Wojciech Buczek

Wstęp

W ramach projektu należało napisać następujące algorytmy dostępu wielokrotnego do kanału:

- Aloha
- Slotted Aloha
- CSMA

1 Aloha

W każdym ticku wysyłamy dane zawsze, kiedy się pojawią. Jeżeli ta czynność się nie powiedzie, czekamy losową ilość czasu. Górne ograniczenie tej losowości zależy od ilości porażek, wykładniczo.

2 Slotted Aloha

Idea taka sama jak w algorytmie *Aloha*, jednak tym razem zaczynamy nadawać dopiero wtedy, kiedy rozpocznie się odpowiednia szczelina (slot).

3 CSMA

Zanim spróbujemy wysłać dane, patrzymy, czy ktoś aktualnie nadaje na kanale. Jeżeli tak, to czekamy na zwolnienie tego kanału (taką ilość czasu, aby wyrównać do ramki). Jeżeli nie, to wysyłamy dane.

Testy

Do skryptu *simulator.py* została dodana możliwość dodania parametru *-s* (silent), która powoduje wypisanie tylko wyników (bez słów). Zwracany jest więc wtedy output: *number of frames*, *number of success*, *effectiveness*.
Do przeprowadzenia testów zostały napisany pythonowy skrypt *test.py*. Dla zdefiniowanych w tym skrypcie zbiorów, przeprowadzane są testy dla wszystkich możliwych kombinacji. Output dla tego skryptu daje gotową latech'ową tabelkę. Każdy z tych testów został uruchomiony 10 razy.

Wyniki

Clients = 5, Length = 1, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	346	336	97.11%
SlottedAloha	390	373	95.64%
CSMA	476	473	99.37%

Clients = 5, Length = 2, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	213	180	84.51%
SlottedAloha	213	181	84.98%
CSMA	411	402	97.81%

Clients = 5, Length = 5, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	114	69	60.53%
SlottedAloha	112	67	59.82%
CSMA	215	190	88.37%

Clients = 10, Length = 1, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	414	344	83.09%
SlottedAloha	420	357	85.00%
CSMA	794	782	98.49%

Clients = 10, Length = 2, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	235	148	62.98%
SlottedAloha	231	152	65.80%
CSMA	470	407	86.60%

Clients = 10, Length = 5, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	132	41	31.06%
SlottedAloha	133	44	33.08%
CSMA	216	128	59.26%

Clients = 15, Length = 1, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	415	299	72.05%
SlottedAloha	424	308	72.64%
CSMA	768	705	91.80%

Clients = 15, Length = 2, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	268	134	50.00%
SlottedAloha	266	137	51.50%
CSMA	431	318	73.78%

Clients = 15, Length = 5, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	178	35	19.66%
SlottedAloha	167	23	13.77%
CSMA	210	71	33.81%

Clients = 20, Length = 1, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	453	290	64.02%
SlottedAloha	435	266	61.15%
CSMA	849	723	85.16%

Clients = 20, Length = 2, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	308	119	38.64%
SlottedAloha	280	88	31.43%
CSMA	406	244	60.10%

Clients = 20, Length = 5, Frames = 0.1

Algorithm	total	success	effectivness
Aloha	221	25	11.31%
SlottedAloha	216	25	11.57%
CSMA	245	51	20.82%

Wnioski z testów

We wszystkich testach najlepiej prezentował się algorytm CSMA. Algorytmy Aloha i SlottedAloha miały bardzo podobną skuteczność (dla większej liczby klientów skuteczność ta była znacznie mniejsza niż w algorytmie CSMA).