Πανεπιστήμιο Θεσσαλίας

Σχολή Θετικών Επιστημών

Τμήμα Πληροφορικής με εφαρμογές στην Βιοϊατρική

Δίκτυα Υπολογιστών

Σύνδεση κινητών χρηστών με APs

Εκπονήθηκε από τον:

Κρικέλη Λάμπρο

2023-2024

13 Νοεμβρίου 2023

Περίληψη

Τα δίκτυα υπολογιστών, είναι αναπόσπαστο μέρος της σύγχρονης επικοινωνίας, εκτείνονται σε διάφορες μορφές όπως LAN, WAN και PAN. Τα σημεία πρόσβασης (AP) είναι ζωτικής σημασίας στην ασύρματη συνδεσιμότητα, λειτουργώντας ως φύλακες για συσκευές.Τα ασύρματα πρότυπα, τα πρωτόκολλα ασφαλείας και η στρατηγική τοποθέτηση ΑΠ είναι κρίσιμα ζητήματα. Οι προκλήσεις περιλαμβάνουν παρεμβολές, ασφάλεια, επεκτασιμότητα και αποτελεσματική διαχείριση.

***Λέξεις-Κλειδιά**: Δίκτυα Υπολογιστών, Wi-FI AP,Mobile Users*

Περιεχόμενα

1 Εισαγωγή

Στην παρακάτω υλοποίηση επεξεργαζόμαστε δεδομένα από μια ασύρματη επικοινωνία μεταξύ κινητών χρηστών και σταθερών σημείων προσβάσεις ¨κόμβων¨. Για το συγκεκριμένο παράδειγμα ,για ευκολία, καθορίζουμε ότι ο χρήστης θα συνδεθεί στο πιο κοντινό Wi-Fi AP το οποίο αυτόματα αποδεχόμαστε πως είναι και αυτό που εκπέμπει και το πιο ισχυρό σήμα. Η πρώτη γραμμή περιέχει τον αριθμό M των κινητών χρηστών και τον αριθμό K των APs. Οι επόμενες M γραμμές περιέχουν τις συντεταγμένες (x_i, y_i), $i = 1, \ldots, M.$ των κινητών στο χώρο.

2 Υλοποίηση

Για την παρακάτω υλοποίηση χρησιμοποίησα τη γλώσσα προγραμματισμού python. Αρχικά ανοίγω το .txt αρχείο το οποίο περιέχει όλα μου τα δεδομένα με τη χρήση της εντολης open().Τα δεδομένα στο αρχείο είναι σε διανυσματική μορφή (x,y) έτσι για να μπορώ να τα επεξεργαστώ στην ευκλείδεια απόσταση

$$d_{i,j} = \sqrt{(x_i - a_j)^2 + (y_i - b_j)^2}$$

Θα έπρεπε να επεξεργαστώ τα δεδομένα με έναν τρόπο τέτοιο ώστε να μπορώ να διαχειριστώ την σχέση από πάνω .Αυτό μου επιτρέπεται μέσα απ΄τη συνάρτηση split η οποία χωρίζει τα δεδομένα σε (x_i, y_i). Όπως βλέπουμε παρακάτω.

```
open("MobileData.txt", "r")
M, K = map(int, file.readline().split())
```

Η python περιεχέι αυτόματα αυτές τις συναρτήσεις για να τις χρησιμοποιήσεις.

- Από τη γραμμή 20 έως και τη 43 υλοποιώ το 1ο ερώτημα.
- Με τη συνάρτηση calculate-sum-of-distances γραμμές 12-22 υπολογίζω το άθροισμα της ελάχιστης απόστασης των χρηστών με τους κόμβους.
- Από τη γραμμή 71 και κάτω υπάρχει η υλοποίηση της οπτικοποίηση των κόμβων.

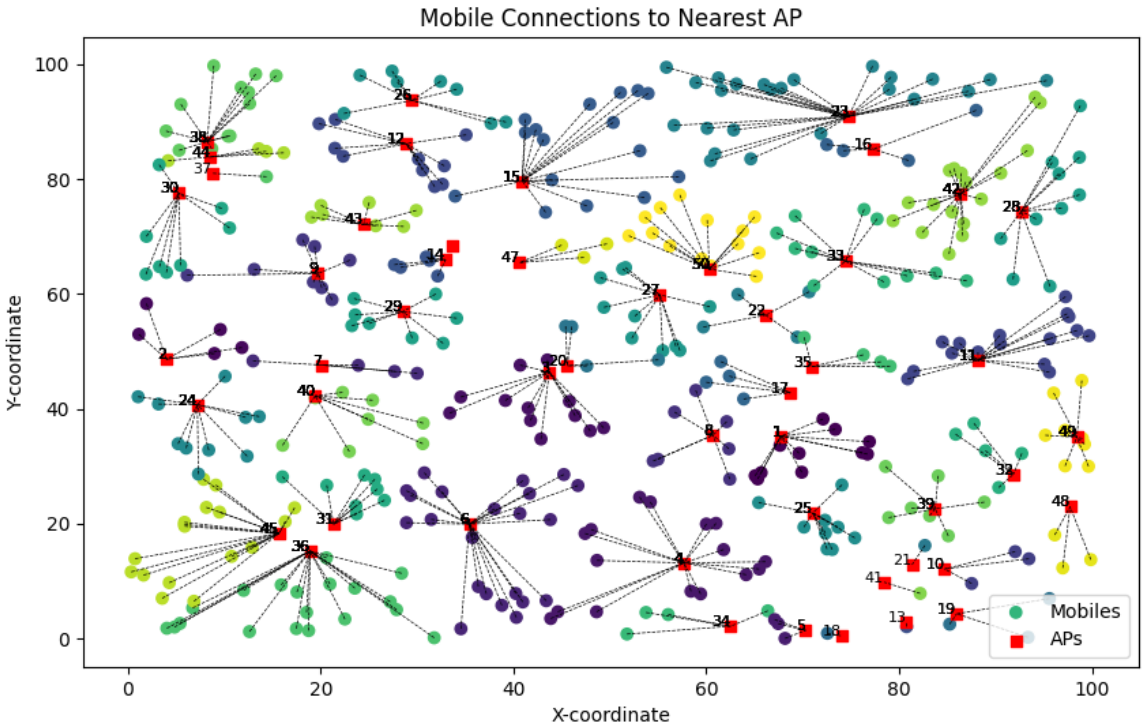
Ολόκληρος ο κώδικας υπάρχει στη βιβλιογραφία [3]

3 Αποτελέσματα

Με τον τρόπο που έχουμε ορίσει το πρόβλημα στην εισαγωγή της αναφοράς τα αποτελέσματα από το αρχείο MobileData.txt έχουν διαμορφωθεί ως εξής:

```
lamrpos@UbuntuVm:~/Documents/Connecting Mobile Users$ python3 com_ne
t1.py
We have 50 Vectors/Access Points
Each Access point containing the following number of users on them:
[11, 5, 12, 16, 3, 22, 4, 7, 8, 3, 16, 10, 1, 5, 16, 4, 4, 1, 3, 4,
1, 4, 24, 10, 9, 8, 10, 10, 8, 8, 8, 5, 11, 4, 4, 19, 1, 12, 7, 8, 1
, 17, 7, 4, 17, 0, 3, 3, 8, 14]
Sum of Minimum Distances for All Mobiles: 3209.8439599283024
Minimum AP Load: AP 46 with Load 0
Maximum AP Load: AP 23 with Load 24
```

Το Min and Max Load υπάρχει ως output διότι με αυτό αρχίζει η λύση που προτείνω στο προαιρετικό ερώτημα το οποίο βάλατε. Πιο αναλυτικά στη παράγραφο 4!



Οπτικοποίηση Αποτελεσμάτων.

4 Προαιρετικό Ερώτημα

Αυτό που θα σκεφτόμουν να κάνω είναι για αρχή να βρω το φορτίο που έχει κάθε κόμβος , access point, έπειτα να μετρήσω εάν υπάρχει overload σε κάποιον από αυτούς τους κόμβους. Διότι εάν υπάρχει overload σημαίνει πως καθυστερεί σιγουρά να ¨απαντήσει¨ σε καποιον χρήστη. Για παράδειγμα εάν ο κόμβος που πάει να συνδεθεί ο χρήστης έχει overload τότε το σύστημα να τον στέλνει στον αμέσως επόμενο κοντινότερο από την στιγμή που έχουμε αποδεκτή ότι όσο πιο κοντινή απόσταση σημαίνει τόσο και πιο ισχυρό σήμα. Το overload θα μπορούσα να το ορίσουμε αυθαίρετα για οικονομία χρόνου άπλα με ένα thresh hold !

Αναφορές

[1] Τσουκάτος, Κ. (2023-2024). *Δίκτυα Υπολογιστών* [Πανεπιστημιακές Σημειώσεις]. Πανεπιστήμιο Θεσσαλίας, Τμήμα Πληροφορικής με Εφαρμογές στη Βιοϊατρική, Μάθημα ¨Δίκτυα Υπολογιστών¨, Χειμερινό Εξάμηνο 2023-2024, Λαμία!

[2] `https://stackoverflow.com/questions/55193968/minimum-and-maximum-sums-from-a-list-python`

[3] Κώδικας Υλοποιήσεις 1ης εργασίας! imported from com.net1.py

```
# ~UTH.gr
# Department of Computer Science and Biomedical Informatics
# Subject Communication networks
# Krikelis Lampros

import math
import matplotlib.pyplot as plt

def calculate_distance(x1, y1, x2, y2):
    return math.sqrt((x1 - x2)**2 + (y1 - y2)**2)

def find_nearest_ap(mobile_position, ap_positions):
    distances = [calculate_distance(mobile_position[0],
       mobile_position[1], ap_position[0], ap_position[1])
       for ap_position in ap_positions]
    nearest_ap_index = distances.index(min(distances))
    return nearest_ap_index, min(distances)

def calculate_sum_of_distances(mobile_positions,
   ap_positions):
    # Calculate the minimum distances for all mobiles
    min_distances = [find_nearest_ap(mobile_position,
       ap_positions)[1] for mobile_position in
       mobile_positions]

    # Calculate and return the sum of the minimum distances
    return sum(min_distances)

def main():
```

```
# ~~~Here I am woring on the file and intialize  the
   mobile users and the Access points
# Load input data from MobileData.txt
with open("MobileData.txt", "r") as file:
    # Read the first line containing M (number of
       mobile users) and K (number of APs)
    M, K = map(int, file.readline().split())

    ap_loads = [0] * K

    # Read mobile positions
    mobile_positions = [list(map(float,
       file.readline().split())) for _ in range(M)]

    # Read access point positions
    ap_positions = [list(map(float,
       file.readline().split())) for _ in range(K)]

    # Vector n containing the number of cells assigned
       to each AP
    connected_mobiles =
       [find_nearest_ap(mobile_position,
       ap_positions)[0] for mobile_position in
       mobile_positions]
    vector_n = [connected_mobiles.count(j) for j in
       range(K)]
    print(f"We␣have␣{len(vector_n)}␣Vectors/Access␣
       Points\nEach␣Access␣point␣containing␣the␣
       following␣number␣of␣users␣on␣them:␣{vector_n}␣")

    # Calculate the sum of minimum distances for all
       mobiles
    sum_of_distances =
       calculate_sum_of_distances(mobile_positions,
       ap_positions)
    print(f"Sum␣of␣Minimum␣Distances␣for␣All␣Mobiles:␣
       {sum_of_distances}")


    # ~~~~ Here I am storying the calculated distance
```

```
    in the ap loads.

for i, mobile_position in
    enumerate(mobile_positions, start=1):
     nearest_ap_index, min_distance =
        find_nearest_ap(mobile_position,
        ap_positions)
     # print(f"Mobile User {i}: Nearest AP Index =
        {nearest_ap_index + 1}, Distance =
        {min_distance}")

     #  load on the connected AP
     ap_loads[nearest_ap_index] += 1
     # print(f"Load on Each AP: {ap_loads}")


# Find and print the minimum and maximum AP loads
min_load_ap_index = ap_loads.index(min(ap_loads))
    + 1
max_load_ap_index = ap_loads.index(max(ap_loads))
    + 1

min_load = min(ap_loads)
max_load = max(ap_loads)

print(f"Minimum␣AP␣Load:␣AP␣{min_load_ap_index}␣
    with␣Load␣{min_load}")
print(f"Maximum␣AP␣Load:␣AP␣{max_load_ap_index}␣
    with␣Load␣{max_load}")

# Visualize the usage of access points with a
    scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(*zip(*mobile_positions),
    c=connected_mobiles, cmap='viridis',
    label='Mobiles', marker='o')
plt.scatter(*zip(*ap_positions), c='red',
    label='APs', marker='s')

# Draw lines connecting mobiles to their nearest
```

```
            APs
        for i, mobile_position in
            enumerate(mobile_positions, start=1):
            nearest_ap_index, _ =
                find_nearest_ap(mobile_position,
                ap_positions)
            plt.plot([mobile_position[0],
                ap_positions[nearest_ap_index][0]],
                [mobile_position[1],
                ap_positions[nearest_ap_index][1]],
                color='black', linestyle='dashed',
                linewidth=0.5)

            # Adding a label next to each Ap
            plt.text(ap_positions[nearest_ap_index][0],
                ap_positions[nearest_ap_index][1],
                f'{nearest_ap_index + 1}', fontsize=8,
                ha='right')


        plt.xlabel('X-coordinate')
        plt.ylabel('Y-coordinate')
        plt.title('Mobile Connections to Nearest AP')
        plt.legend()
        plt.show()

if __name__ == "__main__":
    main()
```