Sveučilište u Rijeci, Tehnički Fakultet

**Bežične mreže osjetila –**

**MaxTemp projekt**

Kastav, Matija Stojković

7.4.2016. 0069069039

**Zadatak**

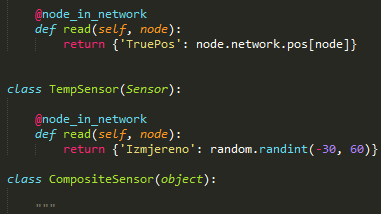
Napraviti klasu za temperaturni senzor. Implementirati algoritam za očitanje maksimalne temperature u mreži. Algoritam mora biti raspodijeljen, a čvorovi mogu koristiti samo informacije dobivene očitanjem senzora i komunikacijom. Eksperimentalno i analitički analizirati memorijsku i vremensku složenost algoritma.

**Ideja**

Temperaturni senzor implementiran je kao podklasa klase senzor i ima nadopunjenu metodu „node\_in\_network“ koja vraća nasumični broj između -30 i 60. Algoritam je zamišljen tako da svaki čvor očita temperaturu i zapiše ju u memoriju te pošalje ju svojim susjedima. Kada primi poruku, usporedi ju sa maksimalnom temperaturom koja mu je zapisana u memoriji i ako je veća, pošalje ju svojim susjedima i zapiše u memoriju.

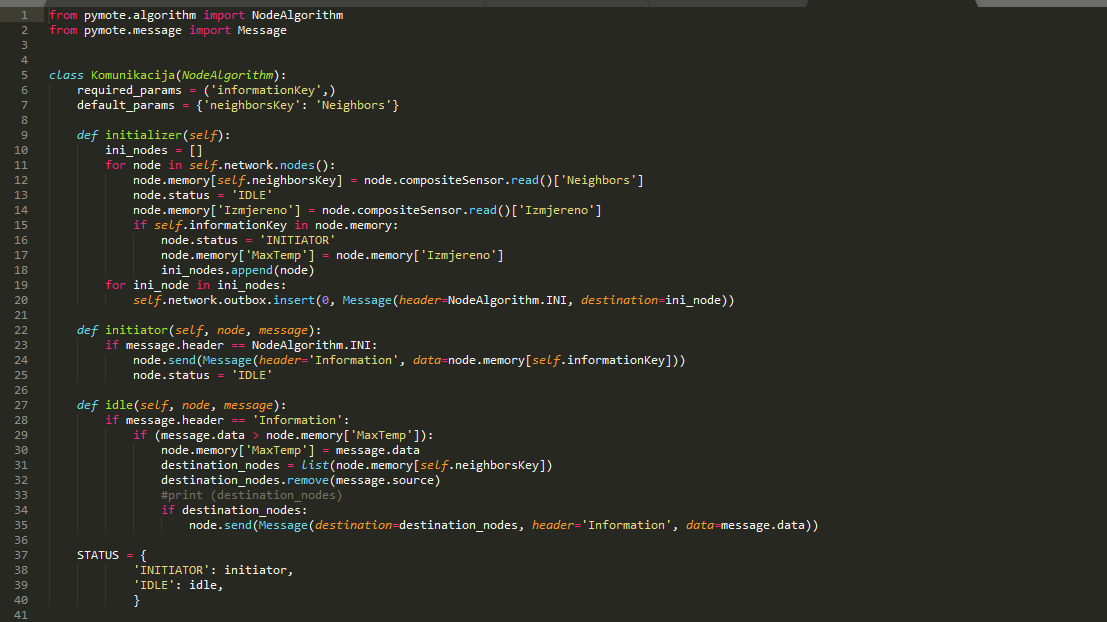
**Implementacija**

Temperaturni senzor zapisan je u datoteci „sensor.py“ kao podklasa klase „Senzor“:



*Slika 1:“Implementacija temperaturnog senzora“*

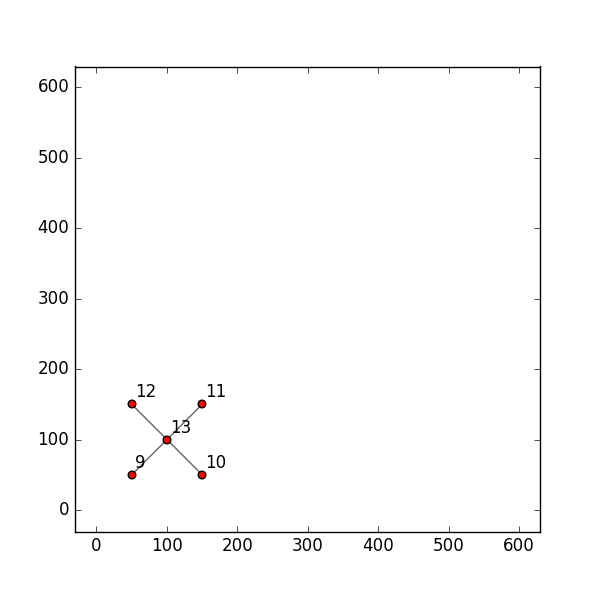
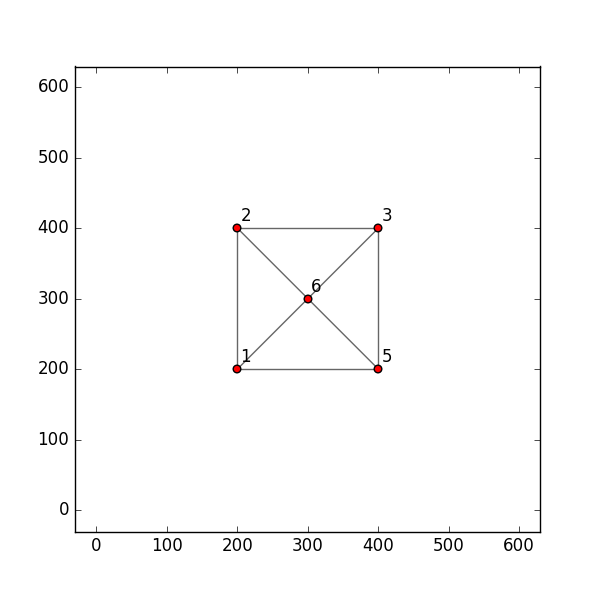
Algoritam je zapisan u klasi„Komunikacija“ nalazi se u datoteci „proba.py“ te nasljeđuje klasu „NodeAlgorithm“:



*Slika 2: „Implementacija algoritma“*

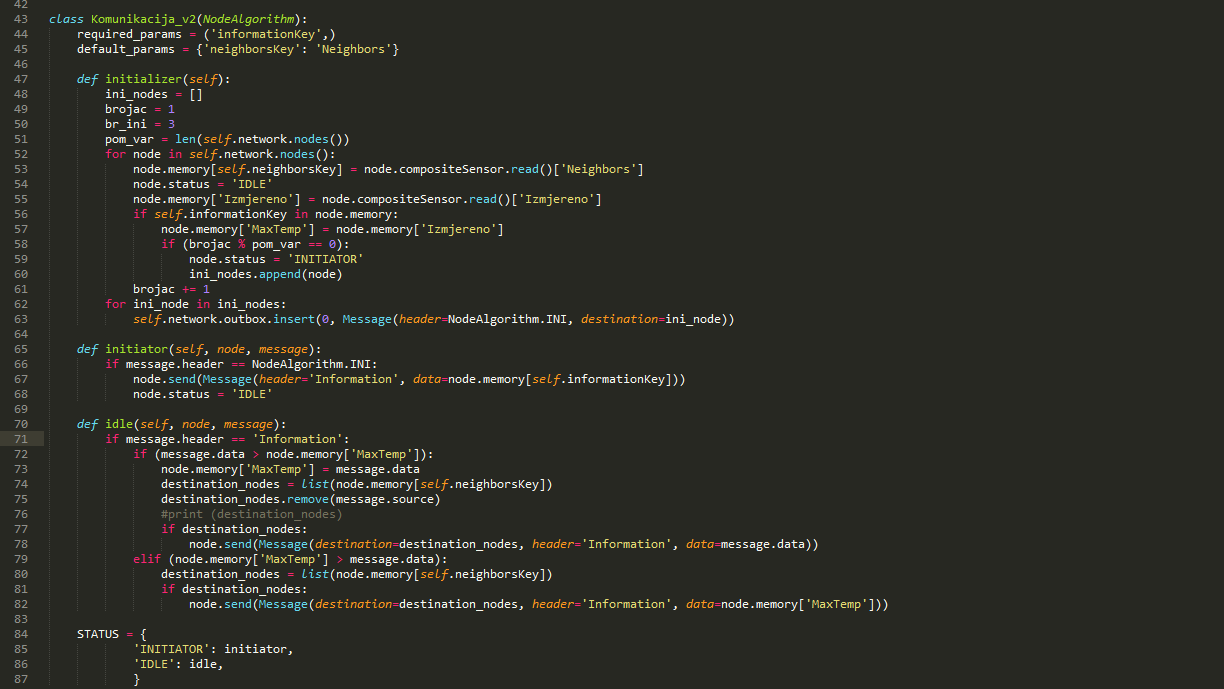
**Analiza**

Broj poruka u algoritmu ovisi o mreži, a sigurno se nalazi u sljedećem rasponu , , gdje je m broj čvorova. Broj koraka također ovisi o mreži i nalazi se u rasponu od . Najbolji slučaj je kada je čvor s najvećom izmjerenom temperaturom povezan sa svim ostalima, a ostali samo s njim. Najgori slučaj je kada postoji čvor koji je povezan sa svim ostalima, ima najmanju izmjerenu temperaturu, svi ostali čvorovi imaju međusobno različitu temperaturu te šalju centralnom čvoru svoje temperature od najmanje.

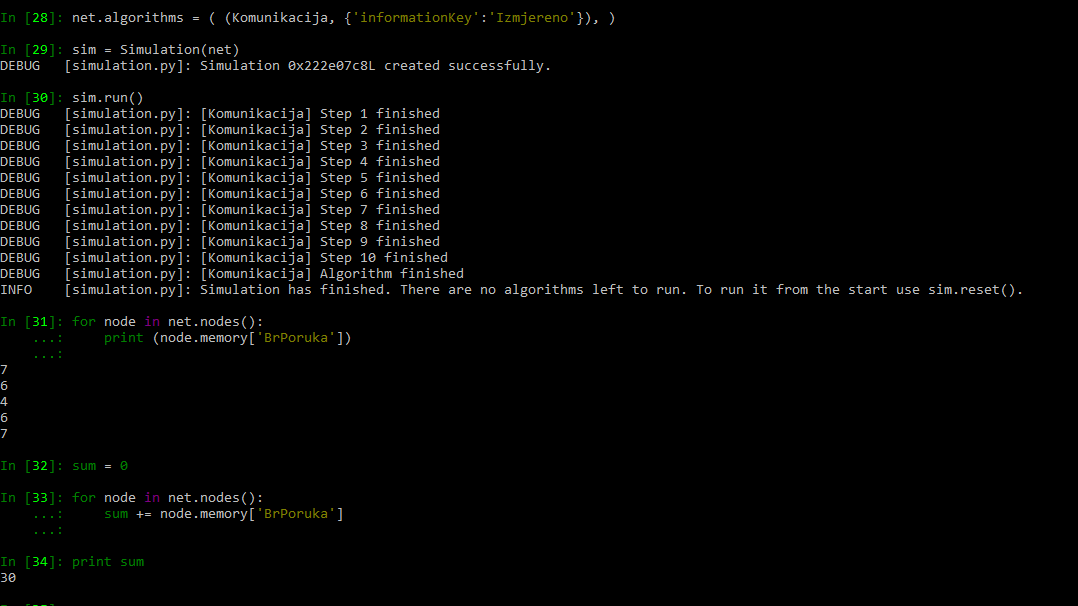
*Slika 3: „Najbolji slučaj – lijevo i najgori slučaj – desno“*

Ovaj algoritam postavlja svaki čvor za inicijalizatora, što nije česti slučaj u praksi. Obično postoji samo par inicijalizatora. Usporedbom takve implementacije algoritma „Komunikacija“ s 3 inicijalizatora (na slici); s našom dobit ćemo da je za mrežu od 100 čvorova potrebno oko 30 puta više koraka kako bi se taj algoritam izvršio. Rezultat u takvom algoritmu znatno ovisi o rasporedu čvorova unutar mreže te odabiru inicijalizatora. Razlika u broju koraka eksponencijalno raste s veličinom mreže.

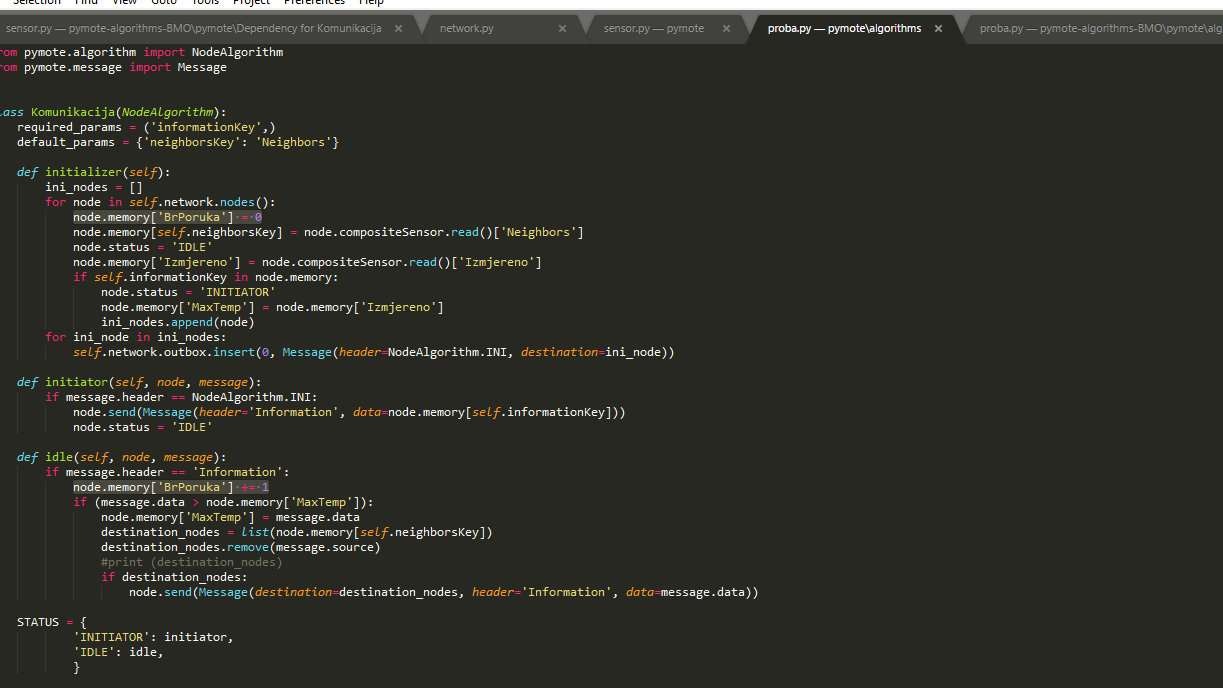


*Slika 4: „Komunikacija\_v2“*

Eksperimentalnom analizom potvrđujemo teoretske pretopstavke te time završavamo analizu. Broj poruka može se provjeriti implementacijom jednostavnog brojača u memoriji čvora, koji se poveća za 1 svaki puta kad primi poruku (Slike 5 i 6).

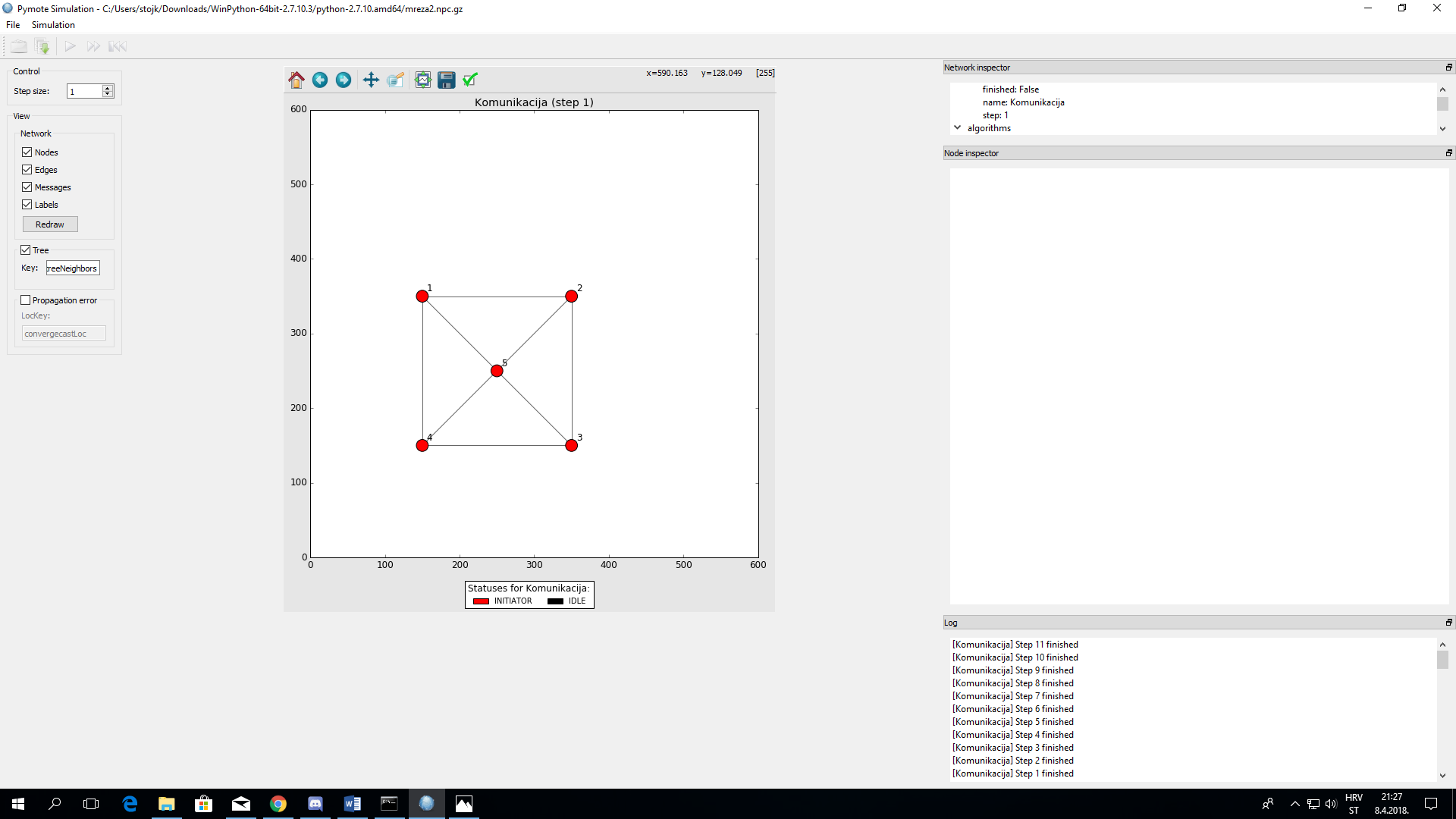


*Slika 5: „Prikaz simulacije mreže s implementiranim brojačem te ispis istoga“*



*Slika 6: „Implementacija brojača – vidi istaknuti tekst“*

Na slici ispod (Slika 7), u donjem desnom kutu vidi se da je za priloženu mrežu bilo potrebno 12 koraka kako bi se algoritam izvršio, što je više nego je teoretski predviđeno, ali teoretska analiza uzima u obzir da se sve poruke obrade u 1 koraku, što nije stvarni slučaj.



*Slika 7:“Simulacija“*