Data Communication Final Project Report

2015-11985 남중혁

1. Introduction

수업시간에 Ethernet은 다양한 MAC schemes을 이용하여 데이터 통신을 한다고 배웠다. 그 중 1-persistent CSMA + binary exponential backoff 방식의 scheme과 IEEE 802.11 CSMA-CA DCF 두 가지 방식을 Simulate 하는 코드를 Event-Driven Simulation 방식으로 작성하여, 두 가지 MAC scheme의 performance를 비교하고자 한다.

1-persistent CSMA + binary exponential backoff 방식의 scheme 은 충돌을 감지하는 기법으로 media 가 idle임을 확인하면 충돌이 나든 말든 전송을 시도하여, 충돌이 나면 backofftime을 기다린다.

IEEE 802.11 CSMA-CA DCF scheme 은 충돌을 회피하는 기법으로, media의 상태를 감지할 수 없기 떄문에, IFS와 backoff time 기법으로 최대한 충돌을 피해야하며, ack 신호로 전송이 완료되었는지 충돌이 났는지 확인해야만 한다.

+ Event-Driven Simulation은 simulation을 event가 일어난 시점을 기준으로 simulation을 작성하기 때문에 아무 event도 일어나지 않는 시간이 많을 경우 효과적인 방법이다.

1. 개발 환경
   1. 언어 : Java
      * + openjdk version "1.8.0\_131"
        + OpenJDK Runtime Environment (build 1.8.0\_131-8u131-b11-0ubuntu1.16.04.2-b11)
        + OpenJDK 64-Bit Server VM (build 25.131-b11, mixed mode)
   2. OS : Linux Ubuntu 16.04.2 LTS
   3. 컴파일 방법 : final\_project 폴더 안에서 $make
   4. 실행 : $java Simulator
2. System Description

<전체적인 설계>

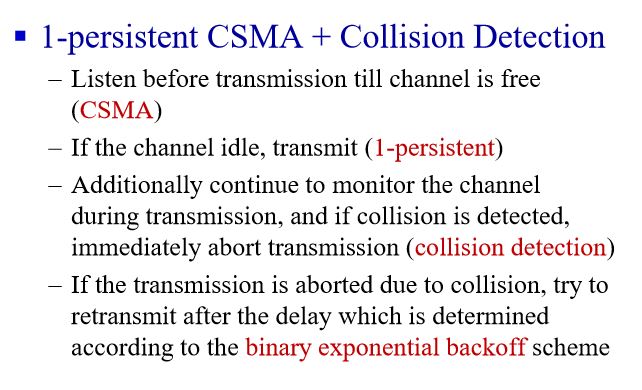
simulator에서 time은 현재 진행 중인 시간을 의미하고, time 1은 10us를 의미한다. Pack size 는 1024 bytes 이고, Transmission rate 는 10Mbps 인데 계산의 편의를 위하여, 한 패킷의 transmission time을 820us (즉 time 82로 하였습니다. (실제로는 1024 \* 8 / 10^7)us = 819.19999 입니다만.. 계산의 편의를 위하여 그렇게 하였습니다.

* 1. 사용자에게 N 과 lambda를 입력받는다.
  2. <Random Generator Class>에서 Poisson Distribution을 이용하여, 각 Node가 발생하는 패킷의 개수를 난수를 통해 정의한다. 그리고 random distribution을 통하여 각 Packet을 0~MaxTime 시간 안에 패킷을 랜덤하게 발생시킨다.

Poisson Distribution 정의에 의하면, “ 단위 시간 안에 어떤 사건이 몇 번 발생할 것인지를 표현하는 [이산 확률 분포](https://ko.wikipedia.org/wiki/%EC%9D%B4%EC%82%B0_%ED%99%95%EB%A5%A0_%EB%B6%84%ED%8F%AC)이다.” 인데, 단위 시간이 정해 지지 않았으므로, 정해주어야 한다. 그래서 maxTime을 설정할 수 있도록 하였다.

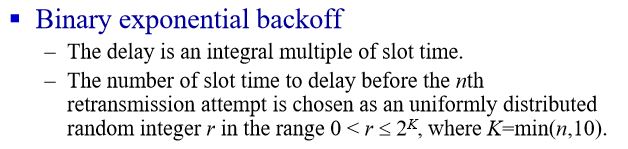
* 1. 이후 발생된 패킷을 Priority Queue에 차례로 넣었다. Priority Queue는 데이터를 넣으면 순서에 맞게 삽입되기 때문에 event가 일어난 time으로 정렬해야하는 지금 simulation에 적당한 자료구조이다.
  2. 각 시뮬레이션에서 Priority Queue에서 순차적으로 꺼내 Communication Simulation을 진행한다. 두 가지 방식의 자세한 설명은 4. 에 적겠습니다.

1. Simulation model
   1. 1-persistent CSMA + Collision Detection



교과서에 있는 1-persistent CSMA + Collision Detection 방식을 그대로 시뮬레이션에 적용하였다. 1-persistent는 채널을 sensing하여 idle이면, 바로 전송을 시도하고, busy라면 idle이 될 때 까지 sensing하여 idle이 되자마자 전송 시도한다.

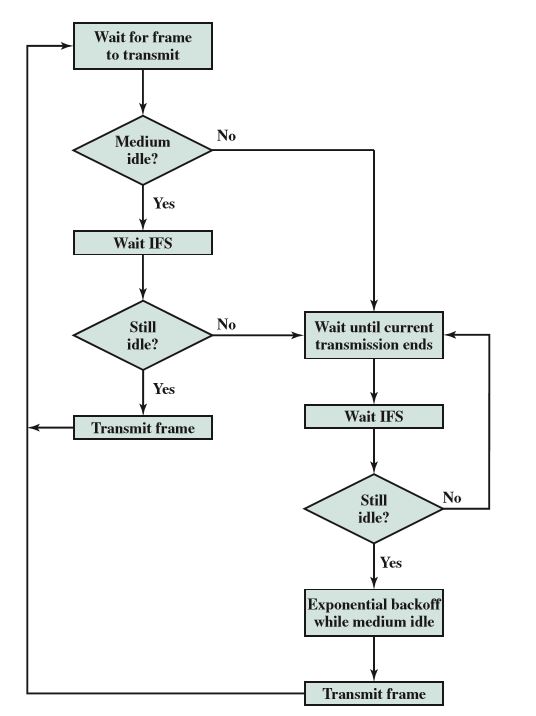
그러다가 collision이 일어나면 binary exponential backoff time 후에 위의 방식으로 다시 돌아가는데, binary exponential backoff time 은



충돌 횟수가 K일 때 0 < r <= 2^K 의 r을 구한 후 r에 Transmission Time 을 곱한 값을 기다리도록 하였다. Binary exponential backoff scheme은 다양한 load traffic 상황에서도 빠르게 적응할 수 있도록 설계 것으로, 충돌이 일어나면 지수 승으로 time이 늘어나기 때문에 high-load traffic에도 빠르게 회복할 수 있도록 설계한 것이다.

* 1. IEEE 802.11 CSMA-CA DCF

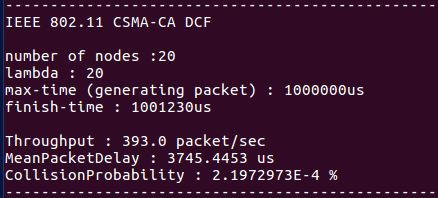
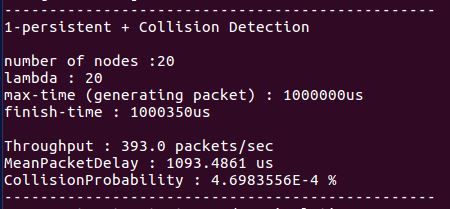
Collision Avoidance 기법으로, media가 idle이 되더라도 SIFS를 기다리고, 또 busy에서 기다린 상태였다면, exponential backoff time을 기다린 후에야 전송한다. 이렇게 조심스럽게 하는 이유는, 무선 통신의 경우 collsion을 감지할 수 없기 때문에, ACK Timer가 만료되기 전까지는 알 수 없다. (낭비) 따라서 최대한 충돌을 피하는 방법으로 채택한 방법이 PCF, DCF 기법이다. Exponential Backoff는 media가 idle일 때만 대기 시간이 감소하도록 설계 되어있는데, 이는 먼저 기다린 Packet이 먼저 전송시도하도록 하기 위함이다. 또한 SIFS < DIFS 인 이유는 SIFS는 ACK를 보내기 위해 기다리는 space인데, 당연히 새로운 Data를 전송하는 것 보다 이미 받은 Data에 대해 ACK를 보내는 것이 우선순위가 높기 때문에 이를 처리해주기 위함이다.



1. Numerical results with discussion

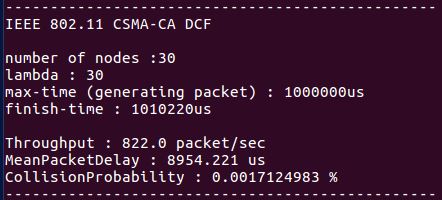
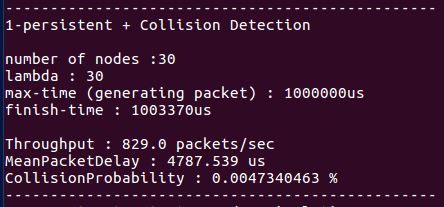
maxTime : Poisson distribution으로 정의된 패킷들이 (0,maxTime)시간 안에서 랜덤하게 발생되도록 정의하였다. 위쪽 그림은 CSMA/CD 이고, 아랫쪽그림이 CSMA/CA이다.

* 1. N=20, Lambda=20



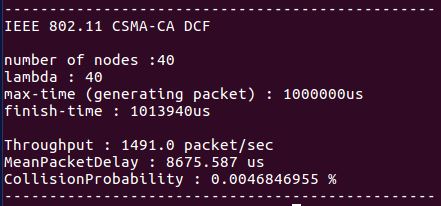
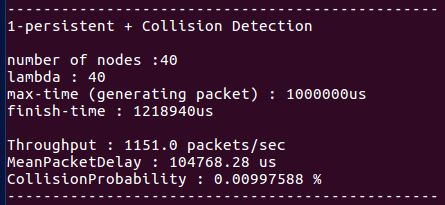
Traffic이 낮은 경우 초당 전송량은 같다. (1초안에는 모든 패킷을 다 보내므로,) 하지만 Packet 전송을 위해 기다리는 시간은 CD가 월등히 낮았다. (바로바로 보내기 때문).

* 1. N=30, Lambda=30



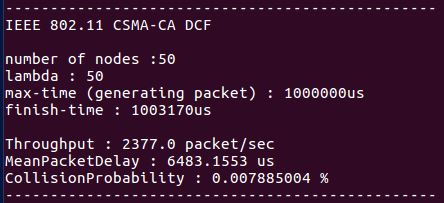
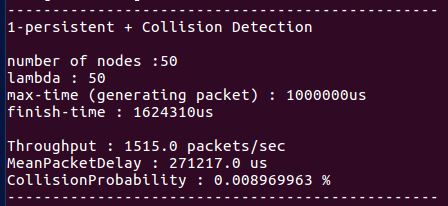
아직까지 CSMA/CD 가 PacketDelay에서 좋은 performance를 보인다.

* 1. N=40, Lambda=40



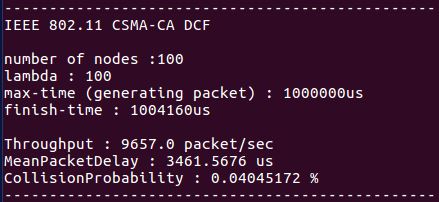
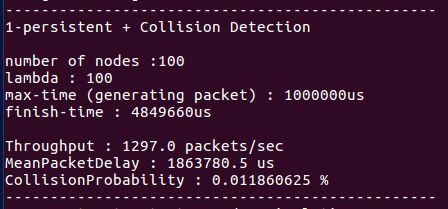
N과 Lambda가 40이 되자 CSMA/CA의 전송속도와 딜레이가 더 좋은 performance를 보이기 시작했다.

* 1. N=50, Lambda=50



Traffic이 heavy해질수록 그 차이는 점점 더 커져갔고,

* 1. N=100, Lambda=100



극단적으로 heavy-traffic을 만들어 보았다. 그 차이를 명확히 느낄 수 있었다

1. Conclusion

1-persistent CSMA/CD scheme은 Collision Detection 기법으로, media가 idle이 되는 즉시 packet을 전송하고, collision이 일어나면 그 때 처리하는 낙천적인 style의 기법이다. 이 방식은 low-traffic 즉, packet이 간헐적으로 일어나면, 상당히 효율적인 방법이다. 위의 시뮬레이션에서도 보았듯이, low-traffic에서는 대기시간을 확연히 줄여주지만, heavy해질수록 그 performance가 상당히 줄어든다. 따라서 low-traffic이 예상되는 환경에서 사용하면 좋을 것이다.

반면에, IEEE 802.11 CSMA-CA DCF 기법은 Collision Avoidance 기법으로, media가 idle이 되더라도 SIFS를 기다리고, 또 busy에서 기다린 상태였다면, exponential backoff time을 기다린 후에야 전송한다. 이처럼 Collision을 피하기 위해 많은 방법들을 집어넣었기 때문에 위의 시뮬레이션 test 결과와 같이, traffic이 heavy해져도 적응능력이 좋다. 하지만 low-traffic에서는 media가 idle인지도 모르고 쓸데없이 많이 쉬기 때문에 performance가 좋지 않다. 따라서 heavy-traffic에서 써야하는 방법이다.