**Dokumentácia k  zadaniu z  predmetu Počítačové siete 2**

**Téma - Simulácia bezdrôtovej siete v  poli**

**Zadanie**

- Nastavte vhodnú fyzickú interpretáciu prostredia pomocou modelov definovaných v dokumentácii NS3.

- Hospodár sa pohybuje v okolí, chvíľu je a potom nie je v dosahu siete. Zvoľte vhodný model pohybu/pohybov.

- Vyberte a nastavte vhodný MAC protokol (WifiManager), upravte nastavenia.

- Vyberte a nastavte vhodný/é smerovací/ie protokol/y, upravte nastavenia.

- Vyberte a nastavte vhodný/é transportný/é protokol/y, upravte nastavenia.

- Aplikačná časť:

* senzorová sieť monitoruje prostredie. Poľnohospodár pritom občas príde a chce stiahnuť informácie zo senzorovej siete. Monitorujte stav batérie.
* simulujte pre každý senzor batériu s energiou (môže byť aj s dobíjaním).
* príklad na udalosť: Vstup poľnohospodár do dosahu senzorovej siete, Pri nízkom stavu batérie sa zmení smerovacia tabuľka, a pod.

Na vizualizáciu simulácie využite program a modul NetAnim. Zadefinujte aspoň 2 merané parametre QoS, ktoré vynesiete do grafov za pomoci modulu a programu GnuPlot. Simuláciu spustite viac krát, meňte pomocou SeedManager nastavenia generovania náhodnej premennej, aby ste získali štandardnú odchýlku merania (Errorbars) pre vynesené body grafu. Vhodne stanovte čas-trvanie simulácie.

Príklad QoS (pomer priemerného počtu poslaných užitočných údajov k celkovému počtu poslaných údajov na jednu cestu/balík), príklad parametra počet vedľajších komunikácii v sieti (údržba siete, pripojenie/odpojenie, hľadanie cesty, …).

Povinne argumenty fungujú (Simultime, anim), anim sa defaultne vygeneruje a čas simulácie je rovný 10.

**Implementácia**

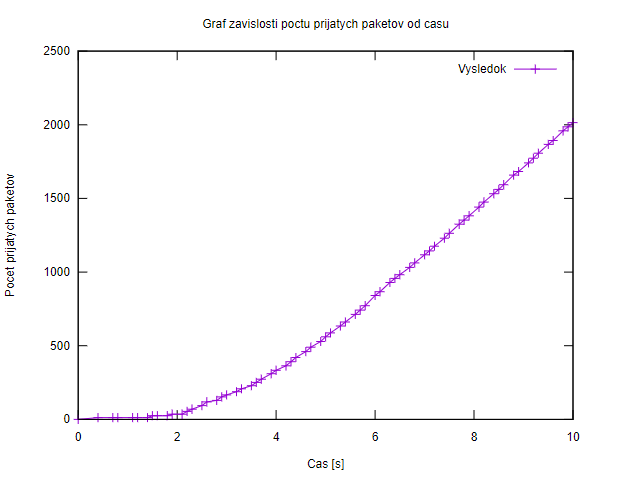
Náš model simuluje farmu v Nemecku na poli o veľkosti 5x5, kde je umiestnených 25 senzorov na monitorovanie vlhkosti pôdy. Senzorová sieť sa dobýja solárnou energiou. Poľnohospodár sa v priebehu dňa pohybuje a zbiera údaje zo senzorov, aby mohol optimalizovať spúšťanie zavlažovacieho systému.

**Popis OSI**

* Poľnohospodár komunikuje so senzorovou sieťou pomocou wifi štandardu 802.11g.
* **L1:** Fyzickým médiom pri prenose údajov wifi je vzduch. Uzly senzorovej siete sú statické. Poľnohospodár náhodne mení svoju polohu závislosti od času.
* **L2:** Senzorová sieť je nastavená v Ad-Hoc móde, nakonfigurované na konštantné vysielanie, lebo meranie vlhkosti pôdy prebieha v pravidelných intervaloch.
* **L3:** Použili sme staticRouting nakoľko sa naša senzorová sieť nepohybuje. Kedže máme relatívne malý počet senzorov použili sme IPv4 adresovanie, Pri použití IPv6 a našom počte senzorov by nebol efektívne využitý daný adresný priestor.
* **L4:** Na komunikáciu sme využili UDP protokol, pretože senzorová sieť odosiela dáta a nevyžaduje potvrdenie o prijatí paketu. Senzorová sieť funguje ako vysielač.

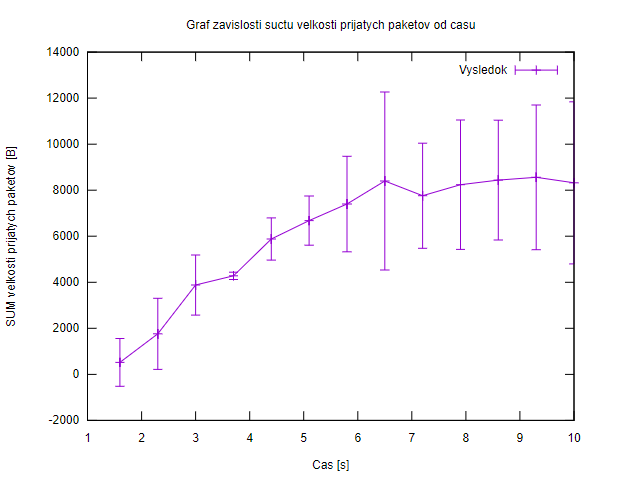
**Zhodnotenie grafov**

**Graf 1.**



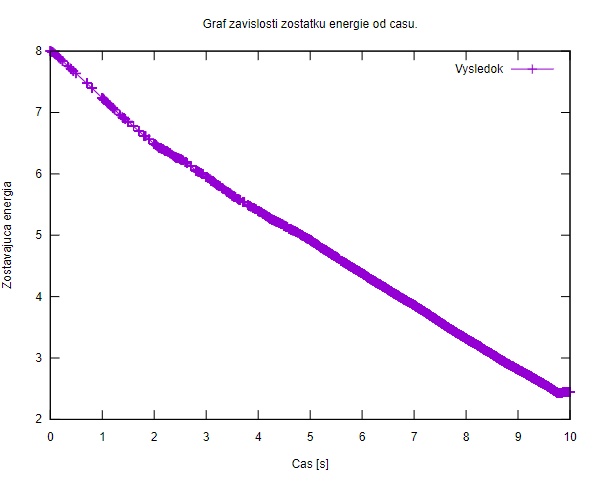
Na grafe môžeme vidieť, že s pribúdajúcim časom sa zväčšoval aj počet prijatých paketov, ktoré boli posielané bez náhodného prerušenia.

**Graf 2.**



Na grafe môžeme vidieť, že po určitej veľkosti prijatých paketov nastane pokles, časom sa však počet prijatých paketov a niečo zvýši.

**Graf 3.**



Na grafe môžeme vidieť, že batérie sa začali nabíjať až na konci simulácie. Čiže sme nastavili skoro dostatočnú počiatočnú energiu pre stav simulácie.

**Body zadania**

**Vizualizácia**: NetAnim, Simulácia vygeneruje bez nejakého argumentu netanim\_pole.xml.

**Reprezentácia merania**: Graf č. 1 Graf závislosti znalosti od cviciaceho, Graf č.2

**Volanie casovych udalostí**:   
Simulator::Schedule(pktInterval, &GenerateTraffic, socket, pktSize, n,pktCount - 1, pktInterval);

* Generovanie paketov v pravidelných intervaloch

Simulator::Schedule(Seconds(startTime + j \* 0.7 + k \* 0.4), &goToNewPosition, hosp.Get(0), k);

* Pohyb poľnohospodára

**Volanie udalosti zmenu stavu:**

Simulator::Schedule(Seconds(startTime + j \* 0.7 + k \* 0.4), &goToNewPosition, hosp.Get(0), k);

* Funkcia goToNewPosition podľa náhodne vygenerovaného čísla sa určí nová pozícia poľnohospodára.

**Použitie vlastnej triedy:**

* Pre jednoduchšie generovanie grafov sme si vytvorili vlastnú triedu.

**Časová zmena v modely na vrstve L1:**

Simulator::Schedule(Seconds(startTime + j \* 0.7 + k \* 0.4), &goToNewPosition, hosp.Get(0), k);

* Pohyb poľnohospodára

**Časová zmena v modely na vrstve L2-L5:**

Simulator::Schedule(pktInterval, &GenerateTraffic, socket, pktSize, n,pktCount - 1, pktInterval);

* Generovanie paketov v pravidelných intervaloch