Router

Mateusz Stępniak

1 Wstęp

W pliku algorithm.py znajdują się implementacje algorytmów LinkStateRouter oraz BFS(oprócz bazowych distancevector i random). W pliku network.py znajduja się scenariusze sieci: hipercube ,randomgeographic ,awaria. Każdy z scenariuszy posiada parametr n charakteryzujący wielkość grafu. Kilka modyfikacji symulacji: uuid w momencie tworzenia jest castowany do stringa, trzymanie obietku uuid powodowało błedy przy generowaniu json , symulator utrzymuje seta pakietow które były już raz zrotowane , i nie liczy ich drugi raz (przy BFS pakiety dochodziły wielokrotnie). Dodany zostal opcjonalny argument sed ustatwiający seed'a dla random w network.py aby uzyskać dobre porównanie testów.

2 Opis algorytmow

2.1 BFS

Algorytm w każdej turze wysyła do wszystkich aktualnie dostępnych routerów pierwszy pakiet na liście. Odebrane pakiety wysyłane są w kolejności FIFO. Algorytm utrzymuje zbiór już wysłanych dzięki czemu nie odbiera pakietów które wysyłał do innych przed chwilą. Jest to realizowane przez set o nazwie sended.

2.2 LinkStateRouter

Algorytm utrzymuje graf całej sieci i aktualizuje go na bieżąco na podstawie informacji odbieranych od innych routerow wysyłanych co 10 tur. Wysyłając metapakiet router wysyła cała swoją informacje o aktualnym stanie grafu wraz z numerem wersji(wersja oznacza ostatnią turę w której w której graf został zaktualizowany). Następnie odpala dijkstrę na grafie i oblicza dla każdego routera kierunek w którym mają być posyłane pakiety (dir_map).

3 Scenariusze sieci

Wszystkie scenariusze co turę generują jeden losowy pakiet.

3.1 Hiperkostka

Parametr n jest wymiarem hiperkoski, wierzchołków w grafie jest odpowiednio 2^n. Wierzchołki są zlinkowane gdy ich odległość edycyjna w zapisie binarnym jest równa 1;

3.2 Awaria

Parametr n charakteryzuje długość cyklu jedna krawedz co 5 tur na przemian pojawia się i znika.

3.3 Randomgeographic

Na kwadracie o rozmiarze *size* rozmieszczonych jest n ruterow każdy z nich porusza się w losowym kierunku, z prędkością 1, wybranym przy inicjalizacji. Poruszający się ruter opisuje klasa vertex.

4 Testy

Wyniki testów znajdują się w folderze testy. Dzięki ustaleniu seeda mamy powtarzalność symulacji. Skrypt run.sh przyjmuje jeden argument, liczbę tur (parametr ticks w symulacji),następnie odpala na każdym scenariuszu sieci każdy z możliwych algorytmów.

Widać, że BFS poprzez zwielokrotnianie pakietów zapycha sieć i pakiety docierają, ale średnio czas dotarcia jest dłuższy niż dla LinkState i DistanceVector. LinkState natomiast ma odrobinę lepszy czas niż distancevector. Biorąc pod uwagę dostarczalność algorytmy BFS DistanceVector LinkState wypadają bardzo podobnie, nie widać dużych różnic w wynikach. Najczęściej BFS ma odrobinę niższy współczynnik. Prawodopodobnie wynika to stąd że symulacja kończy się, a zapchana sięć ma w sobie jeszcze trochę pakietów BFS'a i nie wszyskie dotarły.