ALOHA

Mateusz Stępniak

8 grudnia 2018

1 Wprowadzenie

W pliku algorithm.py znajdują się się odpowiednio klasy Aloha2n slotted ,CSMA będące implementacją odpowiednio algorytmów Aloha , Slotted Aloha , CSMA Symulacja została zmieniona w następujący sposób: dodano zmienną success_count, która jako licznik ramek, które udało się odebrać, oraz dodany została możliwość sprawdzenia czy ktoś inny nadawał w poprzedniej chwili.

2 Opis algorytmów

2.1 Aloha

Algorytm wykonuje transmisje, jeśli się nie powiodła zwiększa zmienną wait i zeruje wait_counter. Wait zostaje zwiększona dwukrotnie i dodana zostaje losowa wartość, z zakresu (1,wait_const). Wait_const jest ustawiona na początku pliku algorithm, przy każdym przypisaniu zwiększaniu wait zostaje zwiększona o 1. Mechanizm ten rozwiązuje problem powstający gdy użytkownicy zaczęli nadawać w tym samym czasie. Zwiększanie wait_const jest użyteczne gdy użytkowników jest dużo wtedy przy jednoczesnym zaczynaniu jest duża szansa że dodawane wartości będą jednakowe

2.2 Slotted Aloha

Algorytm działa analogicznie, z tym że zaczyna nadawać w określonych momentach gdy tick_count%maxlenght=0

2.3 CSMA

Algorytm sprawdza czy w poprzedniej chwili ktoś nadawał, jeśli nie nadawał to nadaje , jeśli nadawał to czeka losowy czas z zakresu (1,wait_const), gdzie wait_const rośnie wykładniczo przy nieudanych próbach (aby uniknąć sytuacji w której w poprzedniej chwili nikt nie nadawał więc teraz 2 klientów nadaje, wyłączają się przez zagłuszenia, znowu nikt nie nadaje i tak w kółko)

3 Testy

W folderze znajduje się skrypt tester.sh. Jako pierwszy pierwszy argument przyjmuje liczbę testów do wykonania jako drugi stringa parametry do wywołania symulacji. Program zwraca średnia liczbę sukcesów Testy dla wszystkich 4 programów są automatyzowane skryptem run.sh Przeprowadzono testy dla każdego algorytmu:

250 testów parametry defaultowe
simple 4.356
aloha2n 17.564
slotted 17.592
CSMA 17.436

100 testów –frame 1
simple 3.74
aloha2n 17.2
slotted 17.89
CSMA 18.22

100 testów –clients 20”
simple 5.05
aloha2n 17.55
slotted 17.46
CSMA 16.97

100 testów –length 10”
simple 4.62
aloha2n 17.23
slotted 18.61
CSMA 17.24

–clients 10 –frame 1 –length 10”
simple 3.58
aloha2n 16.54
slotted 16.51
CSMA 16.99

–clients 10 –frame 1 –length 10 –ticks 1000”
simple 4.98
aloha2n 18.05
slotted 17.18
CSMA 16.80

–clients 10 –frame 1 –length 10 –ticks 1000 –time 300” simple 4.13 aloha2n 15.66 slotted 17.37 CSMA 16.5

Widać że na różnych testach zaawansowane algorytmy wypadają dość po-

dobnie przy użytkownikach którzy dość często chcą coś wysyłać (frame 1) podejście CSMA wydaje się być odrobinę lepsze jednak przy widać testy na których wypada gorzej. Slotted Aloha jak się spodziewaliśmy jest lepszy niż zwykły Aloha jednak nie jest to bardzo duża różnica.