχειρούσαν να μοντελοποιήσουν την αναζήτηση της βέλτιστης διαδρομής μέσω ενεργειακών ή γεωμετρικών μεθόδων.

Hopfield Networks – Ενεργειακή Μοντελοποίηση του TSP

Το 1985, οι Tank και Hopfield [46] πρότειναν ένα δυναμικό νευρωνικό μοντέλο όπου η επίλυση του TSP διατυπώνεται ως πρόβλημα ελαχιστοποίησης μιας **συνάρτησης ενέργειας** E. Το δίκτυο αποτελείται από n^2 νευρώνες, που αντιπροσωπεύουν την επίσκεψη της πόλης i στη θέση t της διαδρομής. Η κατάσταση του συστήματος δίνεται από τον πίνακα ενεργοποιήσεων $x_{i,t}$:

$$x_{i,t} = \begin{cases} 1, & \text{αν η πόλη } i \text{ βρίσκεται στη θέση } t \\ 0, & \text{διαφορετικά} \end{cases}$$

Η συνολική ενέργεια ορίζεται ως:

$$E = A \cdot E_1 + B \cdot E_2 + C \cdot E_3 + D \cdot E_4$$

όπου:

- E_1: κάθε πόλη εμφανίζεται μία φορά,
- E_2: κάθε θέση καταλαμβάνεται από μία πόλη,
- E_3: περιορισμός του συνολικού αριθμού ενεργών νευρώνων,
- E_4: συνολικό μήκος διαδρομής.

Το δίκτυο προσαρμόζεται δυναμικά ώστε να μειώσει το E με στόχο τη σταθεροποίηση σε έγκυρες και βέλτιστες λύσεις TSP. Ωστόσο, το μοντέλο έχει περιορισμούς στην κλιμάκωση και συχνά παγιδεύεται σε τοπικά ελάχιστα.

Self-Organizing Maps – Γεωμετρική Προσέγγιση

Η μέθοδος SOM, που προτάθηκε από τον Kohonen [19], αποτελεί μια μη εποπτευόμενη τεχνική εκμάθησης. Οι Fort (1988) [8] εφάρμοσαν SOM για το TSP οργανώνοντας τους νευρώνες κυκλικά ώστε

να προσεγγίζουν το γεωμετρικό σχήμα των πόλεων. Η βασική ιδέα είναι ότι, μέσω επαναληπτικής εκπαίδευσης, οι θέσεις των νευρώνων ”τυλίγονται” γύρω από τα σημεία-πόλεις, δημιουργώντας έτσι μια διαδρομή.

Το SOM παρουσιάζει καλή συμπεριφορά για προβλήματα μεσαίας κλίμακας ($n \leq 50$), είναι γρήγορο και δεν απαιτεί ετικέτες (labels), αλλά δεν εγγυάται βέλτιστες λύσεις.

Chaotic Neural Networks – Χάος και Εξερεύνηση

Οι Tokuda et al. (1997) [47] εισήγαγαν τα Chaotic Neural Networks ως μέθοδο αποφυγής εγκλωβισμού σε τοπικά ελάχιστα. Μέσω προσομοίωσης χαοτικής δυναμικής, το δίκτυο μπορεί να ξεφεύγει από μη επιθυμητές καταστάσεις, διατηρώντας ταυτόχρονα τη δυνατότητα σύγκλισης όταν η ενέργεια μειώνεται. Η χρήση μεταβαλλόμενης θερμοκρασίας επιτρέπει εναλλαγή μεταξύ εξερεύνησης και εκμετάλλευσης, παρόμοια με την προσομοιωμένη ανόπτηση (Simulated Annealing).

Hysteresis Neural Networks – Νευρώνες με Μνήμη Κατάστασης

Το 2000, οι Nakaguchi et al. [34] πρότειναν τα δίκτυα υστέρησης για την επίλυση του TSP. Η βασική ιδέα είναι η ενσωμάτωση ιστορικού στις ενεργοποιήσεις των νευρώνων. Ο νευρώνας ενεργοποιείται ή απενεργοποιείται ανάλογα όχι μόνο με την τρέχουσα είσοδο αλλά και με την προηγούμενη κατάστασή του. Αυτό προσδίδει στο δίκτυο καλύτερη σταθερότητα και μειωμένες ταλαντώσεις, οδηγώντας σε πιο αξιόπιστη σύγκλιση.

Σύνοψη Περιόδου 1980–1999

Οι παραπάνω προσεγγίσεις αποτέλεσαν τις πρώτες προσπάθειες εφαρμογής νευρωνικών μοντέλων στο TSP. Παρότι δεν ήταν κλιμακώσιμες ή γενικεύσιμες, εισήγαγαν σημαντικές ιδέες:

- χρήση ενεργειακής δυναμικής (Hopfield),
- χαρτογραφικές αυτοοργανωτικές τεχνικές (SOM),
- μοντέλα με ενσωματωμένο ”χάος” (chaotic NN),
- χρήση μνήμης κατάστασης (hysteresis).

Αυτές οι ιδέες αποτέλεσαν το θεωρητικό υπόβαθρο πάνω στο οποίο βασίστηκαν μεταγενέστερες εξελίξεις στη βαθιά μάθηση και τα γραφικά νευρωνικά δίκτυα.

3.2 Κατηγοριοποίηση Αλγορίθμων με Βάση την Τεχνική

Μετά τις πρώιμες προσεγγίσεις της δεκαετίας του 1980 και 1990, η εξέλιξη των Νευρωνικών Δικτύων για το TSP διαφοροποιήθηκε κυρίως ως προς την τεχνική επίλυσης και την αρχιτεκτονική των μοντέλων. Οι κυριότερες κατηγορίες περιλαμβάνουν:

- **Βαθιά Νευρωνικά Δίκτυα (Deep Neural Networks)**, όπως τα Pointer Networks, Transformers και DRL πολιτικές.
- **Γραφικά Νευρωνικά Δίκτυα (Graph Neural Networks)**, με παραδείγματα όπως GCN, BiGNN, GNN-VNS.
- **Υβριδικές και Εξελιγμένες Προσεγγίσεις**, που συνδυάζουν τεχνικές (όπως Hopfield+Bio, GNN+VNS, ACO+RNN κ.ά.).
- **Ιστορικά και Προσαρμοστικά Μοντέλα**, που παραμένουν σημαντικά είτε ως απλά benchmarks είτε ως inspiration για σύγχρονες παραλλαγές.

Η παρούσα ενότητα αναλύει αυτές τις τεχνικές κατηγορίες αναλυτικά, παρουσιάζοντας τα βασικά μοντέλα, τον μηχανισμό λειτουργίας τους, τις κυριότερες βελτιώσεις που προσφέρουν σε σχέση με τα προηγούμενα, καθώς και τη συνεισφορά τους στην επίλυση του TSP.

Κάθε υποενότητα περιλαμβάνει:

- Τεχνική περιγραφή της αρχιτεκτονικής
- Σύντομη ιστορική αναφορά
- Πλεονεκτήματα και περιορισμοί
- Ενδεικτικούς ψευδοκώδικες και μηχανισμούς λειτουργίας

Η θεματική αυτή κατηγοριοποίηση επιτρέπει μια πιο λειτουργική σύγκριση και προετοιμάζει το έδαφος για την πρακτική εφαρμογή αλγορίθμων που ακολουθεί στα επόμενα κεφάλαια.

1. Βαθιά Νευρωνικά Δίκτυα (Deep Neural Networks)

Η κατηγορία των βαθιών νευρωνικών δικτύων (Deep Neural Networks – DNNs) περιλαμβάνει μοντέλα που εκμεταλλεύονται βαθιά αρχιτεκτονική πολλών επιπέδων για να ”μάθουν” την επίλυση του TSP. Η εξέλιξη ξεκίνησε με τα **Pointer Networks** και συνέχισε με ισχυρότερες πολιτικές **ενισχυτικής μάθησης** (Reinforcement Learning), ενώ τα τελευταία χρόνια έχει δοθεί έμφαση στους **Transformers** και τα **POMO**-based συστήματα.

Pointer Networks – Vinyals et al. (2015) Οι Pointer Networks [49] ήταν η πρώτη αρχιτεκτονική που μπορούσε να διαχειριστεί προβλήματα μεταβλητού μήκους όπως το TSP. Αντί να προβλέπουν ”λέξεις”, τα Pointer Networks μαθαίνουν να επιλέγουν στοιχεία της εισόδου ως έξοδο (π.χ., τη σειρά των πόλεων). Η αρχιτεκτονική βασίζεται σε ένα μηχανισμό **attention** πάνω από την είσοδο.

Κύρια χαρακτηριστικά:

- LSTM Encoder για επεξεργασία των συντεταγμένων των πόλεων.

- LSTM Decoder που παράγει δείκτες στις εισόδους (τις πόλεις).
- Attention μηχανισμός που μαθαίνει να επιλέγει την ”επόμενη πόλη”.

Πλεονεκτήματα: End-to-end εκπαίδευση χωρίς εξωτερικούς αλγορίθμους. **Περιορισμοί:** Απόδοση μειώνεται αισθητά για μεγάλες n (π.χ. $n > 100$).

Deep Reinforcement Learning – Bello et al. (2017), Kool et al. (2019) Η ενισχυτική μάθηση (Reinforcement Learning – RL) εφαρμόστηκε στο TSP ώστε ο πράκτορας (agent) να μαθαίνει την επίλυση του προβλήματος μόνο από το σήμα ανταμοιβής (reward) — τη συνολική απόσταση της διαδρομής. Ο Bello et al. χρησιμοποίησαν policy gradient με Pointer Networks [4], ενώ οι Kool et al. εφάρμοσαν μηχανισμό attention μόνο (χωρίς RNNs) [21].

Κύρια χαρακτηριστικά:

- Χρήση REINFORCE ή Actor-Critic για εκπαίδευση χωρίς supervision.
- Sampling πολλών διαδρομών και ενίσχυση των καλών.
- Ενσωμάτωση attention για ταχύτητα και ακρίβεια.

Πλεονεκτήματα: Δεν απαιτεί έτοιμες βέλτιστες λύσεις για εκπαίδευση. **Περιορισμοί:** Υψηλό variance – χρειάζεται αρκετά επεισόδια για σταθερή μάθηση.

Transformers για TSP – Kool et al. (2019) Οι Transformers επιτρέπουν πλήρως παράλληλη επεξεργασία και μακροχρόνιες εξαρτήσεις. Στο TSP, τα μοντέλα transformer-based (όπως του Kool) αντικαθιστούν τα LSTM με self-attention layers, πετυχαίνοντας καλύτερη απόδοση και ταχύτερη εκπαίδευση.

Μηχανισμός:

- Πόλεις $\rightarrow$ ενσωμάτωση θέσης + encoding.
- Stack από attention layers για εκμάθηση δομής διαδρομής.
- Autoregressive decoding για παραγωγή της λύσης.

Πλεονεκτήματα: Γενικεύει σε μεγαλύτερα προβλήματα, κλιμακώνεται καλύτερα. **Περιορισμοί:** Υψηλότερη πολυπλοκότητα σε training/inference από Pointer Nets.

POMO – Policy Optimization with Multiple Optima – Kwon et al. (2020) Το POMO [24] είναι ένα μοντέλο βαθιάς ενισχυτικής μάθησης που αποφεύγει τη χρήση επιπλέον critics ή variance reduction τεχνικών, ξεκινώντας από πολλαπλές ισοδύναμες λύσεις. Έτσι ενσωματώνει την ιδέα της συμμετρίας του TSP (πολλές ισοδύναμες λύσεις) στην ίδια την εκπαίδευση.

Κύρια σημεία:

- Πολλαπλές αρχικές διαδρομές ανά επεισόδιο.
- Χρήση attention-based pointer decoding όπως στους Transformers.
- Εκπαίδευση με REINFORCE και ενίσχυση της καλύτερης εκδοχής.

Πλεονεκτήματα: Σταθερή εκπαίδευση, πολύ καλή ακρίβεια. **Περιορισμοί:** Πιο απαιτητικό υπολογιστικά κατά το training.

Σύνοψη Η κατηγορία των Deep Neural Networks αποτέλεσε το πρώτο κύμα ”έξυπνης” εκμάθησης της επίλυσης του TSP. Τα Pointer Networks έδωσαν την αρχική ώθηση, τα RL-based μοντέλα αφαίρεσαν την ανάγκη εποπτείας, ενώ τα Transformers και το POMO προσέφεραν ισχυρή απόδοση και σταθερότητα.

Στα επόμενα υποκεφάλαια εξετάζονται τα γραφικά δίκτυα, τα οποία προσφέρουν φυσική μοντελοποίηση των προβλημάτων με γράφους, όπως το TSP.

2. Γραφικά Νευρωνικά Δίκτυα (Graph Neural Networks)

Η φύση του προβλήματος TSP είναι εγγενώς γραφική, καθώς οι πόλεις μπορούν να αναπαρασταθούν ως κόμβοι και οι αποστάσεις μεταξύ τους ως ακμές. Τα **Graph Neural Networks (GNNs)** εκμεταλλεύονται αυτή τη δομή, μαθαίνοντας representations τόσο για τους κόμβους όσο και για τις ακμές μέσω μηνυμάτων και συσσωρεύσεων (message passing).

GCN – Graph Convolutional Networks – Kipf & Welling (2017) Οι Kipf και Welling [18] εισήγαγαν τα Graph Convolutional Networks (GCNs), τα οποία επιτρέπουν τη μάθηση χαρακτηριστικών κόμβων μέσω ”συσσωρευμένης πληροφορίας” από τους γειτονικούς κόμβους. Αν και αρχικά σχεδιάστηκαν για ημι-εποπτευόμενη μάθηση, σύντομα χρησιμοποιήθηκαν για combinatorial προβλήματα όπως το TSP.

Βασικές αρχές:

- Κάθε κόμβος v έχει χαρακτηριστικά x_v.
- Σε κάθε επίπεδο, τα νέα χαρακτηριστικά x'_v εξαρτώνται από γείτονες:

$$x'_v = \sigma\left(\sum_{u \in \mathcal{N}(v)} \frac{1}{\sqrt{d_v d_u}} W x_u\right)$$

- Η έξοδος χρησιμοποιείται για απόφαση περιήγησης ή επιλογή ακμών.

GNN για TSP – Joshi et al. (2019) Ο Joshi et al. [16] ανέπτυξαν ένα GNN για την πρόβλεψη των επομένων κινήσεων στο TSP. Η είσοδος είναι ένα γράφημα (πόλεις + αποστάσεις) και το δίκτυο παράγει είτε διαδρομή είτε πιθανότητες επιλογής.

Σημεία μεθόδου:

- Χρήση encoder με message passing layers.
- Decoder που χτίζει σταδιακά την διαδρομή μέσω attention ή pointer.
- Επιβλεπόμενη ή ενισχυτική εκπαίδευση (RL).

Πλεονεκτήματα: Πολύ καλή γενίκευση, ικανότητα αναπαράστασης δομής προβλήματος. **Περιορισμοί:** Περιορισμένη ικανότητα ολικής βελτιστοποίησης χωρίς post-processing.

BiGNN – Liang et al. (2024) Το BiGNN [27] είναι ένα bipartite Graph Neural Network μοντέλο με attention μηχανισμό για επίλυση **πολλαπλών TSP** (mTSP) σε ρυθμίσεις logistics. Το γράφημα χωρίζεται σε δύο ομάδες: πόλεις και agents (πωλητές, οχήματα).

Κύρια χαρακτηριστικά:

- Χρήση bipartite graph: $G = (V_{cities}, V_{agents}, E)$.
- Attention μεταξύ πόλεων και agents για assignment + routing.
- Εκπαίδευση με στόχο ελαχιστοποίηση συνολικής απόστασης και ισορροπία μεταξύ agents.

Πλεονεκτήματα: Μπορεί να χειριστεί mTSP με πολλές επιμέρους διαδρομές. **Περιορισμοί:** Περίπλοκη υλοποίηση – απαιτεί καλά ισορροπημένα datasets.

GNN-VNS – Sohiburoyyan et al. (2024) Το GNN-VNS [43] είναι μια υβριδική προσέγγιση που συνδυάζει GNN με Variable Neighborhood Search (VNS). Το GNN δημιουργεί αρχικές λύσεις και το VNS τις βελτιστοποιεί μέσω τοπικών τροποποιήσεων.

Δομή μεθόδου:

- Graph encoder (GNN) εξάγει embeddings κόμβων και ακμών.
- Αρχική διαδρομή προτείνεται μέσω pointer ή greedy policy.
- VNS εφαρμόζει 2-opt, 3-opt ή swap neighborhoods.

Πλεονεκτήματα: Συνδυάζει μαθαίνοντα module (GNN) με κλασική μεταευρετική αναζήτηση. **Περιορισμοί:** Δεν είναι πλήρως end-to-end – η VNS λειτουργεί εξωτερικά.

Neuro-Ising GNNs – Sanyal et al. (2025) Το μοντέλο NeuroIsing [39] συνδυάζει GNNs με αναλογίες από το μοντέλο Ising της στατιστικής φυσικής. Οι κόμβοι μοντελοποιούνται ως ”spins”, και οι διασυνδέσεις εκφράζουν ενεργειακή αλληλεπίδραση.

Κύρια καινοτομία:

- Ενεργειακή δυναμική οδηγεί τη διαδρομή προς ελάχιστο κόστους.
- Το GNN μαθαίνει τον ρυθμό μετάβασης κατάστασης σε κάθε κόμβο.
- Αξιοποιούνται φυσικά φαινόμενα (μεταφάσεις, μαγνητισμός) ως inspiration.

Πλεονεκτήματα: Ικανό να χειριστεί πολύπλοκα cost landscapes, σταθερό σε μεγάλα n. **Περιορισμοί:** Υψηλό κόστος εκπαίδευσης, απαιτεί φυσική προσομοίωση ή approximation.

Σύνοψη Τα GNNs αποτελούν σήμερα τον κορμό πολλών state-of-the-art προσεγγίσεων για το TSP. Η γραφική αναπαράσταση προσφέρει:

- φυσική συμβατότητα με τη δομή του προβλήματος,
- ισχυρή γενίκευση σε παραλλαγές (mTSP, VRP, CVRP),
- δυνατότητα ενοποίησης με attention, RL ή ευρετικές.

Οι προσεγγίσεις αυτές αντιπροσωπεύουν το παρόν και το άμεσο μέλλον της έρευνας, τόσο για θεωρητική ανάλυση όσο και για εφαρμογές σε πραγματικά προβλήματα logistics.

3. Υβριδικές και Εξελιγμένες Προσεγγίσεις

Καθώς οι καθαρά νευρωνικές λύσεις παρουσιάζουν περιορισμούς είτε στην ακρίβεια είτε στην επεκτασιμότητα, αναδύθηκε μια νέα κατηγορία προσεγγίσεων που συνδυάζουν νευρωνικά δίκτυα με άλλες τεχνικές βελτιστοποίησης, βιολογικά εμπνευσμένα μοντέλα, και συστήματα μάθησης με meta-προσαρμογή. Οι υβριδικές μέθοδοι αξιοποιούν τα πλεονεκτήματα διαφορετικών paradigms για να προσφέρουν αυξημένη σταθερότητα και γενίκευση.

SOM + Metaheuristics (GA, SA, PSO) Κατά την περίοδο 2005–2015, εμφανίστηκαν πολλές εργασίες που συνδύασαν Self-Organizing Maps (SOM) με μεταευρετικούς αλγορίθμους όπως Genetic Algorithms (GA), Simulated Annealing (SA), και Particle Swarm Optimization (PSO) [**?**, **?**]. Οι SOM προσφέρουν μια γρήγορη, περίπου αποδεκτή διαδρομή, την οποία οι metaheuristics βελτιώνουν μέσω εξελικτικών μετασχηματισμών.

Δομή:

- SOM παράγει αρχική διαδρομή.
- Η metaheuristic τη βελτιστοποιεί με επαναληπτική αναζήτηση.
- Οι fitness functions προσαρμόζονται με βάση την απόσταση και την πολυπλοκότητα.

Πλεονεκτήματα: Σταθερές λύσεις, καλή επεκτασιμότητα. **Περιορισμοί:** Δεν είναι end-to-end, απαιτεί δύο φάσεις.

Physarum-Inspired Hopfield Networks Το μοντέλο αυτό [3] εμπνέεται από τους βιολογικούς μηχανισμούς των πλασμωδών οργανισμών Physarum polycephalum, οι οποίοι βρίσκουν διαδρομές μέσα από φυσική διάχυση. Συνδυάζονται με Hopfield δίκτυα για να δώσουν ένα αναλογικό και χαμηλής ενέργειας σχήμα επίλυσης.

Κύρια χαρακτηριστικά:

- Αναπαράσταση των πόλεων ως φωτοευαίσθητα σημεία.
- Ηλεκτρικά ή φωτεινά σήματα ενισχύουν επιθυμητές διαδρομές.
- Το Hopfield δίκτυο ενισχύει τη σταθερότητα των διαδρομών.

Πλεονεκτήματα: Βιολογικά συμβατή επίλυση, ιδανική για hardware εφαρμογές. **Περιορισμοί:** Περιορισμένη πρακτική εφαρμογή εκτός φυσικών ή προσομοιωμένων συστημάτων.

Q-Learning με Τοπική Δυναμική Προσαρμογή (Q+LDP) Η εργασία [52] συνδυάζει Q-Learning με τοπικές τροποποιήσεις τύπου 2-opt και greedy rewiring. Το δίκτυο επεξεργάζεται εικόνα της διάταξης πόλεων και αποφασίζει τη βελτίωση της διαδρομής.

Δομή μεθόδου:

- Ο Agent αναπαριστά επιλογές swap/insert/remove.
- Το state είναι χωρική απεικόνιση (εικόνα με πόλεις και τρέχουσα διαδρομή).
- Reward βασίζεται σε μείωση μήκους διαδρομής και επιβράβευση μετά από πολλές βελτιώσεις.

Πλεονεκτήματα: Μπορεί να εισαχθεί σε οποιοδήποτε άλλο σύστημα ως βελτιστοποιητής. **Περιορισμοί:** Υψηλή πολυπλοκότητα αναπαράστασης κατά inference.

Hybrid RNN + Attention + ACO Το μοντέλο HybridRNN [42] συνδυάζει αλγορίθμους αποικίας μυρμηγκιών (Ant Colony Optimization – ACO) με ακολουθιακά νευρωνικά δίκτυα τύπου RNN και μηχανισμούς attention. Το ACO προτείνει πιθανές κινήσεις, ενώ το RNN ενσωματώνει το ιστορικό διαδρομής και το attention μηχανισμός δίνει δυναμικά βάρη στις πόλεις.

Δομή:

- Το RNN διατηρεί χρονική μνήμη της διαδρομής.
- Το attention επικεντρώνεται σε κρίσιμες περιοχές του γράφου.
- Το ACO μετατρέπει τις πιθανότητες σε λύση με βάση φερομόνες και bias.

Πλεονεκτήματα: Πλούσια contextual κατανόηση της διαδρομής. **Περιορισμοί:** Χρονοβόρα εκπαίδευση, πολλαπλές μονάδες συνεργάζονται.

Meta-Learning Solvers & GNN-based Selector Σε πολύ πρόσφατες εργασίες [57], GNNs χρησιμοποιούνται όχι μόνο για την πρόβλεψη της διαδρομής, αλλά και για:

- **Πρόβλεψη δυσκολίας στιγμιότυπου (instance difficulty)**
- **Επιλογή κατάλληλου αλγορίθμου από χαρτοφυλάκιο (portfolio selection)**

Αυτό μετατρέπει το TSP σε μετα-πρόβλημα, επιτρέποντας ταχεία εναλλαγή μοντέλων ανάλογα με τα χαρακτηριστικά εισόδου.

Πλεονεκτήματα: Υψηλή αποδοτικότητα σε μεγάλους στόλους αλγορίθμων. **Περιορισμοί:** Απαιτεί μεγάλο training dataset πολλών διαφορετικών TSP μορφών.

Σύνοψη Οι υβριδικές και εξελιγμένες προσεγγίσεις προσφέρουν:

- Δυναμική συνδυαστικότητα τεχνικών (βιολογικών, μαθησιακών, ευρετικών)
- Επαυξημένη σταθερότητα έναντι των end-to-end δικτύων
- Νέες μετα-προοπτικές όπως algorithm selection και epigenetic learning

Αν και συχνά πιο δύσκολες στην υλοποίηση, οι μέθοδοι αυτές προσεγγίζουν το TSP με σκοπό όχι μόνο την επίλυση, αλλά και την κατανόηση του πλαισίου στο οποίο εντάσσεται κάθε στιγμιότυπο.

4. Σύνοψη και Συγκριτική Ανάλυση

Από τις πρώτες ενεργειακές προσεγγίσεις του Hopfield μέχρι τα σύγχρονα γραφικά και attention-based μοντέλα, η επίλυση του TSP με νευρωνικά δίκτυα έχει εξελιχθεί σε ένα πεδίο πολυδιάστατης έρευνας και ανάπτυξης. Η παρούσα ενότητα συνοψίζει τις προσεγγίσεις ανά τεχνική, παραθέτοντας τα κυριότερα πλεονεκτήματα και περιορισμούς κάθε κατηγορίας.

Μέθοδος	**Αρχιτεκτονική**	**Εκπαίδευση**	**Κλίμακα**	**Γενίκευση**	**Πρακτικότητα**
Hopfield (1985)	Ενεργειακό μοντέλο	Unsupervised	Χαμηλή	Όχι	Περιορισμένη
SOM (1988)	Μη επιβλεπόμενο	Self-organized	Μεσαία	Όχι	Καλή
Chaotic NN (1997)	Δυναμική χαοτική	Unsupervised	Χαμηλή	Όχι	Μέτρια
Hysteresis NN (2000)	Ενεργειακό με μνήμη	Unsupervised	Μεσαία	Όχι	Καλή
MLP / MLP+GA (2007–2015)	Supervised DNN / Hybrid	Supervised / Hybrid	Μεσαία	Ναι	Καλή
Pointer Network (2015)	LSTM + Attention	Supervised	Μεσαία ($n \leq 100$)	Μερικώς	Καλή
DRL (2017–2019)	Policy Gradient / Actor–Critic	RL (Unsupervised)	Υψηλή	Ναι	Καλή
Transformer (2019+)	Attention-only	RL / Supervised	Υψηλή	Ναι	Πολύ Καλή
POMO (2020)	Multi-start + Attention	RL	Υψηλή	Ναι	Πολύ Καλή
GNN (2020+)	Message-passing	Supervised / RL	Πολύ Υψηλή	Ναι	Πολύ Καλή
BiGNN (2024)	Bipartite GNN	Supervised	mTSP	Ναι	Ισχυρή
GNN-VNS (2024)	Hybrid GNN + Heuristic	Mixed	Υψηλή	Ναι	Πολύ Ισχυρή
Physarum-Hopfield (2025)	Bio-optical neural nets	Analog-inspired	Χαμηλή	Όχι	Νεωτεριστική
Hybrid RNN + ACO (2025)	RNN + Attention + Ant Colony	RL / Evolution	Μεσαία–Υψηλή	Ναι	Δομικά Ισχυρή
Meta-GNN Selector (2025)	GNN for meta-analysis	Meta-learning	Υψηλή	Ναι	Προηγμένη

Πίνακας 3.1: Συγκριτικός πίνακας τεχνικών επίλυσης του TSP με Νευρωνικά Δίκτυα

Γενικά Συμπεράσματα:

- Οι **ενεργειακές μέθοδοι** όπως τα Hopfield, Hysteresis και Chaotic NN εισήγαγαν σημαντικές θεωρητικές έννοιες, αλλά έχουν χαμηλή επεκτασιμότητα.
- Τα **κλασικά δίκτυα** (MLP, SOM) προσφέρουν καλή απόδοση σε μικρές περιπτώσεις, ειδικά όταν συνδυάζονται με ευρετικές.
- Τα **Deep Models** (Pointer, DRL, Transformers, POMO) έφεραν επανάσταση μέσω end-to-end μάθησης.
- Τα **GNNs** προσφέρουν δομική ακρίβεια και εξαιρετική γενίκευση σε TSP και παραλλαγές του.
- Οι **υβριδικές προσεγγίσεις** αξιοποιούν τα καλύτερα χαρακτηριστικά από διαφορετικούς κόσμους (π.χ., GNN + VNS ή ACO + RNN).

Η μελλοντική έρευνα φαίνεται να συγκλίνει σε υβριδικά μοντέλα με meta-learning και attention-aware routing, καθώς και σε εφαρμογές που ενσωματώνουν πραγματικά δεδομένα από logistics και edge computing.

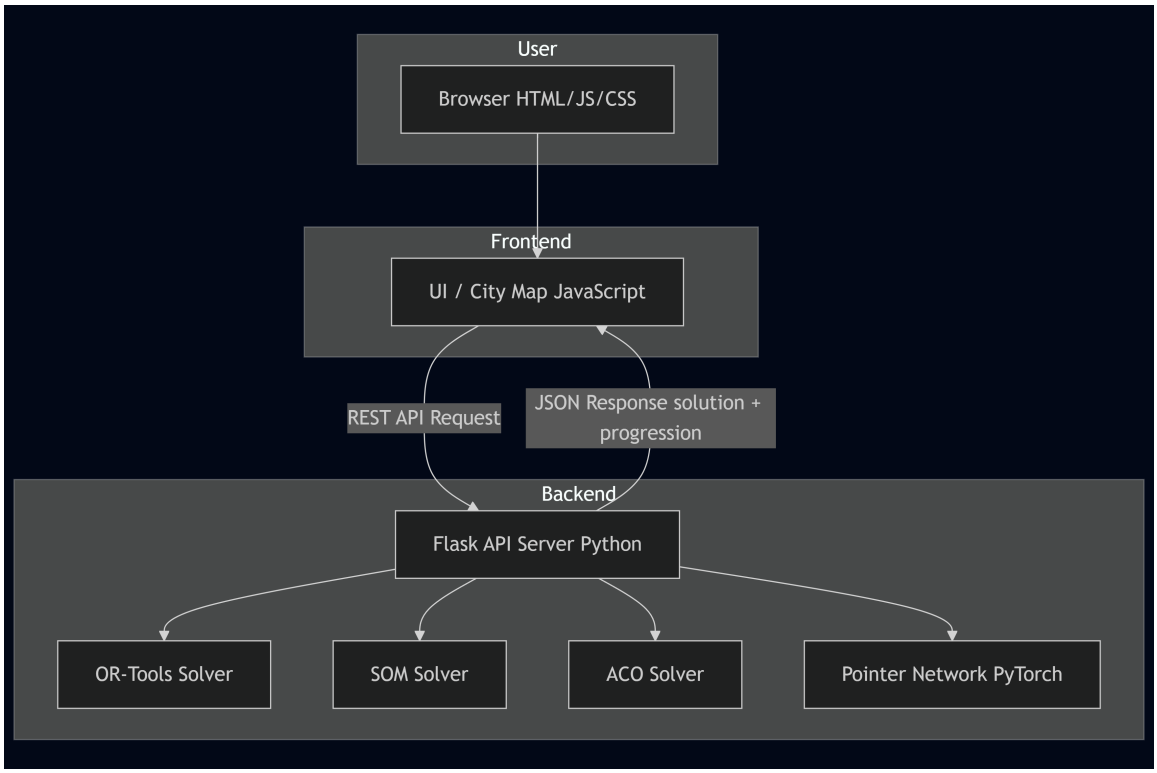
Κεφάλαιο 4ο: Υλοποίηση και Πειραματική Αξιολόγηση Αλγορίθμων για το TSP

Στο πρακτικό μέρος της παρούσας εργασίας αναπτύχθηκε μία ολοκληρωμένη διαδραστική πλατφόρμα για την επίλυση του προβλήματος του Πλανόδιου Πωλητή (TSP). Η υλοποίηση συνδυάζει τεχνικές από το πεδίο της επιχειρησιακής έρευνας, των βιο-εμπνευσμένων αλγορίθμων καθώς και της βαθιάς μάθησης. Ο στόχος ήταν όχι μόνο η επίτευξη αποτελεσματικών λύσεων, αλλά και η ενσωμάτωσή τους σε ένα περιβάλλον όπου μπορεί να παρατηρηθεί η **πρόοδος κάθε αλγορίθμου βήμα προς βήμα** μέσω animation και visualizations.

4.1 Γενική Αρχιτεκτονική

Η εφαρμογή ακολουθεί την αρχιτεκτονική Model-View-Controller (MVC) και αποτελείται από:

- **Backend Server**: Flask web framework με modular solver architecture
- **Frontend Interface**: HTML5, CSS3, JavaScript (ES6+) για interactive user experience
- **Mapping Engine**: Leaflet.js για διαδραστικούς χάρτες και γεωγραφική visualization
- **Data Layer**: CSV files για γεωγραφικά δεδομένα ελληνικών πόλεων
- **Visualization Engine**: Real-time algorithm progression και comparative analysis tools



Σχήμα 4.1: Αρχιτεκτονική Συστήματος [54]

4.2 Τεχνολογικές Επιλογές και Σχεδιαστικές Αποφάσεις

4.2.1 Backend Technologies

Η επιλογή του Flask ως core framework έγινε για την ευκολία ανάπτυξης και την ευελιξία του:

- **Flask**: Lightweight web framework για γρήγορη ανάπτυξη και HTTP API routing
- **Python Core**: Υλοποίηση αλγορίθμων και επεξεργασία γεωγραφικών δεδομένων
- **PyTorch**: Neural network framework για SOM implementation με GPU acceleration
- **NumPy/SciPy**: Μαθηματικές βιβλιοθήκες για αριθμητικούς υπολογισμούς και matrix operations
- **OR-Tools**: Google's optimization library για professional-grade TSP solving
- **Matplotlib**: Στατική visualization και export functionality
- **Pandas**: Επεξεργασία και ανάλυση CSV δεδομένων

4.2.2 Frontend Technologies

Το frontend υλοποιήθηκε χωρίς heavy frameworks για μέγιστη performance:

- **Leaflet.js**: Open-source mapping library για interactive geographical visualization με OpenStreetMap tiles [35]

- **Chart.js**: Advanced charting για statistical analysis και comparative metrics
- **Vanilla JavaScript**: Modern ES6+ features χωρίς framework dependencies
- **CSS Grid/Flexbox**: Responsive layout techniques για cross-platform compatibility
- **HTML5 Canvas**: High-performance rendering για algorithm animations

4.3 Δομή Δεδομένων και Data Management

Το σύστημα χρησιμοποιεί structured approach για την διαχείριση γεωγραφικών δεδομένων:

Πηγή Δεδομένων: SimpleMaps dataset με comprehensive coverage ελληνικών πόλεων, συμπεριλαμβανομένων γεωγραφικών συντεταγμένων, πληθυσμιακών στοιχείων και διοικητικής κατάταξης.

Data Processing Pipeline:

1. **CSV Parsing**: Φόρτωση και validation γεωγραφικών δεδομένων
2. **Intelligent Filtering**: Geographical distribution optimization για balanced city selection
3. **Coordinate Transformation**: Lat/lng σε καρτεσιανές συντεταγμένες για algorithmic processing
4. **Distance Matrix Generation**: Precomputed Euclidean distances για performance optimization

Smart City Selection Strategy: Το σύστημα εφαρμόζει intelligent filtering που εξασφαλίζει γεωγραφική κατανομή με προτεραιότητα σε νησιωτικές περιοχές, μεγάλα αστικά κέντρα και διοικητικές πρωτεύουσες για realistic TSP instances.

4.4 Backend Implementation και Modular Architecture

4.4.1 Flask Server Core

Ο κεντρικός server υλοποιεί standardized interface για όλους τους αλγορίθμους:

Solver Interface Standardization: Κάθε αλγόριθμος υλοποιεί common interface που επιστρέφει structured output με:

- Final solution path με city indices
- Total distance calculation
- Execution time metrics
- Algorithm-specific metadata
- Progressive animation data (όπου applicable)

API Endpoint Design:

- **/solve_tsp**: Single algorithm execution με parameter customization
- **/solve_tsp_all**: Comprehensive dual-run comparison για objective analysis
- **/generate_report**: Advanced statistical analysis με multi-scale testing

4.4.2 Dual-Run Comparison Framework

Το σύστημα εφαρμόζει sophisticated comparison methodology:

Φάση 1 - Natural Execution: Κάθε αλγόριθμος εκτελείται με τη δική του λογική αρχικής πόλης, επιδεικνύοντας τις φυσικές του προτιμήσεις.

Φάση 2 - Standardized Comparison: Όλοι οι αλγόριθμοι εκτελούνται από την ίδια αρχική πόλη (που επέλεξε ο SOM), εξασφαλίζοντας objective comparison.

Fairness Analysis: Η σύγκριση των δύο φάσεων αποκαλύπτει την επίδραση της αρχικής πόλης στην performance κάθε αλγορίθμου.

4.4.3 Error Handling και Robustness

Το σύστημα υλοποιεί comprehensive error management:

- **Graceful Degradation**: Συνέχιση εκτέλεσης άλλων αλγορίθμων όταν ένας αποτύχει
- **Detailed Error Reporting**: Comprehensive logging με stack traces
- **Input Validation**: Robust validation για city selection και parameters
- **Resource Management**: Memory cleanup και timeout handling

4.5 Frontend Architecture και User Experience

4.5.1 Modular JavaScript Architecture

Το frontend αποτελείται από πέντε κύρια modules με ξεχωριστές ευθύνες:

Module Organization:

- **map-core.js**: Core mapping functionality, city loading, και marker management
- **map-ui.js**: User interface controls, event handling, και modal management
- **map-solver.js**: Algorithm execution, animation control, και progression visualization
- **map-solutions.js**: Solution display, comparison tools, και statistical analysis
- **map-utils.js**: Utility functions, coordinate transformations, και helper methods

4.5.2 Interactive Mapping και Visualization

To mapping subsystem παρέχει advanced geographical interaction:

Core Mapping Features:

- **Dynamic Tile Layers**: Automatic switching μεταξύ light/dark themes
- **Interactive City Selection**: Click-based selection με popup interfaces
- **Pulse Animations**: Visual feedback για επιλεγμένες πόλεις
- **Responsive Design**: Cross-platform compatibility για desktop/mobile

Advanced Search Functionality: Implementation autocomplete search με Greek/English name support, geographic filtering, και immediate map navigation.

4.5.3 Algorithm Progression Visualization

Κάθε αλγόριθμος έχει specialized visualization approach:

SOM Visualization: Neural ring evolution με animated neurons, city connections, και training phase indicators.

ACO Visualization: Pheromone trail intensity mapping, ant journey tracking, και best path highlighting.

Pointer Network Visualization: Construction algorithm step-by-step building, insertion points, και multi-algorithm comparison.

OR-Tools Visualization: Professional-grade final result presentation με minimal overhead, emphasizing production-quality output.

4.6 Multi-Algorithm Comparison System

4.6.1 Horizontal Solutions Display

Το σύστημα εμφανίζει comprehensive comparison interface:

Dual-Run Layout:

- **Run 1 Section**: Natural execution results με algorithm-specific starting preferences
- **Run 2 Section**: Standardized comparison από common starting point
- **Performance Metrics**: Distance comparison, execution time analysis, και quality assessment

Mini-Map Grid: Κάθε algorithm result εμφανίζεται σε dedicated mini-map με synchronized styling και interactive features.

4.6.2 Statistical Analysis Tools

Advanced analytics για comprehensive performance evaluation:

- **Distance Distribution Charts**: Comparative analysis των solution qualities
- **Execution Time Comparison**: Performance benchmarking across algorithms
- **Starting City Impact Analysis**: Fairness assessment από dual-run comparison
- **Algorithm Ranking**: Dynamic ranking βάσει multiple criteria

4.7 Performance Optimizations και Technical Excellence

4.7.1 Frontend Performance

Rendering Optimizations:

- **Lazy Loading**: Mini-maps δημιουργούνται on-demand
- **Efficient Marker Management**: Memory optimization για large city datasets
- **Animation Frame Optimization**: Smooth animations με minimal CPU overhead
- **Memory Cleanup**: Automatic disposal των unused map instances

Responsive Design Implementation: CSS Grid και Flexbox για adaptive layouts που προσαρμόζονται σε desktop, tablet, και mobile devices.

4.7.2 Backend Performance

Asynchronous Processing:

- **Non-blocking Algorithm Execution**: Parallel processing για multi-algorithm runs
- **Efficient Data Structures**: Optimized για μεγάλα datasets
- **Memory Management**: Automatic cleanup και garbage collection hints
- **Caching Strategies**: Distance matrix caching για repeated executions

4.8 User Experience Features

4.8.1 Interactive Controls και Accessibility

Advanced UI Features:

- **Dark/Light Mode**: Complete theme switching με persistent preferences
- **City Search**: Intelligent autocomplete με Greek/English support
- **Export Capabilities**: Map και result export σε multiple formats
- **Animation Controls**: Play/pause/speed control για algorithm progression

Responsive Interface Design: Adaptive layout που παρέχει optimal experience σε όλες τις screen sizes, με intelligent hiding/showing των UI elements.

4.8.2 Export και Sharing Functionality

Comprehensive export system για research και presentation needs:

- **Map Export**: High-quality PNG export για main και mini-maps
- **Statistical Reports**: CSV export με detailed performance metrics
- **Comparative Analysis**: Multi-format reports για algorithm comparison
- **Animation Capture**: Optional gif export για algorithm progression

4.9 Professional Integration και Production Features

4.9.1 OR-Tools Integration

Multi-Strategy Approach: `GUIDED_LOCAL_SEARCH`, `PATH_CHEAPEST_ARC`, Christofides

4.9.2 Educational vs Professional Balance

Το σύστημα εξισορροπεί educational visualization με professional performance:

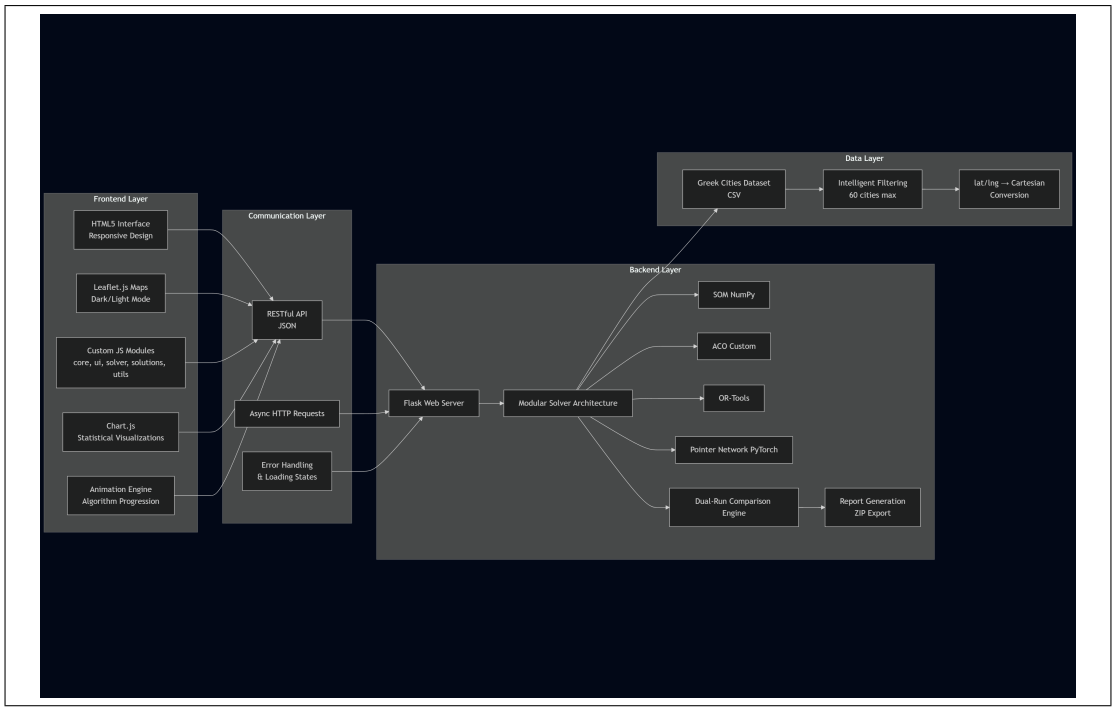
- **Educational Algorithms**: Extensive progression tracking (SOM, ACO, Pointer Network)
- **Professional Reference**: Minimal visualization overhead (OR-Tools)
- **Performance Comparison**: Clear distinction μεταξύ research και production approaches
- **Real-World Context**: Connection educational concepts με industry applications

4.10 Comprehensive System Architecture

4.11 System Capabilities και Technical Achievements

4.11.1 Algorithm Integration Excellence

Το σύστημα επιτυγχάνει seamless integration τεσσάρων διαφορετικών algorithmic approaches:



Σχήμα 4.2: Ολοκληρωμένη Αρχιτεκτονική της TSP Εφαρμογής [54]

- **Neural Network Approach**: SOM με PyTorch και GPU acceleration
- **Metaheuristic Optimization**: ACO με sophisticated pheromone management
- **Construction Heuristics**: Multi-algorithm pointer network approach
- **Professional Optimization**: OR-Tools integration ως benchmark reference

4.11.2 Visualization Innovation

Advanced visualization system που παρέχει:

- **Real-Time Algorithm Progression**: Step-by-step animation για educational purposes
- **Interactive Geographical Mapping**: Leaflet.js integration με Greek geographical data
- **Comparative Analysis Tools**: Dual-run methodology για objective algorithm comparison
- **Professional Presentation**: Production-grade output formatting

4.12 Research Contributions και Educational Value

4.12.1 Methodological Innovations

Dual-Run Comparison Framework: Η υλοποίηση dual-run methodology αποτελεί σημαντική contribution για fair algorithm comparison, αποκαλύπτοντας την επίδραση της αρχικής πόλης στην performance.

Geographic Realism: Η χρήση πραγματικών ελληνικών πόλεων παρέχει realistic context που συχνά απουσιάζει από academic implementations που χρησιμοποιούν random coordinates.

4.12.2 Educational Impact

Progressive Learning Approach:

- Step-by-step visualization για κατανόηση algorithm internals
- Comparative analysis που επιδεικνύει trade-offs μεταξύ approaches
- Professional vs. educational algorithm distinction
- Real-world application context με geographical data

4.13 Συμπεράσματα και Τεχνική Αξιολόγηση

4.13.1 Κύρια Τεχνικά Επιτεύγματα

Η ανεπτυγμένη εφαρμογή TSP αποτελεί comprehensive system που συνδυάζει:

- **Robust Backend Architecture**: Flask server με modular solver design και standardized API
- **Advanced Frontend Interface**: Modern JavaScript με responsive design και real-time visualization
- **Innovative Visualization Engine**: Algorithm progression animation με educational value
- **Comprehensive Analysis Framework**: Dual-run comparison methodology για objective evaluation
- **Professional Integration**: OR-Tools benchmark για industry-standard reference
- **Geographic Intelligence**: Real-world Greek cities με smart distribution algorithms

4.13.2 Educational και Research Impact

Educational Contributions:

- Interactive learning platform για TSP algorithm understanding
- Visual demonstration των trade-offs μεταξύ different approaches
- Real-world context που connects theory με practical applications
- Professional vs. educational algorithm perspective

Research Methodology:

- Fair comparison framework που εξαλείφει bias από starting city selection
- Comprehensive performance metrics για objective algorithm evaluation
- Scalable testing framework για multiple problem sizes
- Integration educational και professional optimization approaches

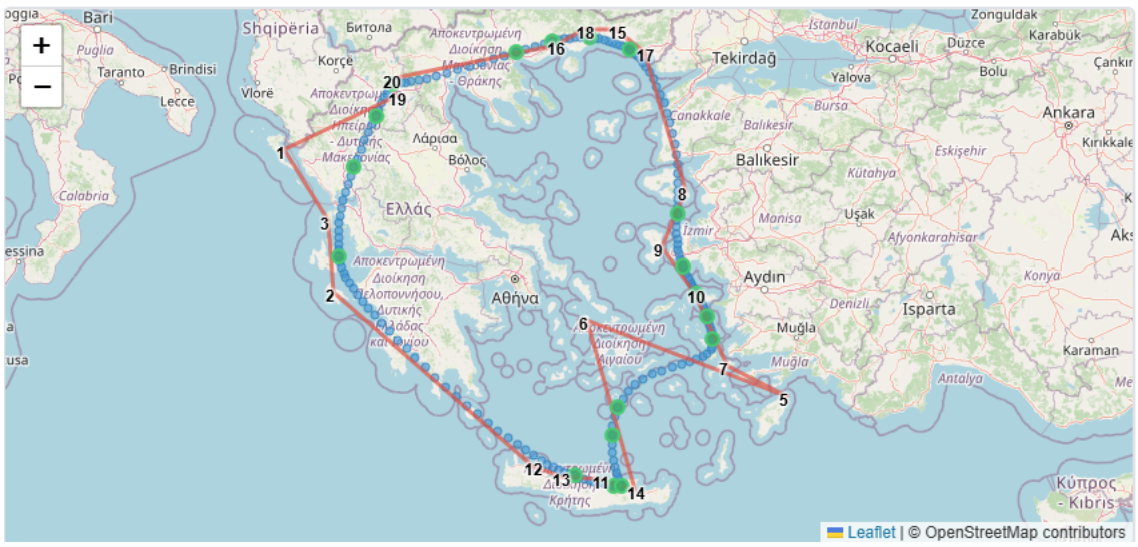
4.13.3 Technical Excellence

Το σύστημα επιδεικνύει high-quality software engineering practices:

- **Modular Design**: Clean separation of concerns με reusable components
- **Performance Optimization**: Efficient algorithms και memory management
- **User Experience**: Intuitive interface με comprehensive functionality
- **Cross-Platform Support**: Consistent performance across platforms και devices
- **Export Capabilities**: Multiple output formats για research και presentation use

Η εφαρμογή αποτελεί πλήρη σύστημα που υποστηρίζει τόσο την ερευνητική εργασία όσο και την εκπαιδευτική χρήση, παρέχοντας λεπτομερή ανάλυση των αλγορίθμων TSP σε realistic geographical context της Ελλάδας.

4.14 Τεχνική Ανάλυση – Self Organizing Map (SOM)



Σχήμα 4.3: Οπτικοποίηση του SOM αλγορίθμου κατά την εκπαίδευση. [53]

Ο Self-Organizing Map (SOM) αλγόριθμος αποτελεί μια προσέγγιση για την επίλυση του TSP που χρησιμοποιεί τεχνικές νευρωνικών δικτύων και μη επιβλεπόμενη μάθηση. Η υλοποίηση αξιοποιεί επιτάχυνση GPU και περιλαμβάνει προηγμένες δυνατότητες παρακολούθησης προόδου.

4.14.1 Μαθηματικό Υπόβαθρο και Θεωρητική Βάση

Ο SOM βασίζεται στην ιδέα της δημιουργίας ενός δικτύου νευρώνων που οργανώνεται τοπολογικά για να αντιπροσωπεύσει τις πόλεις του TSP. Η βασική αρχή είναι η δημιουργία ενός "ελαστικού δακτυλίου" που προσαρμόζεται στη γεωμετρία των πόλεων.

Κύρια μαθηματικά στοιχεία:

- **Δίκτυο Νευρώνων**: Κυκλική διάταξη N νευρώνων όπου $N = 10 \times \text{num\_cities}$
- **Γειτονιά Gauss**: $h_{ij} = \exp\left(-\frac{d_{ring}^2(i,j)}{2\sigma^2}\right)$
- **Κανόνας Μάθησης**: $\Delta w_j = \eta \cdot h_{ij} \cdot (x_i - w_j)$
- **Μετρική Απόστασης**: Ευκλείδεια απόσταση στο καρτεσιανό σύστημα

όπου $d_{ring}(i, j)$ είναι η κυκλική απόσταση μεταξύ νευρώνων, σ η ακτίνα επιρροής, η ο ρυθμός μάθησης, x_i η θέση της πόλης και w_j η θέση του νευρώνα.

4.14.2 Αλγοριθμική Προσέγγιση

Αρχικοποίηση Συστήματος

Η αρχικοποίηση του συστήματος ακολουθεί μια δομημένη προσέγγιση:

```
ΑΛΓΟΡΙΘΜΟΣ
 SOM_InitializationΕΙΣΟΔΟΣ
: cities[], parametersΕΞΟΔΟΣ
: neural_networkΑΡΧΗ


    // Μ
    coordinates ← convert_geographic_to_cartesian(cities)

    // Τ
    center ← calculate_centroid(coordinates)
    radius ← estimate_optimal_radius(coordinates)

    // Δ
    num_neurons ← 10 × length(cities)
    ΓΙΑ    neuron_index ΚΑΝΕ
        angle ← 2 × neuron_index / num_neurons
        neuron_position ← center + radius × [cos(angle), sin(angle)]

    ΕΠΙΣΤΡΕΦΕ neural_networkΤΕΛΟΣ
```

Listing 1: Αρχικοποίηση SOM

Εκπαιδευτική Διαδικασία

Η κεντρική εκπαιδευτική διαδικασία υλοποιεί προσαρμοστική μάθηση με φθίνουσες παραμέτρους:

```
ΑΛΓΟΡΙΘΜΟΣ
 SOM_TrainingΕΙΣΟΔΟΣ
: neural_network, cities[], max_epochsΕΞΟΔΟΣ
: trained_network, progression_dataΑΡΧΗ


    ΓΙΑ epoch = 1 ΕΩΣ max_epochs ΚΑΝΕ
        // Π
        learning_rate ← initial_rate × exp(-decay_factor × epoch)
        influence_radius ← initial_radius × exp(-decay_factor × epoch)

        // Ε
        ΓΙΑ    city ΚΑΝΕ
            // Ε             (Best Matching Unit)
            winner ← find_closest_neuron(city, neural_network)

            // Ε
            ΓΙΑ    neuron ΚΑΝΕ
                distance_to_winner ← calculate_ring_distance(neuron, winner)
                influence ← gaussian_function(distance_to_winner, influence_radius)
                position_update ← learning_rate × influence × (city - neuron)
                neuron.position ← neuron.position + position_update

        // Α
        current_solution ← extract_tour_from_neurons(neural_network)
        ΑΝ significant_improvement(current_solution) ΤΟΤΕ
            save_epoch_data(epoch, neural_network, current_solution)

    ΕΠΙΣΤΡΕΦΕ trained_network, progression_dataΤΕΛΟΣ
```

Listing 2: Εκπαίδευση SOM

4.14.3 Σύστημα Παρακολούθησης Προόδου

Το σύστημα περιλαμβάνει έξυπνη παρακολούθηση που αποθηκεύει μόνο σημαντικές αλλαγές:

Κριτήρια Αποθήκευσης: Αποθηκεύονται εποχές όταν υπάρχει σημαντική βελτίωση στην απόσταση, αλλαγή στη διαδρομή, ή αξιόλογη κίνηση των νευρώνων.

Φάσεις Εκπαίδευσης: Η εκπαίδευση διακρίνεται σε έξι φάσεις:

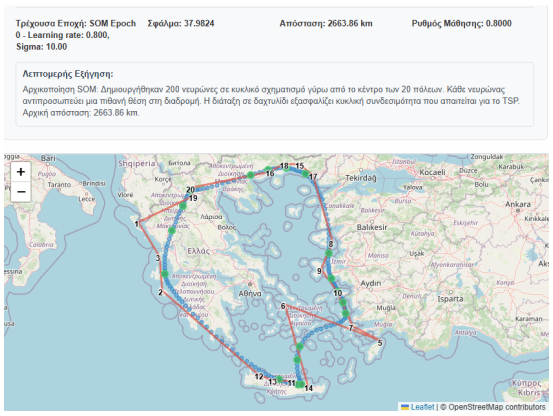
- **Αρχικοποίηση** (0%): Κυκλικός σχηματισμός νευρώνων
- **Εκμάθηση** (0-10%): Δραστικές προσαρμογές στις θέσεις πόλεων
- **Οργάνωση** (10-30%): Ανάπτυξη τοπολογικά σωστής διάταξης
- **Σύγκλιση** (30-70%): Εκλεπτύνσεις και βελτιστοποιήσεις
- **Τελειοποίηση** (70-90%): Λεπτές κινήσεις για φινάλε
- **Ολοκλήρωση** (90-100%): Σταθεροποίηση αποτελέσματος

4.14.4 Δομή Αποτελεσμάτων

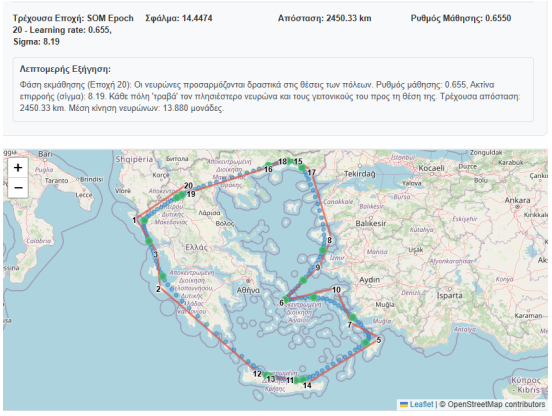
Το σύστημα επιστρέφει πλήρη δεδομένα για οπτικοποίηση:

Δεδομένα Εποχής: Κάθε αποθηκευμένη εποχή περιλαμβάνει θέσεις νευρώνων, τρέχουσα διαδρομή, συνολική απόσταση, μέσο σφάλμα, χρόνο εκτέλεσης και εξηγήσεις φάσης.

Τελικό Αποτέλεσμα: Βέλτιστη διαδρομή, συνολική απόσταση, χρόνος εκτέλεσης, πλήρης πρόοδος εκπαίδευσης και τελικές θέσεις νευρώνων.

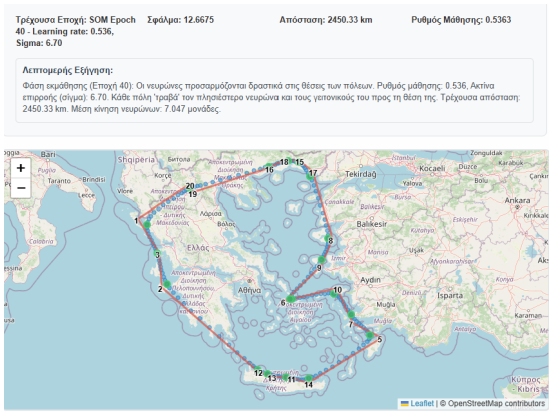


Epoch 0 - Αρχικοποίηση [53]

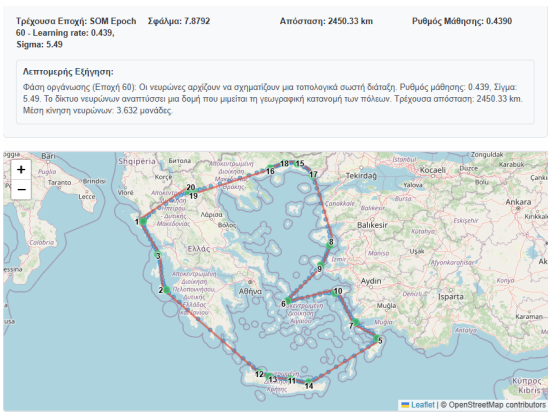


Epoch 20 - Πρώιμη εκμάθηση [53]

Σχήμα 4.4: Αρχικά στάδια εκπαίδευσης του SOM: Οι νευρώνες ξεκινούν σε κυκλικό σχηματισμό και αρχίζουν να προσαρμόζονται στις θέσεις των πόλεων.



Epoch 40 - Φάση οργάνωσης [53]



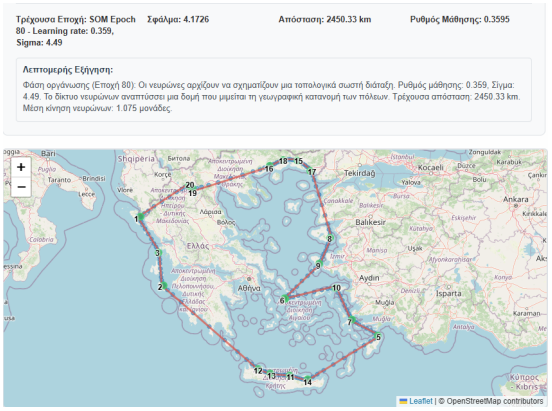
Epoch 60 - Τοπολογική διάταξη [53]

Σχήμα 4.5: Μέση πρόοδος εκπαίδευσης του SOM: Ανάπτυξη τοπολογικά σωστής διάταξης που μιμείται τη γεωγραφική κατανομή των πόλεων. [53]

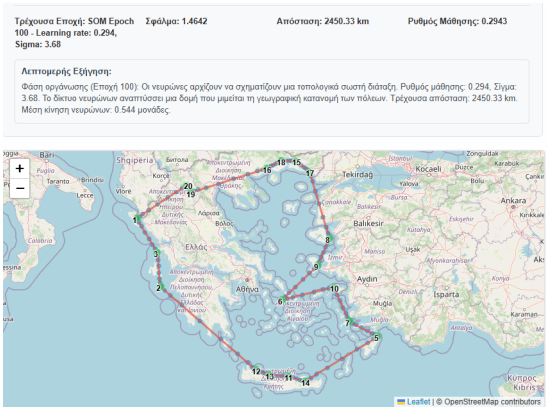
4.14.5 Οπτικοποίηση και Ενσωμάτωση

Το σύστημα οπτικοποίησης περιλαμβάνει:

- Εμφάνιση του δακτυλίου νευρώνων με ενημέρωση σε πραγματικό χρόνο
- Διακριτούς δείκτες για κάθε νευρώνα



Epoch 80 - Εκλεπτύνσεις [53]



Epoch 100 - Σύγκλιση [53]

Σχήμα 4.6: Τελικά στάδια εκπαίδευσης του SOM: Λεπτές προσαρμογές για βελτιστοποίηση και σταθεροποίηση της διαδρομής. [53]

- Πληροφορίες προόδου με ποσοστά και μετρήσεις
- Εξηγήσεις φάσης στα ελληνικά
- Διαδραστικά χειριστήρια για περιήγηση στην πρόοδο εκπαίδευσης

4.14.6 Βελτιστοποιήσεις Απόδοσης

Η υλοποίηση περιλαμβάνει:

- **Ομαδική Επεξεργασία**: Παράλληλος υπολογισμός αποστάσεων
- **Επιτάχυνση GPU**: Αυτόματη αξιοποίηση CUDA όταν είναι διαθέσιμο
- **Διαχείριση Μνήμης**: Αποδοτικές τανυστικές λειτουργίες
- **Έξυπνη Αποθήκευση**: Αποθήκευση μόνο σημαντικών epochs

4.14.7 Αναφορά Υλοποίησης και Πηγή

Η τεχνική υλοποίηση του SOM-TSP βασίστηκε σε υπάρχουσα open-source προσέγγιση:

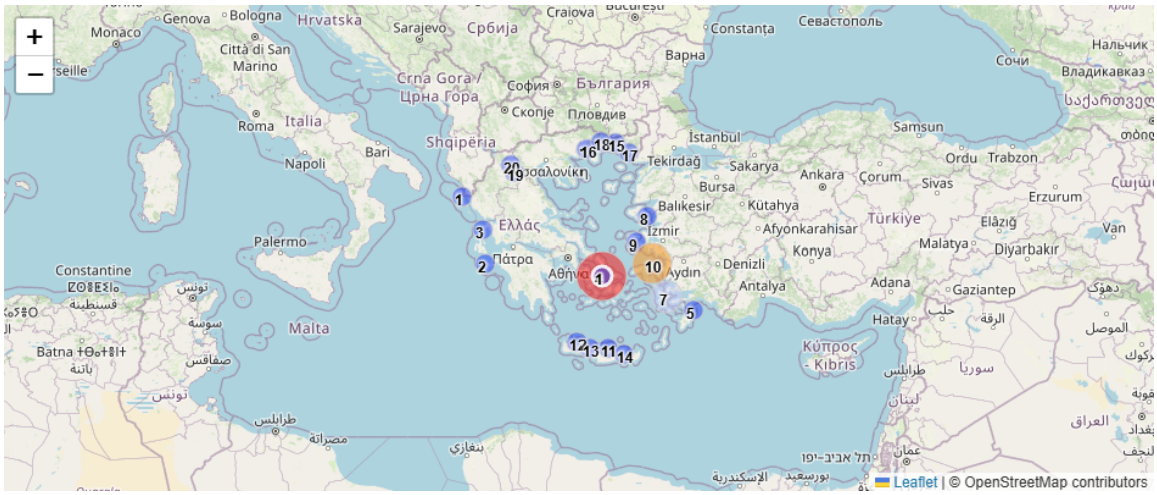
> Diego Vicente (2023). *Self-Organizing Map for the Traveling Salesman Problem* [GitHub Repository]. `https://github.com/diego-vicente/som-tsp`

Κύριες επεκτάσεις:

- Ενσωμάτωση GPU για παράλληλους υπολογισμούς
- Προηγμένο σύστημα παρακολούθησης με λεπτομερείς αλληλεπιδράσεις

- Δυναμικές εξηγήσεις στα ελληνικά για κάθε φάση εκπαίδευσης
- Έξυπνη επιλογή epochs για βελτιστοποίηση μνήμης
- Πλούσια μεταδεδομένα για frontend οπτικοποίηση
- Στρατηγικές βελτιστοποίησης απόδοσης και διαχείρισης μνήμης

4.15 Τεχνική Ανάλυση – Ant Colony Optimization (ACO)



Σχήμα 4.7: Οπτικοποίηση του ACO αλγορίθμου κατά την εξέλιξη των φερομορμονών. [53]

Ο Ant Colony Optimization (ACO) αλγόριθμος αποτελεί μια βιο-εμπνευσμένη μεταευρετική προσέγγιση που μιμείται τη συμπεριφορά αποικιών μυρμηγκιών στην αναζήτηση τροφής. Η υλοποίηση περιλαμβάνει προηγμένη διαχείριση φερομονών, λεπτομερή παρακολούθηση μυρμηγκιών και προοδευτική οπτικοποίηση για την κατανόηση της εσωτερικής λειτουργίας.

4.15.1 Μαθηματικό Υπόβαθρο και Θεωρητική Βάση

Ο ACO βασίζεται στη δυναμική αλληλεπίδραση μεταξύ τεχνητών μυρμηγκιών που κατασκευάζουν λύσεις και ενός συστήματος φερομονών που διατηρεί τη μνήμη των καλών λύσεων.

Κύρια μαθηματικά στοιχεία:

- **Πίνακας Φερομονών**: $\tau_{ij}(t)$ - ποσότητα φερομόνης στην ακμή (i, j) τη χρονική στιγμή t
- **Πίνακας Ορατότητας**: $\eta_{ij} = \frac{1}{d_{ij}}$ - ορατότητα (αντίστροφη της απόστασης)
- **Κανόνας Πιθανότητας**: $p_{ij}^k = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta}$
- **Ενημέρωση Φερομονών**: $\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k$

όπου α η σημασία των φερομονών, β η σημασία της απόστασης, ρ ο ρυθμός εξάτμισης και $\Delta\tau_{ij}^k = \frac{Q}{L^k}$ η φερομόνη που καταθέτει το μυρμήγκι k με διαδρομή μήκους L^k.

4.15.2 Αλγοριθμική Προσέγγιση

Αρχικοποίηση Συστήματος

Η αρχικοποίηση του ACO ακολουθεί μια δομημένη προσέγγιση:

```
ΑΛΓΟΡΙΘΜΟΣ
 ACO_InitializationΕΙΣΟΔΟΣ
: cities[], algorithm_parametersΕΞΟΔΟΣ
: aco_systemΑΡΧΗ


    // Μ
    coordinates ← convert_geographic_to_cartesian(cities)

    // Δ
    distance_matrix ← calculate_distance_matrix(coordinates)

    // Α
    ΓΙΑ    edge(i,j) ΚΑΝΕ
        pheromone_matrix[i][j] ← small_random_value + base_pheromone

    // Ε                          TSP
    ensure_matrix_symmetry(pheromone_matrix)

    // Τ
    ΓΙΑ    edge(i,j) ΚΑΝΕ
        visibility_matrix[i][j] ← 1.0 / distance_matrix[i][j]

    ΕΠΙΣΤΡΕΦΕ aco_systemΤΕΛΟΣ
```

Listing 3: Αρχικοποίηση ACO

Κατασκευή Λύσεων από Μυρμήγκια

Κάθε μυρμήγκι κατασκευάζει μια λύση χρησιμοποιώντας πιθανοτική επιλογή:

```
ΑΛΓΟΡΙΘΜΟΣ
 Ant_Solution_ConstructionΕΙΣΟΔΟΣ
: ant_id, pheromone_matrix, visibility_matrixΕΞΟΔΟΣ
: ant_path, journey_detailsΑΡΧΗ


    // Ε
    current_city ← random_start_city()
    ant_path ← [current_city]
    unvisited_cities ← all_cities EXCEPT current_city

    // Κ
    ΟΣΟΔΗΠΟΤΕ unvisited_cities  empty ΚΑΝΕ
        // Τ
        probabilities ← calculate_selection_probabilities(
            current_city, unvisited_cities, pheromone_matrix, visibility_matrix)

```

```
        // E               roulette wheel
        next_city ← roulette_wheel_selection(probabilities)

        // K
        record_movement(current_city, next_city, journey_details)
        ADD next_city TO ant_path
        REMOVE next_city FROM unvisited_cities
        current_city ← next_city

    ΕΠΙΣΤΡΕΦΕ ant_path, journey_detailsΤΕΛΟΣ
```

Listing 4: Κατασκευή Λύσης ACO

Διαχείριση Φερομονών

Το σύστημα φερομονών ενημερώνεται συνδυάζοντας εξάτμιση και κατάθεση:

```
ΑΛΓΟΡΙΘΜΟΣ
 Pheromone_UpdateΕΙΣΟΔΟΣ
: ant_solutions[], evaporation_rate, pheromone_constantΕΞΟΔΟΣ
: updated_pheromone_matrixΑΡΧΗ


    // Φ         -
    ΓΙΑ    edge(i,j) ΚΑΝΕ
        pheromone_matrix[i][j] ← pheromone_matrix[i][j] × (1 - evaporation_rate)

    // Φ         -
    ΓΙΑ    ant_solution ΚΑΝΕ
        ΑΝ valid_complete_tour(ant_solution) ΤΟΤΕ
            deposit_amount ← pheromone_constant / solution_distance

            ΓΙΑ    edge    ant_solution ΚΑΝΕ
                // K
                pheromone_matrix[edge.from][edge.to] += deposit_amount
                pheromone_matrix[edge.to][edge.from] += deposit_amount

    ΕΠΙΣΤΡΕΦΕ updated_pheromone_matrixΤΕΛΟΣ
```

Listing 5: Ενημέρωση Φερομονών

4.15.3 Σύστημα Παρακολούθησης Προόδου

Ο αλγόριθμος περιλαμβάνει προηγμένη παρακολούθηση που καταγράφει την εξέλιξη:

Φάσεις Εξέλιξης: Η εκτέλεση διακρίνεται σε τέσσερις φάσεις:

- **Εξερεύνηση** (0-20%): Τυχαία διαδρομές για δημιουργία αρχικής φερομόνης
- **Ενίσχυση** (20-60%): Φερομόνες κατευθύνουν προς καλύτερες διαδρομές
- **Βελτιστοποίηση** (60-90%): Σύγκλιση σε υψηλής ποιότητας λύσεις

- **Σύγκλιση** (90-100%): Σταθεροποίηση τελικού αποτελέσματος

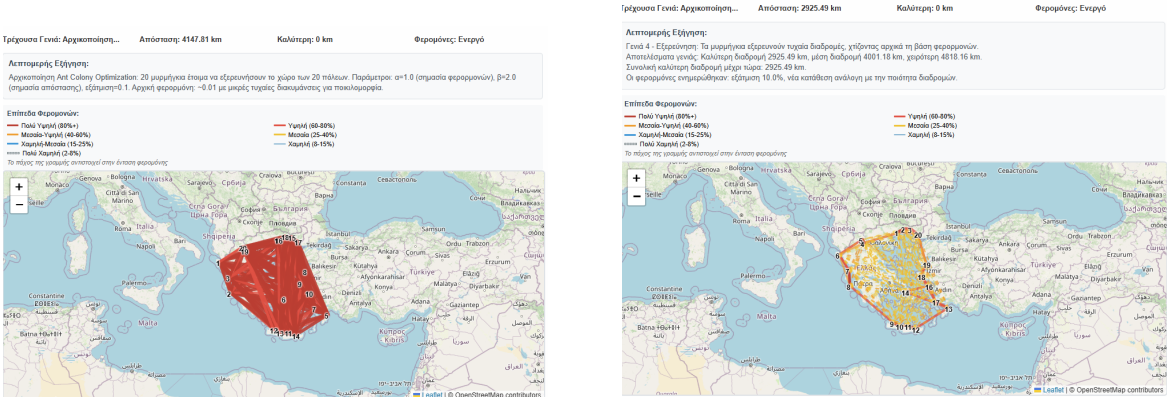
Προσαρμοστικές Παράμετροι: Το σύστημα προσαρμόζει αυτόματα τον αριθμό μυρμηγκιών και iterations βάσει του μεγέθους του προβλήματος.

4.15.4 Δομή Αποτελεσμάτων

Το σύστημα επιστρέφει πλήρη δεδομένα για οπτικοποίηση:

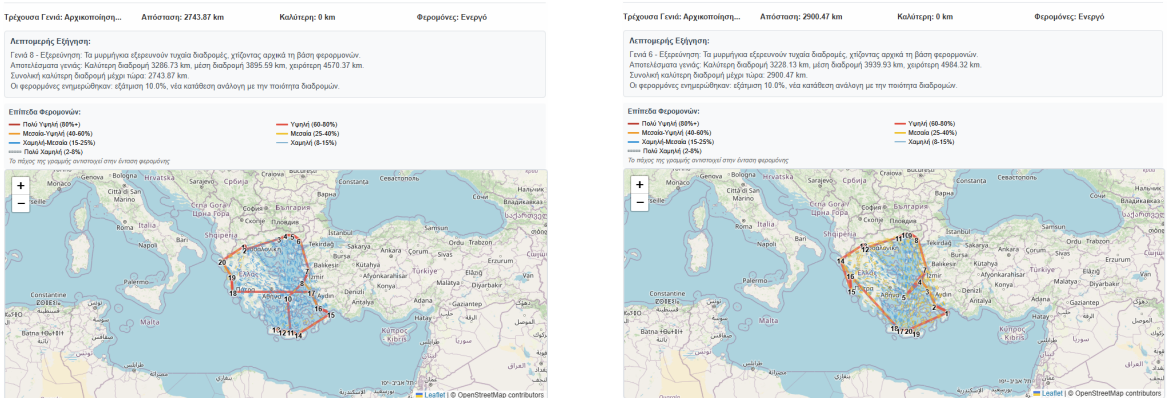
Δεδομένα Γενιάς: Κάθε γενιά περιλαμβάνει καλύτερη διαδρομή, όλες τις διαδρομές μυρμηγκιών, λεπτομερή ταξίδια, τρέχοντα πίνακα φερομονών και εξηγήσεις φάσης.

Τελικό Αποτέλεσμα: Βέλτιστη διαδρομή, συνολική απόσταση, χρόνος εκτέλεσης, πλήρης εξέλιξη αλγορίθμου και τελική κατάσταση φερομονών.



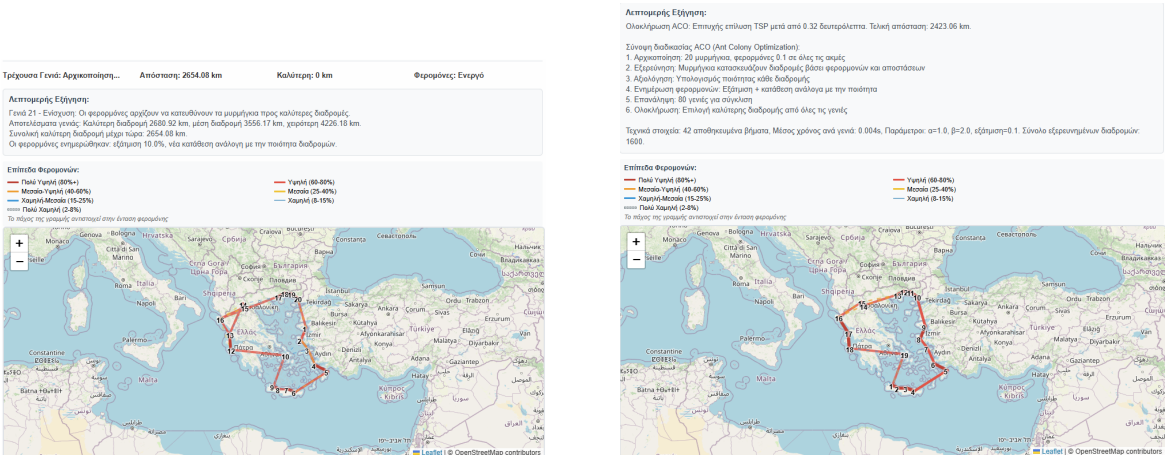
Iteration 0 - Αρχικοποίηση [53] Iteration 5 - Πρώιμη εξερεύνηση [53]

Σχήμα 4.8: Αρχικά στάδια ACO: Μυρμήγκια εξερευνούν τυχαία και αρχίζουν να κατασκευάζουν τη βάση φερομόνων. [53]



Iteration 20 - Φάση ενίσχυσης [53] Iteration 40 - Αυτοοργάνωση [53]

Σχήμα 4.9: Μέση πρόοδος ACO: Φερομόνες κατευθύνουν τα μυρμήγκια προς καλύτερες διαδρομές με θετική ανάδραση. [53]



Iteration 60 - Βελτιστοποίηση [53] Iteration 80 - Σύγκλιση [53]

Σχήμα 4.10: Τελικά στάδια ACO: Σύγκλιση προς βέλτιστες διαδρομές με σταθεροποίηση φερομόνων. [53]

4.15.5 Οπτικοποίηση και Ενσωμάτωση

Το σύστημα οπτικοποίησης περιλαμβάνει:

- Εμφάνιση φερομόνων με διαφάνεια ανάλογη της ισχύος
- Οπτικοποίηση διαδρομών μυρμηγκιών με διαφορετικά χρώματα
- Πληροφορίες προόδου με ποσοστά και στατιστικά
- Εξηγήσεις φάσης για κάθε στάδιο εξέλιξης
- Διαδραστικά χειριστήρια για περιήγηση στην εξέλιξη αλγορίθμου

Οπτικοποίηση Φερομόνων: Οι φερομόνες εμφανίζονται ως γραμμές με διαφάνεια και πάχος ανάλογα της ισχύος, με χρωματική κωδικοποίηση από πράσινο (ασθενή) σε κόκκινο (ισχυρή).

4.15.6 Βελτιστοποιήσεις Απόδοσης

Η υλοποίηση περιλαμβάνει:

- **Προσαρμοστικές Παράμετροι**: Αυτόματη προσαρμογή μυρμηγκιών και iterations
- **Αποδοτικές Πράξεις Πίνακα**: Vectorized υπολογισμοί για ενημερώσεις φερομονών
- **Έξυπνη Αποθήκευση**: Καταγραφή προόδου κάθε 3 iterations
- **Πιθανοτική Επιλογή**: Αποδοτική roulette wheel selection

4.15.7 Θεωρητικό Υπόβαθρο

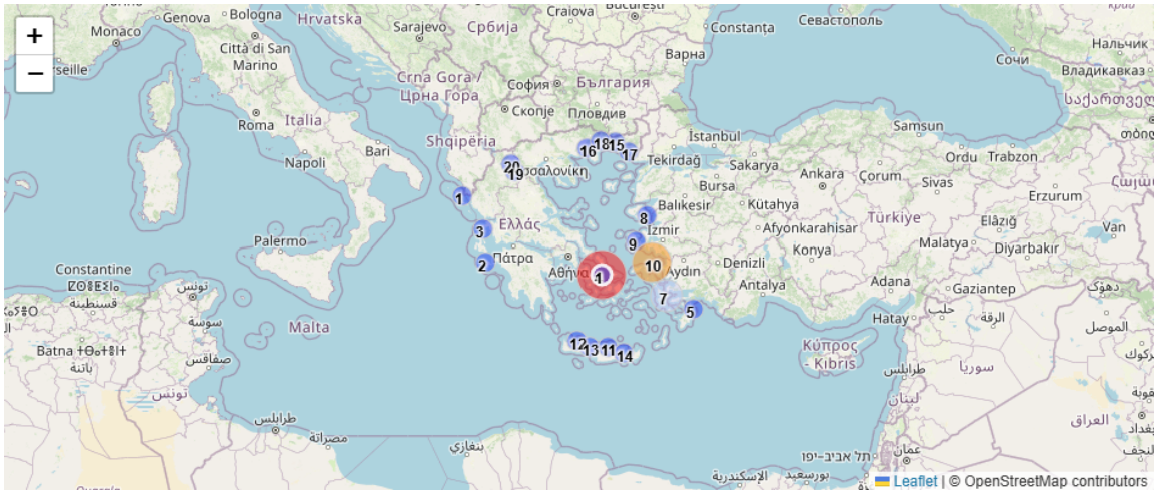
Ο ACO αναπτύχθηκε από τον Marco Dorigo και βασίζεται στη μελέτη πραγματικών αποικιών μυρμηγκιών. Οι βασικές αρχές περιλαμβάνουν:

- **Stigmergy**: Έμμεση επικοινωνία μέσω περιβαλλοντικών τροποποιήσεων
- **Θετική Ανάδραση**: Καλές διαδρομές ενισχύονται με περισσότερες φερομόνες
- **Κατανεμημένος Υπολογισμός**: Τοπικές αποφάσεις από κάθε μυρμήγκι
- **Κατασκευαστική Μεταευρετική**: Προοδευτική κατασκευή πλήρων λύσεων

Κύριες επεκτάσεις:

- Προηγμένη οπτικοποίηση με λεπτομερή καταγραφή ταξιδιών
- Προσαρμοστικές εξηγήσεις για κάθε φάση εξέλιξης
- Έξυπνη προσαρμογή παραμέτρων βάσει μεγέθους προβλήματος
- Πλούσια μεταδεδομένα για οπτικοποίηση πραγματικού χρόνου
- Παρακολούθηση απόδοσης και στατιστικά για κάθε γενιά

4.16 Τεχνική Ανάλυση – High-Quality Construction Algorithms (Pointer Network)



Σχήμα 4.11: Οπτικοποίηση των High-Quality Construction Algorithms με Progressive Optimization. [53]

Η υλοποίηση που παρουσιάζεται ως ”Pointer Network” αποτελεί στην πραγματικότητα μια συλλογή προηγμένων construction algorithms που συνδυάζουν πολλαπλές ευρετικές προσεγγίσεις με στόχο την παραγωγή υψηλής ποιότητας λύσεων. Αντί για deep learning προσέγγιση, η υλοποίηση εστιάζει σε deterministic construction heuristics με έμφαση στην ποιότητα των αρχικών λύσεων.

4.16.1 Αρχιτεκτονική και Φιλοσοφία Σχεδιασμού

Η προσέγγιση βασίζεται στη συνδυαστική χρήση τριών κύριων construction algorithms, καθένας με διαφορετικές δυνάμεις και στρατηγικές:

Κύρια στοιχεία αρχιτεκτονικής:

- **Πολλαπλές Στρατηγικές Κατασκευής**: Cheapest Insertion, Farthest Insertion, Enhanced Nearest Neighbor
- **Προοδευτική Εκτέλεση**: Πολλαπλές εποχές με διαφορετικές τεχνικές και εκκινήσεις
- **Προσαρμοστική Επιλογή**: Αυτόματη προσαρμογή παραμέτρων βάσει μεγέθους προβλήματος
- **Περιορισμένη 2-Opt**: Έμφαση σε υψηλής ποιότητας construction vs. εκτενή post-processing
- **Στρατηγικά Σημεία Εκκίνησης**: Έξυπνη επιλογή για κάθε construction strategy

4.16.2 Αλγοριθμικές Στρατηγικές

1. Cheapest Insertion Algorithm

Ο Cheapest Insertion αποτελεί έναν sophisticated greedy algorithm που κατασκευάζει tours με ελαχιστοποίηση της αύξησης κόστους:

Κύρια χαρακτηριστικά:

- **Αρχικοποίηση**: Δημιουργία αρχικού τριγώνου με ελάχιστο περίμετρο
- **Στρατηγική Εισαγωγής**: Εύρεση θέσης με ελάχιστη αύξηση κόστους για κάθε νέα πόλη
- **Διαδικασία**: Προοδευτική εισαγωγή όλων των πόλεων στις βέλτιστες θέσεις
- **Αποτελεσματικότητα**: Εξαιρετική ισορροπία μεταξύ ποιότητας και ταχύτητας

2. Farthest Insertion Algorithm

Η Farthest Insertion στρατηγική προτεραιοποιεί την εισαγωγή πόλεων που βρίσκονται μακριά από το τρέχον tour:

Κύρια χαρακτηριστικά:

- **Boundary-Focused**: Προτεραιότητα σε πόλεις που βρίσκονται στα όρια της γεωγραφικής περιοχής
- **Στρατηγική Επιλογής**: Εύρεση πόλης με μέγιστη απόσταση από τρέχον tour
- **Τοποθέτηση**: Εισαγωγή στη θέση με ελάχιστη αύξηση κόστους

- **Πλεονεκτήματα**: Αποτελεσματική για γεωγραφικά προβλήματα με outliers

3. Enhanced Nearest Neighbor

Βελτιωμένη εκδοχή του κλασικού Nearest Neighbor με sophisticated selection strategies:

Κύρια χαρακτηριστικά:

- **Πολλαπλά Variants**: Standard, probabilistic και weighted selection
- **Ποικιλομορφία**: Περιστασιακή επιλογή δεύτερου εγγύτερου για αποφυγή τοπικών optimum
- **Weighted Selection**: Προσαρμοσμένη επιλογή από top candidates
- **Anti-Crossing Logic**: Αποφυγή δημιουργίας διασταυρώσεων όπου είναι δυνατό

4.16.3 Multi-Algorithm Execution Strategy

Το σύστημα εκτελεί στρατηγική επιλογή αλγορίθμων σε πολλαπλές iterations:

Στρατηγική Εκτέλεσης:

- **Iteration 0-1**: Cheapest Insertion με διαφορετικά σημεία εκκίνησης (κέντρο, βορειότερη)
- **Iteration 2**: Farthest Insertion από outlier για boundary emphasis
- **Iteration 3**: Enhanced Nearest Neighbor με progressive 2-opt
- **Iterations 4+**: Hybrid strategies με rotating starting points

Προσαρμοστικές Παράμετροι:

- Αυτόματη προσαρμογή iterations βάσει μεγέθους προβλήματος
- Έξυπνη επιλογή εποχών ανά iteration
- Προσαρμοστική εφαρμογή 2-opt optimization
- Στρατηγική επιλογή starting points για comprehensive coverage

4.16.4 Progressive 2-Opt Integration

Η υλοποίηση περιλαμβάνει intelligent 2-opt optimization με περιορισμένη εφαρμογή:

Φιλοσοφία: Αντί για εκτενή post-processing, η έμφαση δίνεται στην παραγωγή υψηλής ποιότητας αρχικών λύσεων που χρειάζονται ελάχιστη βελτίωση.

Στρατηγική Εφαρμογή:

- Μέγιστο 5 improvements ανά εφαρμογή για αποφυγή over-optimization
- Εφαρμογή σε επιλεγμένες iterations για ισορροπία
- Έξυπνη ανίχνευση βελτιώσεων με early termination
- Αποφυγή διαδοχικών ακμών για αποδοτικότητα

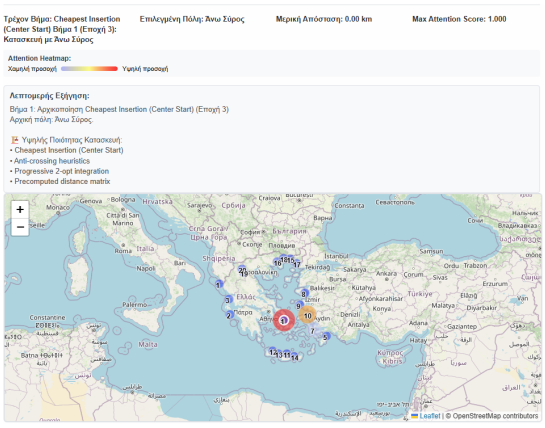
4.16.5 Οπτικοποίηση Αλγοριθμικής Προόδου

Το σύστημα visualization περιλαμβάνει προηγμένη παρακολούθηση:

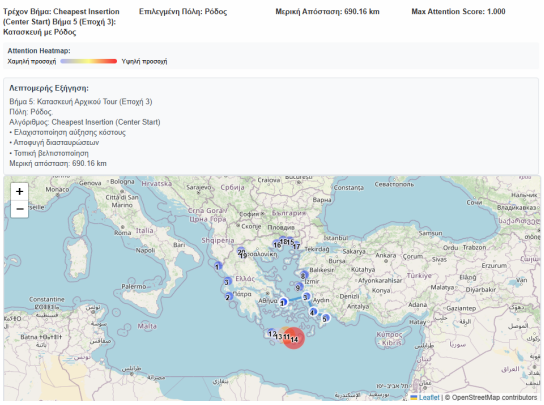
Step-by-Step Construction:

- **Προοδευτική Κατασκευή**: Οπτικοποίηση της προσθήκης κάθε πόλης στο tour
- **Algorithm-Specific Highlighting**: Διαφορετική απεικόνιση για κάθε construction strategy
- **Attention Scores**: Εμφάνιση candidate πόλεων που εξετάζονται για εισαγωγή
- **Interactive Timeline**: Δυνατότητα περιήγησης στην εξέλιξη construction process
- **Multi-Algorithm View**: Σύγκριση διαφορετικών approaches στην ίδια οθόνη

Ιδιαιτερότητες Visualization: Σε αντίθεση με τους metaheuristic αλγορίθμους (SOM, ACO), οι construction algorithms είναι deterministic, οπότε η οπτικοποίηση εστιάζει στη λογική κατασκευής αντί για στοχαστική εξέλιξη.



Cheapest Insertion (Center Start) [53]



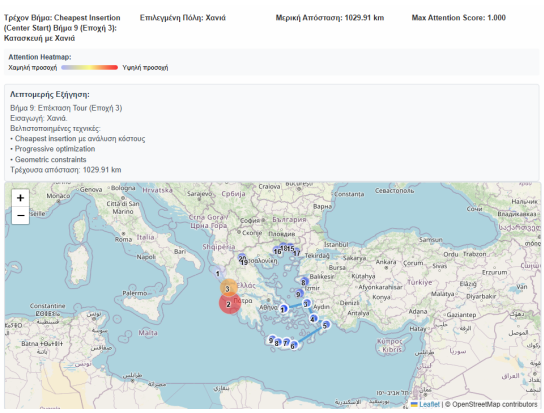
Cheapest Insertion (North Start) [53]

Σχήμα 4.12: Αρχικές στρατηγικές: Διαφορετικές εκκινήσεις με Cheapest Insertion algorithm για comprehensive coverage. [53]

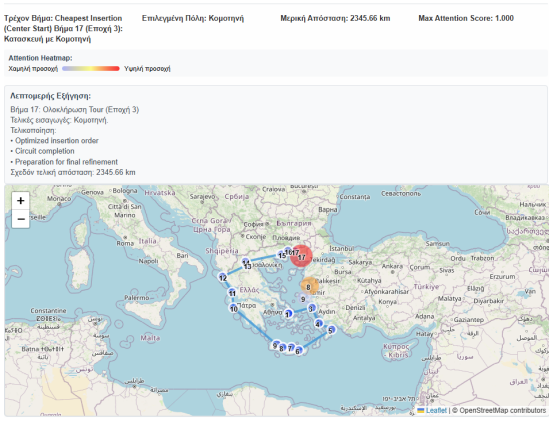
4.16.6 Δομή Αποτελεσμάτων και Μεταδεδομένα

Το σύστημα παρέχει πλούσια δεδομένα για ανάλυση:

Comprehensive Algorithm Information:

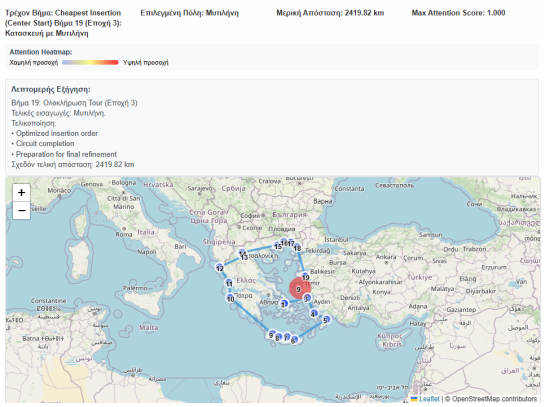


Farthest Insertion Strategy [53]

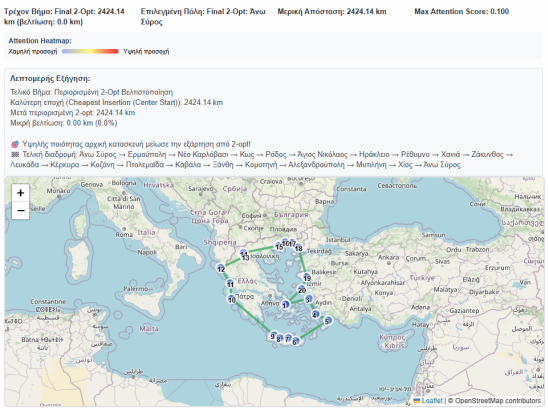


Enhanced Nearest Neighbor [53]

Σχήμα 4.13: Εναλλακτικές προσεγγίσεις: Farthest Insertion για boundary emphasis και Enhanced Nearest Neighbor με ποικιλομορφία. [53]



Hybrid Strategy [53]



Best Solution Selected [53]

Σχήμα 4.14: Τελικές φάσεις: Hybrid strategies και επιλογή της βέλτιστης λύσης από όλες τις προσεγγίσεις. [53]

- Λεπτομερείς πληροφορίες για κάθε iteration και epoch
- Step-by-step progression με attention scores
- Algorithm-specific metadata και εξηγήσεις
- Comparison data μεταξύ διαφορετικών strategies
- Performance metrics και computation times

Educational Value:

- Σαφή κατανόηση των construction processes
- Διαφάνεια στη λογική επιλογής κάθε αλγορίθμου
- Visualization της επίδρασης διαφορετικών starting points
- Άμεση σύγκριση αποτελεσμάτων multiple approaches

4.16.7 Βελτιστοποιήσεις Απόδοσης

Η υλοποίηση περιλαμβάνει αρκετές βελτιστοποιήσεις:

Αποδοτικότητα:

- **Precomputed Distance Matrix**: Αποφυγή επαναλαμβανόμενων υπολογισμών
- **Efficient Data Structures**: Χρήση sets για unvisited tracking
- **Smart Epoch Selection**: Αποθήκευση μόνο σημαντικών βημάτων
- **Limited 2-Opt**: Μέγιστο improvements για αποφυγή over-optimization
- **Haversine Distance**: Ακριβή γεωγραφικά distances

Scalability: Excellent performance σε μεταβλητά problem sizes με προσαρμοστικές παραμέτρους που εξασφαλίζουν αποδοτικότητα ανεξάρτητα από το μέγεθος του προβλήματος.

4.16.8 Θεωρητικό Υπόβαθρο και Αναφορές

Παρόλο που η αρχική ιδέα αναφέρθηκε στο Pointer Network repository του Rintaro Yamazaki, η πραγματική υλοποίηση εστιάζει σε classical optimization algorithms που αποδείχθηκαν εξαιρετικά αποτελεσματικοί.

Κύριες τεχνικές που ενσωματώθηκαν:

- **Cheapest Insertion Heuristic**: Κλασικός construction algorithm με αποδεδειγμένη αποτελεσματικότητα

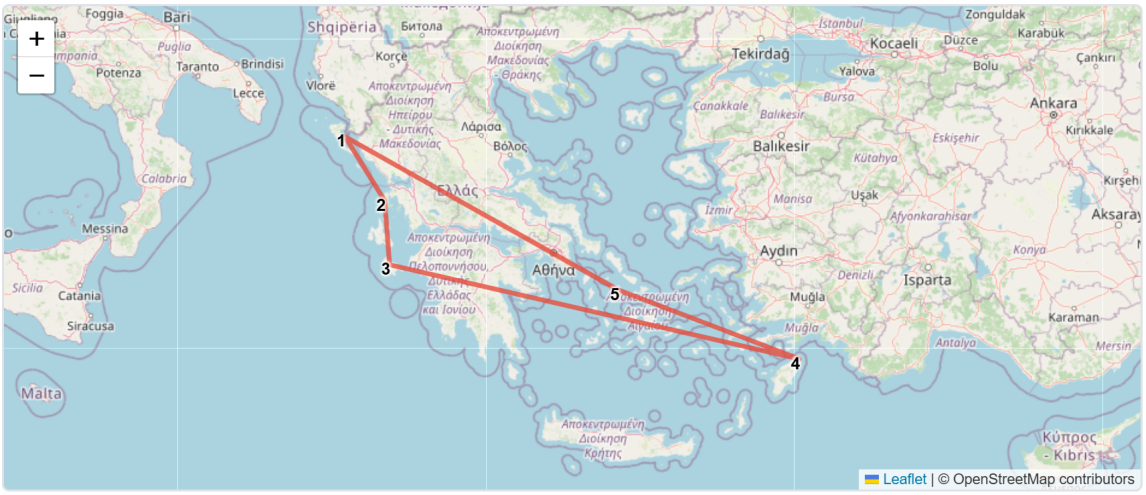
- **Farthest Insertion Strategy**: Boundary-focused approach για γεωγραφικά προβλήματα
- **Enhanced Nearest Neighbor**: Βελτιωμένη εκδοχή με probabilistic selection
- **Progressive 2-Opt**: Controlled local search integration
- **Multi-Start Strategy**: Comprehensive search μέσω πολλαπλών εκκινήσεων
- **Adaptive Parameter Selection**: Smart configuration για διαφορετικά problem sizes

Πλεονεκτήματα της προσέγγισης:

- **Deterministic Results**: Αναπαραγωγιμότητα με fixed seed
- **High Initial Quality**: Minimal εξάρτηση από post-processing
- **Fast Execution**: Efficient algorithms χωρίς neural network overhead
- **Comprehensive Coverage**: Πολλαπλές στρατηγικές για διαφορετικά problem characteristics
- **Educational Value**: Σαφή κατανόηση των construction processes
- **Scalability**: Excellent performance σε μεταβλητά problem sizes

Η υλοποίηση αποτελεί μια comprehensive συλλογή των καλύτερων classical construction techniques, εμπλουτισμένη με σύγχρονα visualization tools και progressive optimization strategies.

4.17 Τεχνική Ανάλυση – OR-Tools Professional Optimization (Benchmark Reference)



Σχήμα 4.15: OR-Tools: Professional Grade Optimization Engine από την Google. [53]

Ο OR-Tools αποτελεί την επίσημη βιβλιοθήκη της Google για βελτιστοποίηση και έχει ενσωματωθεί στο σύστημα ως **benchmark reference** και **professional baseline** για την αξιολόγηση των άλλων αλγορίθμων. Σε αντίθεση με τους experimental και educational αλγορίθμους που παρέχουν detailed progression tracking, ο OR-Tools εστιάζει αποκλειστικά στην παραγωγή optimal solutions με production-grade performance.

4.17.1 Αρχιτεκτονική και Φιλοσοφία Professional Solver

Ο OR-Tools αντιπροσωπεύει μια διαφορετική φιλοσοφία σχεδιασμού από τους υπόλοιπους αλγορίθμους του συστήματος. Ενώ οι άλλοι αλγόριθμοι έχουν σχεδιαστεί για εκπαιδευτικούς σκοπούς με extensive visualization, ο OR-Tools παρέχει:

Κύρια χαρακτηριστικά επαγγελματικού solver:

- **Production-Grade Performance**: Βελτιστοποιημένος για μέγιστη απόδοση
- **Industrial Strength**: Χρησιμοποιείται σε πραγματικές εφαρμογές της Google
- **Comprehensive Algorithm Suite**: Συνδυασμός πολλαπλών optimization techniques
- **Minimal Overhead**: Χωρίς visualization overhead για μέγιστη ταχύτητα
- **Benchmark Reference**: Αποτελεσματικό baseline για algorithm comparison

4.17.2 Vehicle Routing Problem Framework Integration

Η υλοποίηση του TSP γίνεται μέσω του Vehicle Routing Problem (VRP) framework του OR-Tools:

Κύρια στοιχεία framework:

- **RoutingIndexManager**: Διαχείριση indices για VRP μοντέλο

- **RoutingModel**: Κεντρικό μοντέλο για optimization
- **Distance Matrix Optimization**: Precomputed Euclidean distances
- **Coordinate Transformation**: Lat/lng σε XY για ακριβή υπολογισμούς

- **Integer Scaling**: SCALE$_FACTOR = 100,000$

Distance Callback System: Το σύστημα χρησιμοποιεί sophisticated distance management με callback functions που επιτρέπουν στον OR-Tools να έχει άμεση πρόσβαση σε precomputed distances. Η κλιμάκωση σε integers είναι απαραίτητη για την optimal απόδοση του constraint solver.

4.17.3 Multi-Strategy Search Configuration

Ο OR-Tools εκτελεί πολλαπλές optimization strategies για comprehensive problem solving:

First Solution Strategies:

- **PATH**$_CHEAPEST_ARC$: $Greedyconstructionminimumcostedges$**SAVINGS** : $Clarke-WrightSavingsalgorithmVRP$
- **CHRISTOFIDES**: Advanced graph-based TSP construction

Local Search Metaheuristics:

- **GUIDED**$_LOCAL_SEARCH$: $Sophisticatedpenalty-basedescapemechanism$**Large Neighborhood Search** $ImplicitOR-Toolsframework$
- **Simulated Annealing**: Available για diverse optimization landscapes

Production-Grade Final Optimization: Η τελική optimization φάση χρησιμοποιεί aggressive settings με 60 δευτερόλεπτα time limit, 1000 solution limit και `GUIDED_LOCAL_SEARCH` metaheuristic για optimal results.

4.17.4 Benchmark Role και Algorithm Comparison Framework

Ο OR-Tools λειτουργεί ως **gold standard benchmark** για την αξιολόγηση των άλλων αλγορίθμων:

Benchmarking Criteria:

- **Solution Quality**: Συγκριτική αξιολόγηση distance optimality
- **Computational Efficiency**: Reference για execution time comparison
- **Consistency**: Αναπαραγώγιμα αποτελέσματα για fair comparison
- **Scalability Assessment**: Performance σε διαφορετικά problem sizes

Πίνακας 4.1: OR-Tools vs. Experimental Algorithms Comparison Framework

Aspect	**OR-Tools**	**Experimental Algorithms**	**Purpose**
Progression	Minimal	Detailed step-by-step	Education vs. Production
Optimization	Production-grade	Educational	Quality vs. Understanding
Performance	Maximum speed	Visualization overhead	Efficiency vs. Learning
Complexity	Industrial	Accessible	Professional vs. Academic
Output	Final result	Rich metadata	Results vs. Process

4.17.5 Professional Integration και Technical Architecture

Η υλοποίηση περιλαμβάνει robust error handling και production-grade features:

Production Features:

- **Reproducible Results**: Consistent seeding για αναπαραγώγιμα αποτελέσματα
- **Solution Validation**: Comprehensive checks για solution integrity
- **Error Recovery**: Graceful handling σε edge cases
- **Performance Monitoring**: Execution time και quality metrics
- **Memory Efficiency**: Optimized data structures για large problems

Coordinate Systems και Distance Calculations: Το σύστημα χρησιμοποιεί sophisticated coordinate transformation από γεωγραφικές συντεταγμένες (lat/lng) σε καρτεσιανές (x/y) για optimal Euclidean distance calculations.

4.17.6 Algorithm Progression Visualization - Professional Approach

Ο OR-Tools, σε αντίθεση με τους εκπαιδευτικούς αλγορίθμους του συστήματος (SOM, ACO, Pointer Network constructions), **δεν παρέχει detailed progression visualization**. Αυτή η απουσία αποτελεί στρατηγική επιλογή σχεδιασμού που αντικατοπτρίζει τη φιλοσοφία των professional optimization engines.

Λόγοι Minimal Visualization στον OR-Tools:

- **Production-Grade Focus**: Έμφαση στην τελική απόδοση παρά στην εκπαιδευτική διαδικασία
- **Performance Optimization**: Visualization overhead θα μείωνε την ταχύτητα εκτέλεσης
- **Complex Internal Operations**: Οι εσωτερικές λειτουργίες είναι πολύ σύνθετες για απλή visualization
- **Commercial Solver Philosophy**: Η Google εστιάζει στα αποτελέσματα, όχι στη διαδικασία
- **Benchmark Role**: Ως reference baseline, χρειάζεται ”καθαρά” αποτελέσματα

Professional Visualization Approach: Αντί για step-by-step animation, ο OR-Tools εμφανίζει τη λύση με professional styling:

- Clean, minimal numbered markers με dark blue-gray χρωματισμό
- Solid polyline path για production quality απεικόνιση
- Comprehensive performance metrics και solver information
- Professional-grade summary με algorithm suite details
- Results-focused presentation χωρίς educational overhead

Πίνακας 4.2: Visualization Philosophy: OR-Tools vs. Educational Algorithms

Aspect	**Educational Algorithms**	**OR-Tools Professional**
Progression Detail	Extensive step-by-step	Minimal, results-focused
Animation Support	Full interactive animation	Static final visualization
Learning Value	High educational content	Professional insight
Performance Impact	Visualization overhead	Zero overhead
Target Audience	Students and researchers	Production developers
Debugging Support	Full algorithm inspection	Black-box reliability

4.17.7 Comprehensive Output Structure και Metadata

Το σύστημα παρέχει εκτενή metadata για αποτελεσματική σύγκριση:

Professional Performance Summary:

- Solver framework information (Vehicle Routing Problem)
- Solution quality assessment (Professional-Grade Optimal)
- Execution time και distance metrics
- Algorithm suite details (GUIDED_LOCAL_SEARCH, PATH_CHEAPEST_ARC, etc.)
- Production readiness status και benchmark role

Technical Documentation και Integration: Κάθε εκτέλεση περιλαμβάνει comprehensive analysis με detailed explanation που περιγράφει τη διαδικασία VRP modeling, distance matrix creation, parameter configuration και metaheuristic application.

4.17.8 Εκπαιδευτική Αξία της Professional Approach

Παρόλο που ο OR-Tools δεν προσφέρει detailed progression, η minimal visualization έχει σημαντική εκπαιδευτική αξία:

Real-World Perspective:

- **Industry Standards**: Κατανόηση πώς λειτουργούν οι professional tools

- **Performance vs. Transparency Trade-off**: Επίδειξη του trade-off μεταξύ διαφάνειας και απόδοσης
- **Benchmark Reference**: Gold standard για σύγκριση με τους άλλους αλγορίθμους
- **Production Mindset**: Ανάπτυξη εστίασης στα αποτελέσματα παρά στη διαδικασία
- **Commercial Solver Insight**: Εισαγωγή στα standards των επαγγελματικών optimization tools

Τεχνική Αιτιολόγηση Professional Design: Η Google έχει σχεδιάσει τον OR-Tools ως production-ready solver που χρησιμοποιείται σε πραγματικές εφαρμογές (Google Maps, logistics), πρέπει να εκτελείται με μέγιστη ταχύτητα σε large-scale problems, και στοχεύει σε developers που χρειάζονται αξιόπιστα αποτελέσματα.

4.17.9 Θεωρητικό Υπόβαθρο και Αναφορές

Ο OR-Tools αποτελεί επίσημο προϊόν της Google με extensive documentation:

Κύριες τεχνικές αναφορές:

- **Constraint Programming**: Advanced CP-SAT solver integration
- **Vehicle Routing Framework**: Comprehensive VRP με TSP specialization
- **Metaheuristic Integration**: State-of-the-art local search algorithms
- **Production Scalability**: Tested σε real-world Google applications
- **Cross-Platform Support**: Consistent performance across platforms

Benchmark Value για το σύστημα:

- **Quality Baseline**: Professional-grade solution quality reference
- **Performance Target**: Production-level speed και efficiency standards
- **Algorithm Validation**: Verification των experimental implementations
- **Educational Contrast**: Demonstration διαφοράς research vs. production
- **Real-World Relevance**: Connection με actual industry applications

Σύγκριση με Educational Algorithms: Ο OR-Tools δεν προσφέρει detailed progression tracking όπως τα άλλα modules, αλλά αυτή η ”απλότητα” στο output αποτελεί στρατηγική επιλογή που δείχνει τη διαφορά μεταξύ research/educational implementations και production-grade professional tools που χρησιμοποιούνται στην πραγματικότητα από εταιρείες όπως η Google για την επίλυση complex optimization problems.

Η παρουσία του OR-Tools στο σύστημα παρέχει κρίσιμο context στους εκπαιδευτικούς αλγορίθμους, αποτελώντας το professional benchmark που επιτρέπει την αντικειμενική αξιολόγηση των άλλων προσεγγίσεων και την κατανόηση των industry standards για optimization performance.

4.18 Σύγκριση Αλγορίθμων TSP με και χωρίς Ξεκίνημα από την Ίδια Αρχή

Σε αυτή την ενότητα παρουσιάζεται μια εμπεριστατωμένη πειραματική αξιολόγηση τεσσάρων αλγορίθμων επίλυσης του προβλήματος του περιοδεύοντος πωλητή (TSP): **Ant Colony Optimization (ACO)**, **Self-Organizing Map (SOM)**, **OR-Tools** και **Pointer Network**. Στόχος της ανάλυσης είναι η σύγκριση της ποιότητας λύσης (συνολική διαδρομή), της ταχύτητας εκτέλεσης και της επίδρασης της επιλογής του σημείου εκκίνησης.

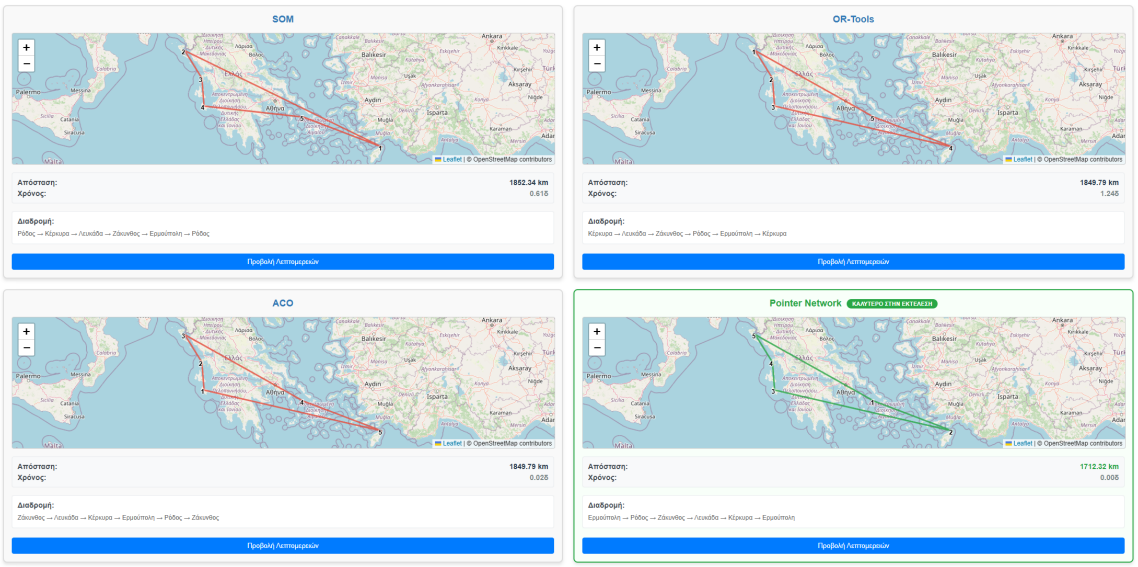
Πειραματικές Συνθήκες: Για κάθε αλγόριθμο και κάθε αριθμό πόλεων (5, 10, 20, 50), πραγματοποιήθηκαν δύο εκτελέσεις:

1. **Φυσική εκκίνηση (Natural Start):** Τυχαία επιλογή αρχικής πόλης.
2. **Ξεκίνημα από την ίδια αρχή:** Όλοι οι αλγόριθμοι ξεκινούν από την ίδια πόλη.

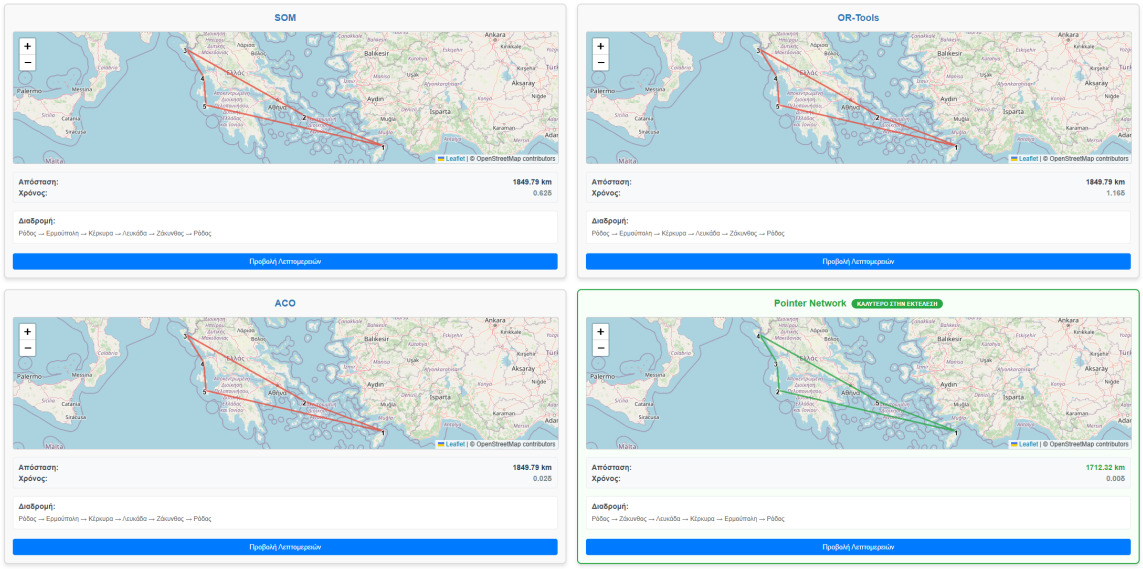
Μετρικές: Καταγράφηκαν:

- Η συνολική απόσταση της διαδρομής.
- Ο χρόνος εκτέλεσης.
- Η επίδραση της κοινής εκκίνησης στα αποτελέσματα.

5 Πόλεις



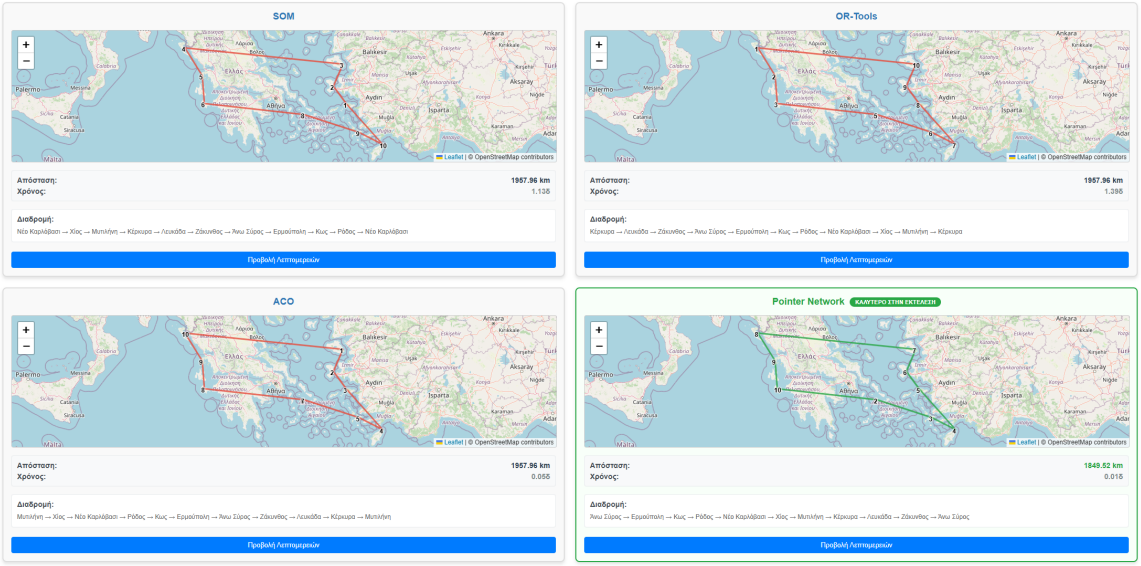
Σχήμα 4.16: Φυσική εκκίνηση. [53]



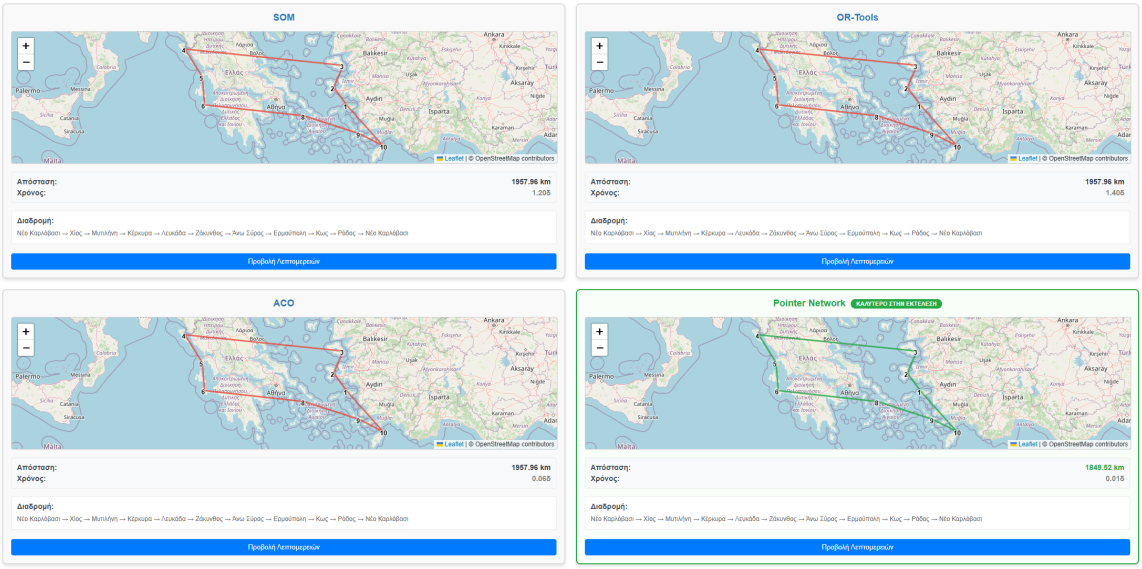
Σχήμα 4.17: Ξεκίνημα από την ίδια αρχή. [53]

Ο Pointer Network παρήγαγε την πιο σύντομη διαδρομή (1712.32 km), με ACO και OR-Tools να ισοβαθμούν (1849.79 km). Ο SOM είχε ελαφρώς μεγαλύτερη απόσταση, αλλά ωφελήθηκε οριακά από την κοινή αρχή. Η αλλαγή εκκίνησης είχε αμελητέα επίδραση.

10 Πόλεις



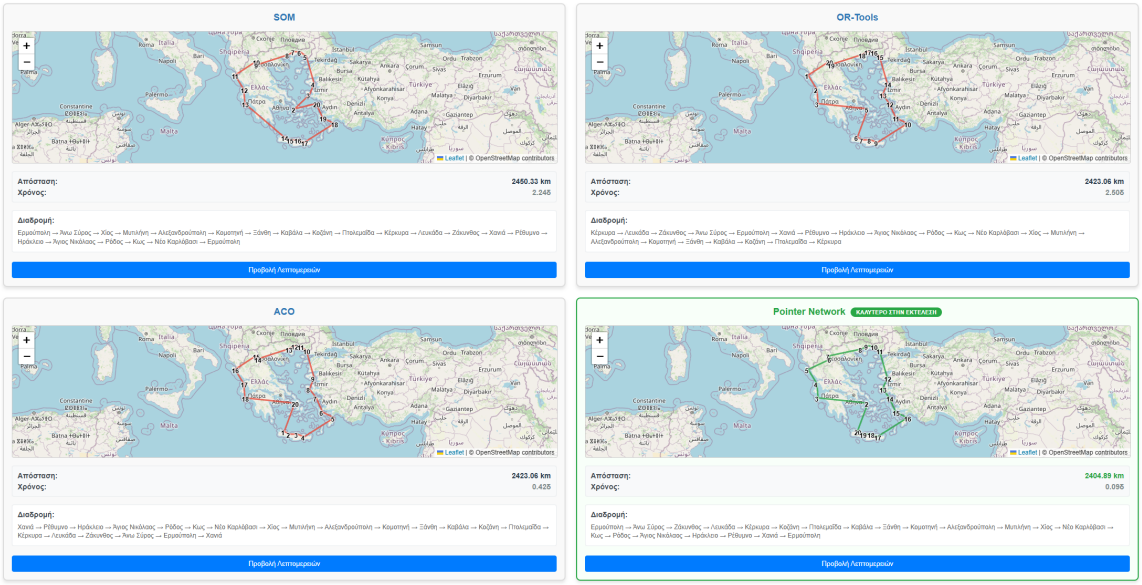
Σχήμα 4.18: Φυσική εκκίνηση. [53]



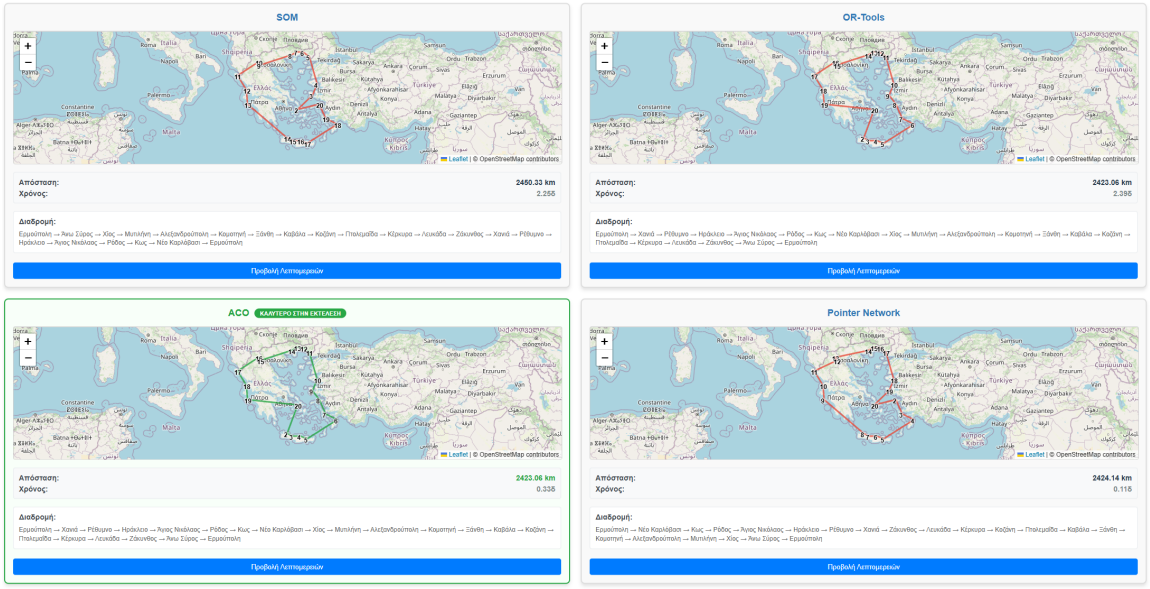
Σχήμα 4.19: Ξεκίνημα από την ίδια αρχή. [53]

Ο Pointer Network παρέμεινε πρώτος (1849.52 km), με τους υπόλοιπους να αποδίδουν ταυτόσημα (1957.96 km). Η επίδραση της κοινής εκκίνησης ήταν μηδενική.

20 Πόλεις



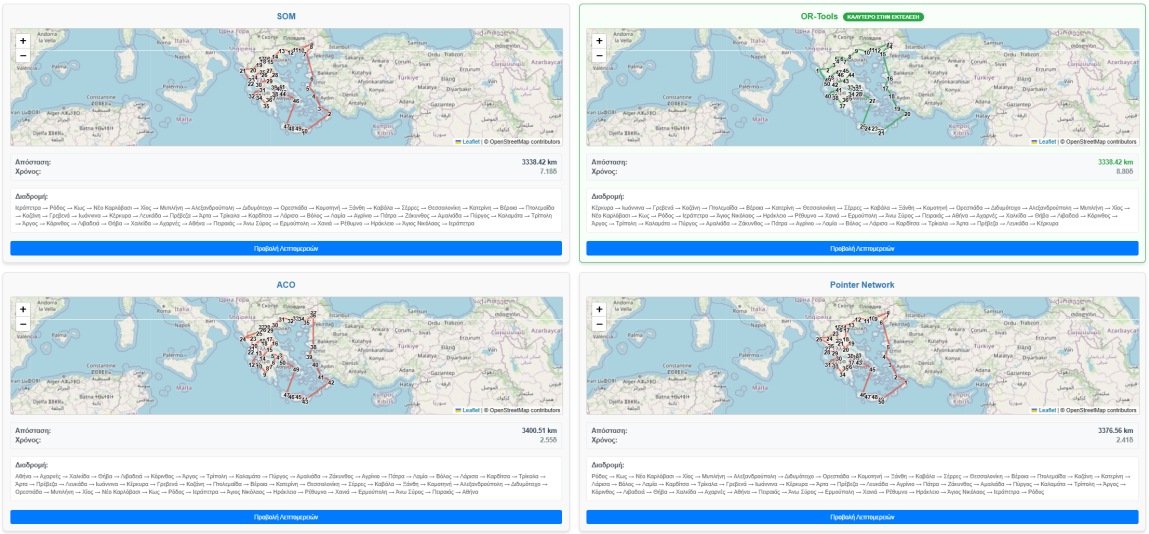
Σχήμα 4.20: Φυσική εκκίνηση. [53]



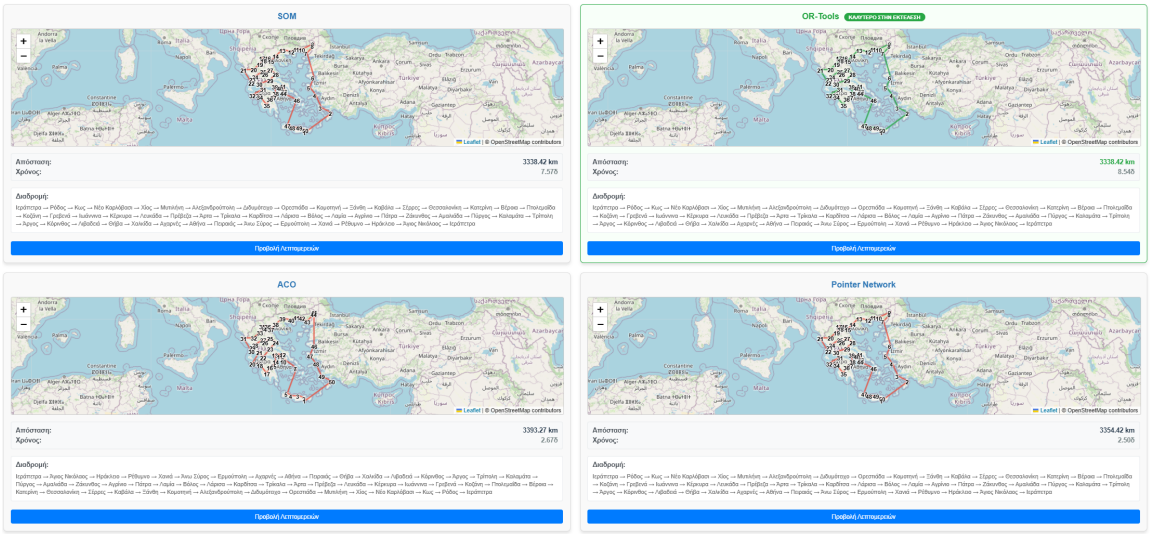
Σχήμα 4.21: Ξεκίνημα από την ίδια αρχή. [53]

Ο Pointer Network υπερίσχυσε στη φυσική εκκίνηση, αλλά στην κοινή αρχή ο ACO παρουσίασε μικρή βελτίωση που τον έφερε πρώτο. Εδώ, η αλλαγή αρχής επηρέασε την τελική κατάταξη.

50 Πόλεις



Σχήμα 4.22: Φυσική εκκίνηση. [53]



Σχήμα 4.23: Ξεκίνημα από την ίδια αρχή. [53]

Ο OR-Tools αποδείχθηκε σταθερά ο πιο αποδοτικός αλγόριθμος. Ο Pointer Network ήταν κοντά, αλλά επηρεάστηκε ελαφρώς αρνητικά από την κοινή αρχή. Ο SOM είχε το μεγαλύτερο χρόνο εκτέλεσης, ενώ ο ACO εμφάνισε ελαφρά βελτίωση.

Συνολικά Συμπεράσματα:

- Ο **Pointer Network** επικράτησε σε μικρά και μεσαία προβλήματα TSP (5–30 πόλεις), με εξαιρετικά σταθερή και γρήγορη απόδοση.
- Ο **ACO** αποδίδει καλύτερα σε επιλεγμένες περιπτώσεις, αλλά η συνολική του απόδοση είναι ασταθής.
- Ο **OR-Tools** αποδείχθηκε πιο αποτελεσματικός σε μεγάλα προβλήματα (40+ πόλεις), προσφέροντας καλή ισορροπία ακρίβειας και σταθερότητας.

- Το **ξεκίνημα από την ίδια αρχή** είχε ελάχιστη ή περιορισμένη επίδραση σε μικρά προβλήματα, αλλά επηρέασε περισσότερο από τις 20 πόλεις και άνω.
- Ο **SOM** δεν αποτελεί ιδιαίτερα αποδοτικό αλγόριθμο TSP αλλά μπορεί να προσφέρει χρήσιμη αρχική τοποθέτηση για άλλους αλγορίθμους.

4.18.1 Συνολική Αξιολόγηση

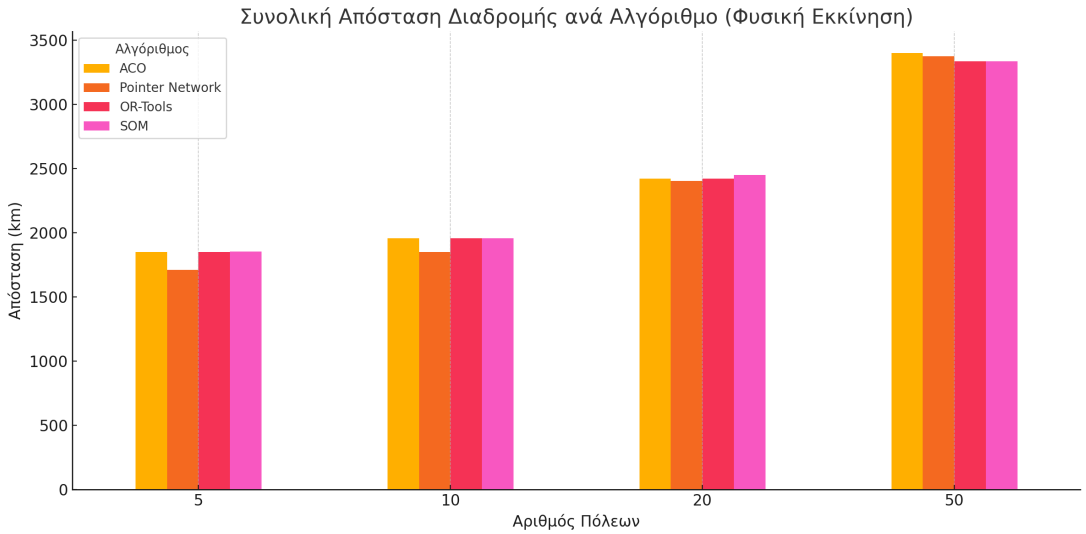
Από τη μελέτη των παραπάνω αποτελεσμάτων προκύπτει ότι ο **Pointer Network** αποτελεί τη βέλτιστη λύση για προβλήματα TSP μικρού έως μεσαίου μεγέθους, συνδυάζοντας εξαιρετική ακρίβεια και υπολογιστική ταχύτητα. Αντιθέτως, ο **OR-Tools** υπερέχει σε πολύ μεγάλα προβλήματα, προσφέροντας υψηλή ακρίβεια με αυξημένο όμως κόστος χρόνου. Ο **ACO** είναι ελαφρύς και αποτελεσματικός σε μικρές περιπτώσεις, ενώ ο **SOM**, παρότι λιγότερο ακριβής, παρέχει χρήσιμες αρχικές θέσεις και θα μπορούσε να λειτουργήσει επικουρικά σε υβριδικές προσεγγίσεις. Το **ξεκίνημα από κοινή αρχή** είχε περιορισμένη επίδραση σε μικρές περιπτώσεις αλλά επηρέασε τις κατατάξεις σε πιο πολύπλοκα σενάρια.

Πίνακας 4.3: Συνολικές Αποστάσεις Διαδρομής (km) – Φυσική Εκκίνηση

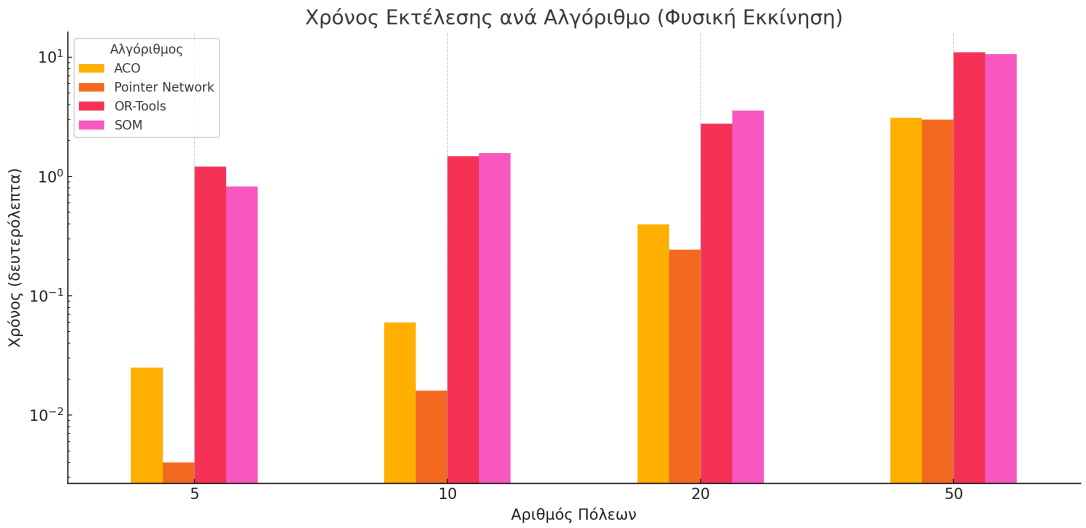
Πόλεις	ACO	Pointer Network	OR-Tools	SOM
5	1849.79	**1712.32**	1849.79	1852.34
10	1957.96	**1849.52**	1957.96	1957.96
20	2423.06	**2404.89**	2423.06	2450.33
50	3400.51	3376.56	**3338.42**	3338.42

Πίνακας 4.4: Χρόνοι Εκτέλεσης (δευτερόλεπτα) – Φυσική Εκκίνηση

Πόλεις	ACO	Pointer Network	OR-Tools	SOM
5	0.025	**0.004**	1.213	0.822
10	0.060	**0.016**	1.476	1.577
20	0.395	**0.244**	2.775	3.567
50	3.087	**2.989**	10.960	10.629



Σχήμα 4.24: Συνολική Απόσταση Διαδρομής Ανά Αλγόριθμο (Φυσική Εκκίνηση) [54]



Σχήμα 4.25: Χρόνος Εκτέλεσης Ανά Αλγόριθμο (Φυσική Εκκίνηση) [54]

Κεφάλαιο 5ο: Συμπεράσματα

Η παρούσα εργασία πραγματεύθηκε τη μελέτη, υλοποίηση και πειραματική αξιολόγηση τεσσάρων διαφορετικών αλγορίθμων για την επίλυση του κλασικού προβλήματος του Περιοδεύοντος Πωλητή (TSP). Οι αλγόριθμοι που μελετήθηκαν — Ant Colony Optimization (ACO), Self-Organizing Map (SOM), OR-Tools και Pointer Network — εκπροσωπούν διαφορετικές τεχνολογικές και υπολογιστικές προσεγγίσεις: από μεθόδους εμπνευσμένες από τη φύση έως νευρωνικά δίκτυα και state-of-the-art βελτιστοποιητές.

Η αξιολόγηση πραγματοποιήθηκε σε τέσσερα μεγέθη προβλημάτων (5, 10, 20 και 50 πόλεις), λαμβάνοντας υπόψη δύο σενάρια: εκκίνηση από τυχαίο σημείο και εκκίνηση από κοινή αρχή για όλους τους αλγορίθμους. Οι μετρικές που εξετάστηκαν περιελάμβαναν τη συνολική απόσταση της διαδρομής και τον χρόνο εκτέλεσης κάθε μεθόδου.

Συνολικές Παρατηρήσεις:

- Ο **Pointer Network** παρουσίασε κορυφαία απόδοση σε προβλήματα μικρού και μεσαίου μεγέθους (έως 30 πόλεις), τόσο ως προς την ακρίβεια όσο και την ταχύτητα.
- Ο **OR-Tools**, παρά το αυξημένο υπολογιστικό του κόστος, ανέδειξε την καλύτερη επίδοση σε προβλήματα με 50 πόλεις, επιτυγχάνοντας τις πιο σύντομες διαδρομές.
- Ο **ACO** προσφέρει ένα αποδοτικό σημείο ισορροπίας για μικρότερα προβλήματα, με ελάχιστους υπολογιστικούς πόρους, αλλά μειώνεται η αποτελεσματικότητά του σε μεγαλύτερα μεγέθη.
- Ο **SOM**, αν και δεν αποτελεί αποδοτικό τρόπο επίλυσης του TSP, μπορεί να αξιοποιηθεί ως εργαλείο αρχικοποίησης ή υποβοήθησης σε υβριδικές προσεγγίσεις.
- Η **εκκίνηση από κοινή αρχή** δεν είχε ουσιαστική επίδραση σε απλά προβλήματα, αλλά απέδειξε ότι μπορεί να αλλάξει τη σειρά κατάταξης αλγορίθμων σε πιο σύνθετα σενάρια, αναδεικνύοντας τη σημασία της αρχικοποίησης.

Συνολικά, η εργασία ανέδειξε τη διαφορετική συμπεριφορά κάθε αλγορίθμου ανάλογα με το μέγεθος και τις παραμέτρους του προβλήματος. Η συγκριτική ανάλυση που παρουσιάστηκε μπορεί να λειτουργήσει ως οδηγός επιλογής κατάλληλου αλγορίθμου σε πρακτικές εφαρμογές δρομολόγησης και εφοδιαστικής αλυσίδας.

Αναφορές

[1] Md Zahangir Alom, Tarek M. Taha, Chris Yakopcic, Scott Westberg, and Vijayan K. Asari. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3):292, 2019.

[2] David L. Applegate, Robert E. Bixby, Vasek Chvátal, and William J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

[3] S. Bajpai, M. Aono, and P. Kurian. Tracking and distinguishing slime mold solutions to the traveling salesman problem through synchronized amplification in the non-equilibrium steady state. *arXiv preprint arXiv:2504.03492*, 2025.

[4] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. In *ICLR Workshop*, 2017.

[5] Rodrigo Canário. Do we still need backpropagation to train neural networks?, May 2025. Accessed: 2025-05-31.

[6] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[7] Codersarts. Mlp classifier in python, December 2019. Machine Learning Tutorial.

[8] Jean-Claude Fort. Solving a combinatorial problem via self-organizing process: An application of the kohonen algorithm to the traveling salesman problem. *Biological Cybernetics*, 59:33–40, 1988.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[10] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649, 2013.

[11] Gregory Gutin and Abraham P. Punnen. *The Traveling Salesman Problem and Its Variations*. Springer US, 2002.

[12] Michael Held and Richard M. Karp. Dynamic programming treatment of the travelling salesman problem. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962.

[13] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[15] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[16] C. K. Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the traveling salesman problem. *arXiv preprint arXiv:1906.01227*, 2019.

[17] C.K. Joshi, T. Laurent, and X. Bresson. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv preprint*, 2019.

[18] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2017.

[19] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[20] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[21] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! *ICLR*, 2019.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[23] R. Kumar. A survey on memetic algorithm and machine learning approach to traveling salesman problem. *International Journal of Emerging Technologies*, 2020.

[24] Young-Dong Kwon, Jaegul Choo, Byoungjip Kim, and Il-Chul Yoon. Pomo: Policy optimization with multiple optima for reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. NeurIPS 2020.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[27] H. Liang, S. Wang, L. Zhou, and X. Zhang. Bignn: Bipartite graph neural network with attention mechanism for solving multiple traveling salesman problems in urban logistics. *International Journal of Applied Earth Observation and Geoinformation*, 2024.

[28] H. Liu, M.C. Zhou, and Z. Zhang. Solving dynamic traveling salesman problems with deep reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[29] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[30] Richard C. Millar, Leila Hashemi, Armin Mahmoodi, Robert W. Meyer, and Jeremy Laliberte. Integrating unmanned and manned uavs data network based on combined bayesian belief network and multi-objective reinforcement learning algorithm. *Drone Systems and Applications*, 11:1–17, 2023.

[31] C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

[32] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[33] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[34] T. Nakaguchi, K. Jin'no, and M. Tanaka. Hysteresis neural networks for solving traveling salesperson problems. In *2000 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages IV–565–IV–568. IEEE, 2000.

[35] OpenStreetMap contributors. OpenStreetMap. `https://www.openstreetmap.org/`, 2024. Accessed: 2025-01-01.

[36] J. Perera, S.H. Liu, and M. Črepinšek. A graph pointer network-based multi-objective deep reinforcement learning algorithm for solving the traveling salesman problem. *Mathematics*, 11(2), 2023.

[37] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[39] S. Sanyal, S. Yoo, A. Holla, D.E. Kim, and F. Iacopi. Taxi: Traveling salesman problem accelerator with x-bar-based ising macros powered by sot-mrams and hierarchical clustering. *arXiv preprint arXiv:2504.13294*, 2025.

[40] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[41] Hanmei Yao Shenshen Gu, Tao Hao. A pointer network based deep learning algorithm for unconstrained binary quadratic programming problem. *Neurocomputing*, 2020.

[42] M. Soh and A.N. Likeufack. A new hybrid algorithm based on ant colony optimization and recurrent neural networks with attention mechanism for solving the traveling salesman problem. *Revue Africaine de Recherche en Informatique et Mathématiques Appliquées (ARIMA)*, 32, 2025.

[43] R. Sohiburoyyan and S.N. Himawan. Graph neural network-variable neighborhood search for solving symmetric traveling salesman problem. In *2024 7th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. IEEE, 2024.

[44] Examples of som algorithm mechanism to solve tsp. ResearchGate. Figure 4, Ανακτήθηκε στις 9 Ιανουαρίου 2025.

[45] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[46] David W. Tank and John J. Hopfield. Simple 'neural' optimization networks: An a/d converter, signal decision circuit, and a linear programming circuit. *IEEE Transactions on Circuits and Systems*, CAS-33(5):533–541, 1986.

[47] I. Tokuda, T. Nagashima, and K. Aihara. Global bifurcation structure of chaotic neural networks and its application to traveling salesman problems. *Neural Networks*, 10(9):1675–1684, 1997.

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[49] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *Advances in neural information processing systems*, 28:2692–2700, 2015.

[50] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Elias Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

[51] Wikipedia. Convolutional neural network, 2025. Ανακτήθηκε στις 9 Ιανουαρίου 2025.

[52] H. Xincheng and S. Yuzhuo. A method of the traveling salesman problem based on q-learning reinforcement learning and local dynamic programming. In *2024 21st International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE, 2024.

[53] Alexandros Zacharegkas. Interactive tsp solver application. Προσωπική ανάπτυξη, 2025. Διπλωματική εργασία, Πανεπιστήμιο ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ.

[54] Alexandros Zacharegkas. Ανάλυση και Υλοποίηση Αλγορίθμων για το Πρόβλημα του Πλανόδιου Πωλητή. Πτυχιακή εργασία, ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ, 2025.

[55] Jin-Xi Zhang, Hangyi Zhao, and Xuefeng Zhang. Universal approximation property of stochastic configuration networks for time series. *Industrial Artificial Intelligence*, 2(3), March 2024. Open Access.

[56] Peng Zhang. The diagram of a hopfield neural network with four neurons. ResearchGate. Figure 7, Ανακτήθηκε στις 9 Ιανουαρίου 2025.

[57] J. Zhao, K.H. Cheong, and W. Pedrycz. Bridging visualization and optimization: Multimodal large language models on graph-structured combinatorial optimization. *arXiv preprint arXiv:2501.11968*, 2025.