Sveučilište u Rijeci
Tehnički Fakultet Rijeka

Bežične mreže osjetila

Zadatak:

Implementirati algoritam za očitanje maksimalne vrijednosti temperature u mreži.

- Čvorove mreže opremiti temperaturnim senzorom (napraviti klasu)
    - Poslužite se kodom postojećih senzora i dokumentacijom
- Osmisliti i implementirati algoritam.
    - Algoritam mora biti raspodijeljen, a čvorovi mogu koristiti samo informacije dobivene očitanjem senzora i komunikacijom
- Eksperimentalno i analitički analizirati memorijsku i vremensku složenost algoritma.
    - Eksperimentalna analiza sastoji se od ispitivanja algoritma u različitim tipovima mreža, varira se broj čvorova povezanost promjere i slični parametri

Algoritam

Napravljen je algoritam za očitanje maksimalne temperature u mreži unutar *broadcast.py* skripte (klasa *MaxTemperature*). Temperaturni senzor implementiran je unutar *sensor.py* skripte (klasa *TemperatureSensor*). *TemperatureSensor* klasa sastoji se od *read* metode koja vraća random broj između -50 i 50. Unutar *global_settings.py* skripte temperaturni senzor postavljen je kao defaultni senzor svih čvorova mreže (uz ranije postavljeni *NeighborsSensor*). Skriptna *find_max_temp.py* sadrži kod pomoću kojeg je generirana random mreža, određen joj je broj čvorova te dodijeljen MaxTemperature algoritam te je čvor na indeksu 0 postavljen je kao INITIATOR s porukom “Which is your temperature?”. Razmjena poruka razlikuje se prema 2 vrste header-a unutar poruke (klasa *Message*):

- ‘Information’ - poruka kojem se susjeda pita njegova temperatura
- ‘Max Temperature’ - poruka koja označava da se trenutno najveća temperatura šalje unatrag prema INITIATOR čvoru

MaxTemperature klasa sastoji se od 4 metode: Initializer, Initiator, Idle i Done.

MaxTemperature klasa

Initializer metoda

- svakom čvoru unutar mreže u određene registre zapišu se njegovi susjedi i temperatura
- svaki čvor osim INITIATOR čvora postavi se u stanje IDLE
- svakom čvoru se u odgovaralući registar zapiše broj njegovih susjeda te broj primljenih poruka koji se inicijalno postavi na 0
- svakom čvoru postavi se vrijednost registra *isReceived* na *false* što označava da nije primio poruku

- spontani impuls pokreće INITIATOR čvor i zapisuje se njegova trenutna temperatura u njegov registar kao trenutno najveća u mreži

Initiator metoda
- INITIATOR pita sve svoje susjede njihovu temperaturu
- ostaje u stanju INITIATOR dok ne primi natrag odgovore od svih susjeda (temperature svih svojih susjeda zaprimit će kada broj primljenih poruka bude odgovarao broju njegovih susjeda)
- svaki put kada INITIATOR primi odgovor od svojih susjeda
    - ako je primio temperaturu (header poruke je ‘MaxTemperature’) usporedi dobivenu temperaturu s temperaturom zapisanom u memoriji čvora te spremi veću. Nakon toga povećaj broj primljenih poruka za 1.
- kada INITIATOR primi odgovor od svih svojih susjeda postaje DONE

Idle metoda
- svaki čvor osim INITIATOR čvora inicijalno je IDLE
- IDLE čvor zaprimi upit za temperaturom i prosljeđuje ga dalje svojim susjedima (koji su u stanju IDLE) iz kojih isključujemo INITIATOR čvor i čvor koji je poslao poruku (prethodnik trenutnog čvora)
    - ako je čvor proslijedio poruku ostaje u stanju IDLE
    - ako IDLE čvor nema susjeda kojima može proslijediti upit šalje poruku unazad (prema INITIATOR čvoru) i postaje DONE

Done metoda
- stanje u kojemu je čvor u finalnom stanju i ne obavlja akcije

Pseudokod

Status values: S = {INITIATOR, IDLE, DONE}
S_{INIT} = {INITIATOR, IDLE}
S_{START} = {INITIATOR}
S_{TERM} = {DONE}
S_{FINAL} = {DONE}

Spontaneously - spontani impuls koji pokreće slanje poruka
Send(M)- šalji poruku M (upit za temperaturu)
N(x) - susjedi trenutnog čvora
N_{source} - prethodnik trenutnog čvora
Receiving(MaxTemp) - primitak trenutno najveće temperature
Receiving(M) - primitak upita za temperaturu
Receiving(none) - primitak poruke koja ne sadrži temperaturu
SaveMaxTemp(MaxTemp)- usporedi i zapiši temperaturu u registar trenutnog čvora
SaveTemperature(M) - usporedi i zapamti veću temperaturu

INITIATOR

```
Spontaneously
begin
      send(M) to N(x);
end

Receiving(MaxTemp):
      begin
            SaveMaxTemp(MaxTemp)
            numReceivedMessages++
            if  numReceivedMessages == numNeighbours
                  become DONE
      end
```

IDLE

```
Receiving(M)
      begin
            SaveTemperature(M)
            N(x) = N(x) - sender - INITIATOR

            if N(x)
                  send(M) to N(x)
                  become IDLE
            else
                  send(M) to N_source
                  become DONE
      end
```

DONE

```
pass
```

Analiza algoritma

Analitička analiza

n - broj čvorova
m - broj bridova
d(G)-promjer mreže

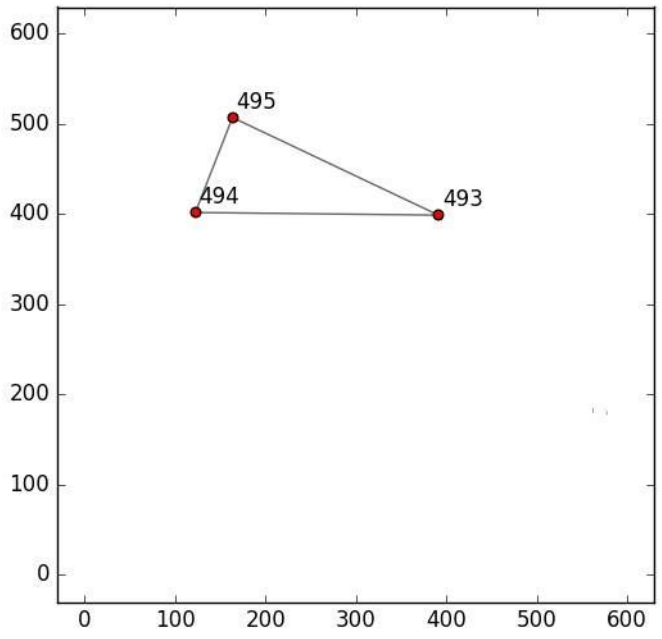
Broj poruka:
M[MaxTemperature] = 2*m = O(n)

Vremenska složenost:
T[MaxTemperature] <= 2*d(G) = O(n)

Eksperimentalna analiza

Slika 1 prikazuje mrežu koja se sastoji od 3 čvora. Sa slike vidimo i da je broj bridova jednak 3. Broj razmjenjenih poruka u mreži prilikom izvođenja algoritma je 6, odnosno broj poruka u najgorem slučaju za ovaj primjer je O(n) = 2*m = 6. Također najdulji najkraći put unutar mreže (promjer) je 1, a budući da se poruke šalju u 2 smjera vremenska složenost za ovu mrežu je 4, tj. formula za ovu sliku je: O(n) = 2*d(G) = 2.

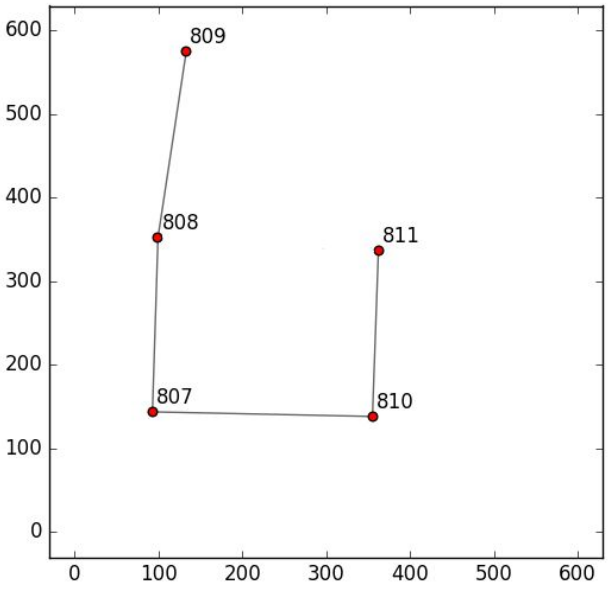
- INITIATOR - čvor 493
- n = 3
- m = 3
- Broj razmjenjenih poruka: 6
- Promjer: 1



Slika 1: Mreža s 3 čvora

Slika 2 prikazuje mrežu povezanu u lanac koja se sastoji od 5 čvorova i 4 brida i gdje je početni čvor 809. Broj razmjenjenih poruka je 15, prema formuli $O(n) = 2*m_i = 8$. Na slici vidimo da je najdulji najkraći put 4, prema čemu dobijemo da je vremenska složenost mreže 8, $O(n) = 2*d(G) = 8$.

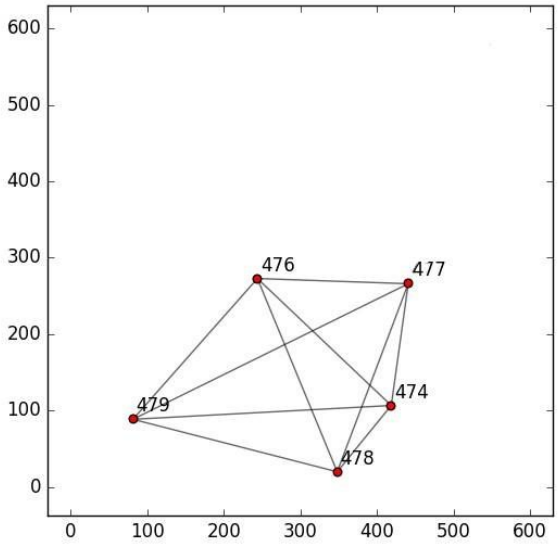
- INITIATOR - čvor 809
- n = 5
- m = 4
- Broj razmjenjenih poruka: 8
- Promjer: 4



Slika 2: Mreža s 5 čvorova (lanac)

Slika 3 prikazuje mrežu sa 5 čvorova i 10 bridova gdje su svi čvorovi povezani (INITIATOR je povezan sa svim čvorovima u mreži). Broj razmjenjenih poruka je 20, prema formuli O(n) = 2*m = 20. Na slici vidimo da je najdulji najkraći put 1, prema čemu dobijemo da je vremenska složenost mreže 2, O(n) = 2*d(G) = 2.

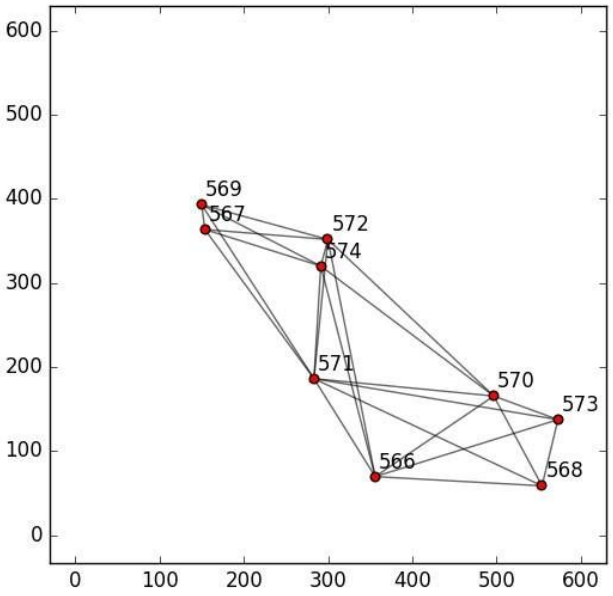
- INITIATOR - čvor 474
- n = 5
- m = 10
- Broj razmjenjenih poruka: 20
- Promjer: 1



Slika 3: Mreža s 5 čvorova

Slika 4 prikazuje mrežu s 9 čvorova i 24 brida. Broj razmjenjenih poruka je 48, prema formuli O(n) = 2*m = 48. Na slici vidimo da je najdulji najkraći put 3, prema čemu dobijemo da je vremenska složenost mreže 6, O(n) = 2*d(G) = 6.

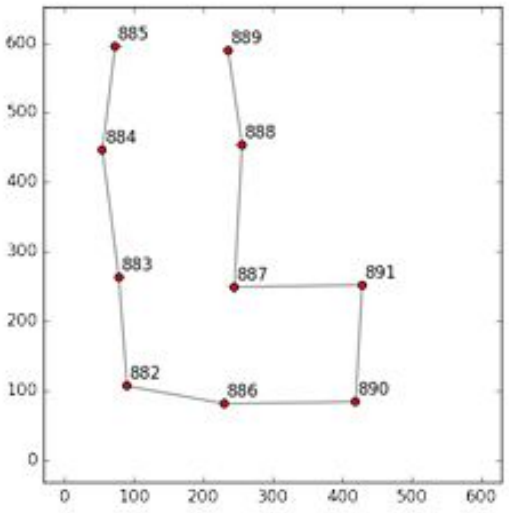
- INITIATOR - čvor 566
- n = 9
- m = 24
- Broj razmjenjenih poruka: 48
- Promjer: 3



Slika 4: Mreža s 9 čvorova

Slika 5 prikazuje mrežu s 10 čvorova i 9 bridova povezanu u lanac. Broj razmjenjenih poruka prilikom izvođenja algoritma je 34, prema formuli O(n) = 2*m = 18. Na slici vidimo da je najdulji najkraći put 9, prema čemu dobijemo da je vremenska složenost mreže 18, O(n) = 2*d(G) = 2*9 = 18.

- INITIATOR - čvor 885
- n = 10
- m = 9
- Broj razmjenjenih poruka: 18
- Promjer: 9



Slika 5: Mreža s 10 čvorova (lanac)