RTT의 가중평균값을 이용한 혼잡 예측 기반 TCP 혼잡 제어 기법

임민기*, 김동회**

요약

TCP(Transmission Control Protocol)는 혼잡한 상황에서 혼잡 제어를 수행하므로 신뢰성 있는 전송을 할 수 있지만 혼잡 회피(Congestion avoidance) 과정에서 혼잡으로 인한 패킷 드롭이 일어날 때까지 혼잡 윈도우의 크기를 증가시키므로 패킷 손실이 증가할 수도 있다. 본 논문에서는 상기 문제점을 해결하기 위하여 RTT(Round Trip Time)를 이용한 새로운 혼잡 예측 TCP 혼잡 제어 기법을 제안한다. 제안하는 기법은 과도한 패킷 누적으로 인한 버퍼 오버플로우가 발생하는 시점에서 RTT 값들의 가중평균값인 SRTT(Smooth RTT)값을 측정한 후 패킷 전송 시에 같은 SRTT값을 가질 때를 혼잡한 상황이라 예측하여 혼잡 윈도우를 감소시키는 알고리즘이다. 시뮬레이션 결과를 통하여, 제안하는 기법이 무선 구간에 의한 패킷 손실보다는 버퍼 오버플로우에 의한 패킷 손실이 클 경우에 패킷 손실률과 처리량 측면에서 좋은 성능을 가지는 것을 확인할 수 있었다.

키워드 : 전송 제어 프로토콜, 가중평균값, 혼잡 제어, 버퍼 오버플로우, 왕복시간

The Congestion Estimation based TCP Congestion Control Scheme using the Weighted Average Value of the RTT

Min-Ki Lim*, Dong-Hoi Kim**

Abstract

TCP, which performs congestion control in congestion condition, is able to help a reliable transmission. However, packet loss can be increased because congestion window is increased by the time the packet is dropped in the process of congestion avoidance. In this paper, to solve the above problem, we propose a new congestion estimation based TCP congestion control scheme using the weighted average value of the RTT. After measuring a SRTT, which means the weighted average value of RTTs, at this point of time when a buffer overflow is occurred by an overloaded packet, the proposed scheme estimates the time, when the same SRTT is made in packet transmission, as a congestion time and then decreases the congestion window. The simulation results show that the proposed scheme has a good performance in terms of packet loss rate and throughput when the packet loss due to buffer overflow is larger than that due to wireless channel.

Keywords : Transmission Control Protocol, Smooth RTT, Congestion Control, Buffer overflow, Round Trip Time

1. 서론

※ Corresponding Author : Dong-Hoi Kim

Received : April 01, 2015

Revised : May 22, 2015

Accepted : May 30, 2015

* Dept. of Electrical and Electronic Engineering,
College of Information Technology, Kangwon
National University

Tel: +82-33-250-6349, Fax: +82-33-259-5674

TCP(Transmission Control Protocol)는 인터넷에서 가장 널리 사용되는 전송 프로토콜로 다

email: donghk@kangwon.ac.kr

** Dept. of Electrical and Electronic Engineering,
College of Information Technology, Kangwon
National University

양한 응용 프로그램 기반 프로토콜로 사용되고 있다. TCP는 혼잡 제어 기법을 통해 신뢰성 있는 데이터 전송이 가능하며 고속 네트워크에서 TCP의 성능을 향상시키기 위한 혼잡 제어 기법 연구들이 진행되어왔다[1][2][3][4][5][6]. 현재 인터넷에서 대표적으로 사용되는 전송 프로토콜인 TCP Reno는 저속 증가(Slow start)와 혼잡 회피(Congestion avoidance)를 통해 혼잡 제어를 수행한다. TCP Reno의 혼잡 제어 알고리즘인 AIMD(Additive Increase Multiplicative Decrease)는 병목 링크의 버퍼 오버플로우로 인한 패킷 손실이 발생할 때 까지 혼잡 윈도우를 가산적으로 증가시킨다. 그리고 네트워크 혼잡이 발생하면 혼잡 윈도우를 감소시킨다. 이는 혼잡한 상황이 자주 발생하는 환경에서 혼잡에 의한 지속적인 패킷 손실을 일으키기 때문에 활용도가 떨어지는 단점을 가지고 있다.

기존의 TCP Reno는 패킷 손실이 발생할 때까지 보수적으로 혼잡 윈도우를 증가시켰다면 본 논문에서는 혼잡이 발생하기 전에 혼잡한 상황을 미리 예측하여 혼잡 윈도우를 감소시킨다. 혼잡 회피를 사용하여 혼잡 윈도우의 크기를 1씩 증가시켜 패킷을 전송하는 내용은 기존의 TCP Reno와 같지만 처음 패킷 손실이 일어났을 때의 SRTT(Smooth RTT)값을 측정하여 이후 패킷 전송 시 같은 SRTT값을 가질 때에 혼잡한 상황이라 예측하여 혼잡 윈도우를 감소시킨다. 혼잡이 발생하면 패킷 손실이 일어나게 되는 경우와 다르게 혼잡을 예측하기 때문에 패킷 손실률을 줄일 수 있고 패킷 손실률이 감소하면 재전송 패킷의 수도 줄어들게 되어 그 만큼 보낼 수 있게 되는 데이터 량이 증가하게 되므로 처리량이 증가하는 결과도 낼 수 있다. 하지만 버퍼 오버플로우로 인한 손실보다 무선 구간에서의 패킷 손실의 비율이 높을 경우 제안하는 기법의 성능이 떨어지게 된다. 그래서 제안하는 기법은 버퍼 오버플로우가 빈번하게 일어나는 환경에서 좋은 성능을 나타내게 된다.

본 논문의 2장에서는 현재 인터넷에서 대표적으로 사용 중인 TCP Reno에 대하여 간략히 서술하고, 3장에서는 제안하는 기법인 혼잡 상황 예측을 위한 시스템 모델에 대하여 상세히 기술하며, 4장에서는 NS-2(Network Simulator-2)[7][8]를 이용한 실험으로 제안하는

기법의 성능을 분석하였고, 5장에서는 결론을 제시하였다.

2. 기존의 TCP Reno 기법

TCP Reno는 (그림 1)과 같이 저속 증가와 혼잡 회피의 두 가지 메커니즘을 가지고 혼잡 제어를 수행하게 된다[9].

(그림 1) 기존 TCP Reno의 혼잡 제어

(Figure 1) The congestion control of the conventional TCP Reno

저속 증가 구간에서 송신측은 패킷을 전송하고 ACK(Acknowledge)를 수신하게 되면 혼잡 윈도우를 지수적으로 증가 시키면서 가용 대역폭까지 빠르게 도달하게 된다. 계속되는 지수적인 증가로 인한 혼잡을 예방하기 위해서 혼잡 윈도우 임계값인 ssthresh(Slow Start Threshold)를 넘어서게 되면 혼잡 회피 구간으로 바뀌어 지수적이 아닌 가산적으로 혼잡 윈도우를 증가시킨다. 그리고 네트워크에 혼잡이 발생하여 패킷이 손실되면 혼잡 윈도우를 감소시킨다. 세 번의 중복된 ACK(3dupACKs)를 수신하게 되면 패킷이 손실되었다고 판단하여 ssthresh와 혼잡 윈도우의 크기를 반으로 줄이고 패킷을 재전송 한다. Time-out이 발생한 경우는 혼잡 윈도우의 크기를 1로 감소시킨 뒤 저속 증가부터 다시 혼잡 윈도우를 증가시킨다. TCP Reno는 혼잡 윈도우를 제어하면서 혼잡한 상황에서 빨리 회피할 수 있다는 장점을 가지고 있지만 현재의 고속 네트워크에서 가용 대역폭을 점유하는데 까지 시간이 오래 걸리며 혼잡한 상황을 예측하기 불가능할 뿐만 아니라 혼잡이 발생하였을 때 혼잡 윈도우를 급격히 감소시켜 효

울적이지 못하다.

3. 제안하는 혼잡 예측 기법

본 논문에서 제안하는 예측 시스템 모델은 혼잡 회피 구간에서의 혼잡 윈도우 조절 알고리즘을 개선한 기법으로 혼잡 윈도우가 가산적으로 증가하여 버퍼에 데이터가 쌓이게 되면 RTT값이 증가한다는 것[10][11]을 바탕으로 고안하였다.

(그림 2) 병목링크 AP의 수신 버퍼에 쌓이는 패킷과 RTT와의 관계

(Figure 2) The relation between packet accumulated to receiving buffer of AP and RTT

(그림 2)와 같이 병목링크 AP의 수신 버퍼에 패킷이 쌓이게 되면 패킷은 버퍼에 머물러있는 시간이 증가하게 되므로 버퍼에 오래 머물수록 RTT값이 점차 증가하게 된다. 여기서 패킷이 최대로 쌓여서 버퍼 오버플로우가 발생하는 시점에서 RTT값의 가중평균값인 SRTT를 측정하게 되면 편차가 크지 않은 비슷한 값을 가지게 된다. 아래의 식 (1)과 식 (2), 식 (3)은 순조로운 RTT인 SRTT가 구해지는 식이다[12].

$$\text{Initially} \rightarrow \text{No value} \quad (1)$$

$$\text{After first measurement} \rightarrow RTT_s = RTT \quad (2)$$

$$\text{After reach measurement} \rightarrow RTT_s = (1 - \alpha)RTT_s + (\alpha \times RTT) \quad (3)$$

이를 바탕으로 하여 네트워크상에서 혼잡이 발생하여 패킷 손실이 일어났을 때의 SRTT값을 지정한 송신 노드(TCP Server)의 혼잡 변수(max_rtt)에 저장하여 버퍼에 패킷이 가득 찼을

때의 정보를 알아낸다. 이후 혼잡 윈도우가 다시 지수 증가 또는 가산 증가를 하게 되어 저장된 혼잡 변수보다 크거나 같을 때 혼잡이 일어나지 않았어도 혼잡 상태라고 예측하고 혼잡 윈도우의 크기를 줄인다. 그리고 무선 환경에서의 거리와 장애물 등에 따라 처음 혼잡이 일어났을 때의 RTT보다 더 큰 RTT에서 혼잡이 발생할 수 있기 때문에 혼잡 변수를 고정시켜놓지 않고 정상적인 ACK수신을 받을 때 마다 값을 1씩 증가시켜 대역폭을 충분히 점유할 수 있도록 하였다. 여기서 한 가지 문제가 발생하는데 처음 일어난 손실이 버퍼가 가득 차지 않은 상태에서 무선 링크 에러율로 인한 손실인 경우에 충분히 커지지 않은 SRTT값을 혼잡 변수에 저장하게 되는데 이것은 이후 버퍼에 패킷이 충분히 쌓이기도 전에 혼잡이라고 예측하고 혼잡 윈도우의 크기를 감소시키며 혼잡 변수가 증가한다고 해도 가용대역폭을 충분히 점유하는데 까지 시간이 오래 걸릴 것이다. 이렇게 되면 처리율이 현저히 감소하는 결과를 가져올 수 있는데 이 문제의 해결 방안으로 식 (4)와 같이 가장 좋은 환경이라고 예측할 수 있는 처음 패킷을 보냈을 때의 SRTT를 기본 RTT(base_rtt) 변수로 저장하고 식 (5)와 같이 현재의 SRTT 값을 현재 RTT(t_srtt) 변수에 저장한다. 이 두 변수의 비율을 나타냈을 때 식 (6)과 같이 혼잡이 일어나지 않는 매우 좋은 환경일 경우 0.5의 값을 가지게 되는데 이를 이용하여 문제를 해결할 수 있다.

$$\text{Initially } RTT_s \rightarrow \text{base_rtt} \quad (4)$$

$$\text{Current } RTT_s \rightarrow t_srtt \quad (5)$$

$$\text{base_rtt} / t_srtt = 0.5 \quad (6)$$

혼잡이 일어났을 때 식 (6)의 값이 0.5보다 크거나 같을 때에는 버퍼 오버플로우에 의한 패킷 손실이 아닌 무선 구간에서 패킷 손실로 간주하고 SRTT값을 혼잡 변수에 저장하지 않는다. 그러면 낮은 혼잡 변수 값으로 인해 버퍼에 패킷이 충분히 채워지지 못하게 되어 충분한 대역폭을 점유할 수 없게 되는 문제를 해결할 수 있게 된다. 위의 내용을 (그림 3)과 (그림 4)와 같이 시뮬레이션 알고리즘으로 나타냈다. 앞서 서술한 내용과 같이 base_rtt는 가장 처음

저장되는 SRTT값인 기본 RTT를 의미하고 t_{srtt} 는 마지막으로 수신된 ACK의 SRTT값인 현재 RTT, max_rtt 는 혼잡 발생 시 저장되는 SRTT값인 혼잡 변수를 의미한다.

(그림 3) 혼잡 발생 시 송신 노드(TCP Server)의 혼잡 변수에 SRTT를 저장하는 제안한 알고리즘

(Figure 3) The proposed algorithm storing SRTT to congestion variable of sending node (TCP server) when the congestion is occurred

(그림 4) SRTT로 혼잡을 예측하여 혼잡 윈도우를 감소시키는 제안한 알고리즘

(Figure 4) The proposed algorithm decreasing the congestion window estimating the congestion by the SRTT

4. 시뮬레이션 실험 결과

본 논문의 연구내용에서는 유선 네트워크와

무선 네트워크의 접합지점인 기지국에서 유무선 네트워크의 대역폭 차이로 인한 혼잡상황만을 고려하였고 유선 네트워크상의 모든 라우터들은 정상동작을 하는 이상적인 모델인 경우로 가정하였다. 상기의 환경하에서 제안한 알고리즘 성능 평가를 위하여 NS-2(Network Simulator-2)를 이용하여 실험을 수행하였다. 네트워크 토폴로지는 데이터를 전송하는 송신 노드(TCP Server)와 데이터를 수신하는 무선 단말 노드, 병목 링크 기지국에 해당하는 AP로 구성되어 있다. 송신 노드와 AP 사이는 유선으로 연결되어 있고 링크 대역폭이 5Mb, 전송 지연이 2ms이며 AP와 수신 단말 노드 사이는 4Mb의 대역폭을 가지는 무선 환경으로 구성되어 있고 유무선 대역폭 차이로 인해 유무선 네트워크의 접합부분에 위치한 기지국에서 데이터 오버플로우로 인한 혼잡 발생이 일어나는 환경이다. 이와 비슷한 시뮬레이션 환경으로 에리 보고를 통하여 무선 TCP의 성능을 향상시키는 논문[13]과 위성 네트워크 환경에서 무선 TCP 알고리즘을 분석하는 논문[14] 등이 있다. 아래 <표 1>은 시뮬레이션에 사용한 파라미터들을 나타낸다.

<표 1> 사용된 시뮬레이션 파라미터들

Used simulation tool	NS-2	
Wlan	Mac 802.11	
Propagation model	Shadowing model	
Shadowing path loss beta	2.7	
Shadowing deviation	4.0	
AP to Mobile node distance	50 m	100 m
Packet size	1,000 byte	
Buffer size	50,000 byte	
TCP data protocol	FTP	
TCP filter type	3	
Simulation time	100 sec	
Data send time	1~99 sec	

<Table 1> The used simulation parameters

실험은 TCP Reno와 제안하는 기법을 사용하여 각각 데이터를 전송하였으며 혼잡 윈도우, SRTT의 변화와 버퍼 크기에 따른 손실률과 처리량을 그래프로 나타내었다. (그림 5)와 (그림 6)은 각각 시간에 따른 기존의 TCP Reno와 제안하는 기법의 혼잡 윈도우와 SRTT의 변화를 나타내고 있다.

(그림 5) 기존 TCP Reno와 제안하는 기법의 SRTT 비교

(Figure 5) The comparison of SRTT between the conventional TCP Reno scheme and proposed scheme

(그림 5)에서 SRTT값은 TCP Reno의 경우 버퍼에 쌓이는 패킷과 비례하여 증가하면서 혼잡이 심각하게 발생하였을 경우 SRTT값이 급격히 증가하여 SRTT의 편차가 크게 나타나고 제안하는 기법의 SRTT는 TCP Reno와 같이 버퍼에 쌓이는 패킷과 비례하여 증가하다가 혼잡이 발생하기 전에 혼잡 윈도우를 감소시켜 버퍼에 쌓여있는 패킷이 빠져나가게 되므로 SRTT 값 또한 감소하여 편차가 작게 나타난다.

(그림 6) 기존 TCP Reno와 제안하는 기법의 혼잡 윈도우 비교

(Figure 6) The comparison of congestion window between the conventional TCP Reno scheme and proposed scheme

(그림 6)과 같이 혼잡 윈도우의 크기를 1부터 지수 증가 시키게 되는 Time-out이 제안하는 기법에서 더 줄어들게 되었다. 혼잡으로 인한 Time-out 발생이 줄어들었다는 것은 패킷 손실률의 감소와 처리량의 증가를 야기한다. 기존의

TCP Reno에서 혼잡 윈도우의 최대 크기가 더 크게 그려져 있어서 처리량이 높다고 생각될 수 있지만 아래 (그림 7)과 (그림 8)을 보면 처리량 역시 제안하는 기법이 더 높게 측정되는 것을 볼 수 있다.

(그림 7) AP와 단말 노드 사이의 거리가 50 m일 때 기존의 TCP Reno와 제안하는 기법의 버퍼 크기에 따른 손실률

(Figure 7) The packet loss rate according to the buffer size between the conventional TCP Reno scheme and proposed scheme when the distance between AP and mobile terminal is 50 m

(그림 8) AP와 단말 노드 사이의 거리가 50 m일 때 TCP Reno와 제안하는 기법의 버퍼 크기에 따른 처리량

(Figure 8) The throughput according to the buffer size between the conventional TCP Reno scheme and proposed scheme when the distance between AP and mobile terminal is 50 m

$$Loss\ rate(\%) = \frac{loss\ packet}{all\ packet} \times 100 \quad (7)$$

$$Throughput(Bps) = \frac{recv\ packet}{8/1000/100(sec)} \quad (8)$$

(그림 7)와 (그림 8)은 AP와 수신 단말 노드 사이의 거리를 50m로 설정하고 버퍼 크기를 30, 60, 90으로 늘려가면서 패킷 손실률과 처리량을 비교하였고 식(7), 식(8)은 패킷 손실률과 처리량 계산식을 각각 나타냈다. 위에서 언급했듯이 혼잡에 의한 패킷 손실과 Time-out의 감소로 TCP Reno보다 제안하는 기법의 손실률이 낮게 그려졌다. 버퍼 크기가 커질수록 최대 RTT값 또한 증가하게 되는데 이때 증가한 SRTT값이 혼잡 변수에 그대로 저장되므로 버퍼 크기와 비례하게 된다. 그러므로 버퍼 오버플로우가 일어나기 전에 혼잡을 예측하여 혼잡 윈도우의 크기를 감소시키는 데에는 변함이 없다. 하지만 무선 링크 에러율이 많이 높거나 버퍼 크기가 많이 큰 경우 버퍼 오버플로우에 의한 손실보다 무선 구간에서 패킷 손실이 더 많이 발생하게 되는데 이때 혼잡을 제대로 예측하기가 힘들어진다.

(그림 9) AP와 단말 노드 사이의 거리가 100 m일 때 기존의 TCP Reno와 제안하는 기법의 버퍼 크기에 따른 손실률

(Figure 9) The packet loss rate according to the buffer size between the conventional TCP Reno scheme and proposed scheme when the distance between AP and mobile terminal is 100 m

(그림 10) AP와 단말 노드 사이의 거리가 100 m일 때 기존의 TCP Reno와 제안하는 기법의 버퍼 크기에 따른 처리량

(Figure 10) The throughput according to the buffer size between the conventional TCP Reno scheme and proposed scheme when the distance between AP and mobile terminal is 100 m

(그림 9)와 (그림 10)은 AP와 수신 단말 노드 사이의 거리를 100m로 (그림 7)와 (그림 8)일 때 보다 거리가 두 배 늘려서 무선 구간에서 패킷 손실률이 높은 환경에서 전과 동일한 버퍼 크기로 패킷 손실률과 처리량을 나타냈다. 버퍼 크기가 30, 60일 때는 제안하는 기법이 좋은 성능을 보이지만 버퍼 크기가 90일 때는 TCP Reno보다 패킷 손실률이 높고 처리량이 낮아졌다. 설명한 것처럼 무선 구간에서 패킷 손실이 버퍼 오버플로우에 의한 패킷 손실보다 높을 경우에 혼잡을 예측하기 힘들기 때문에 제안하는 기법의 효율이 떨어지게 된다. 무선 구간에서 패킷 손실이 많이 높아지게 되면 버퍼에 충분한 패킷이 쌓이기 전에 혼잡 변수에 SRTT를 저장하기 때문에 가용 대역폭을 충분히 점유하지 못하고 혼잡 윈도우를 감소시킨다. 따라서 제안하는 기법은 무선 구간에서 패킷 손실보다는 버퍼 오버플로우에 의한 손실이 클 경우에 좋은 효율을 가지는 것을 볼 수 있다.

5. 결론

버퍼와 SRTT를 사용하여 혼잡한 상황을 예측하여 혼잡을 미연에 방지하는 방법을 제안하였다. 적절한 파라미터를 사용하여 혼잡 변수에 저장된 값에 따라 혼잡이 일어나기 전에 혼잡

윈도우를 감소시켜 혼잡한 상황의 발생 빈도를 줄였고 그 만큼 낮은 손실률과 높은 전송 효율이 나타나는 것을 확인하였다. 하지만 버퍼 오버플로우에 의한 패킷 손실이 생기지 않고 높은 무선 링크 에러율로 인한 패킷 손실만 계속해서 생기게 될 경우 충분한 대역폭을 점유하지 못하여 처리량이 감소하게 된다. 이러한 문제는 무선 구간에서 패킷 손실과 버퍼 오버플로우에 의한 손실을 구분해서 혼잡 윈도우를 제어하여 대역폭을 더욱 효율적으로 사용할 수 있는 방안을 생각하여 해결해야 한다. 따라서 향후 앞의 문제점을 개선하여 높은 무선 링크 에러율에서도 최대의 전송효율을 보여줄 수 있는 기법을 연구할 예정이다.

References

- [1] G. Lim, C. Youn, S. Nam and I. Lee, "TCP Congestion Control Based on Timeout Patterns," Trans. KIEE, Vol. 53D, No. 12, Dec 2004.
- [2] J. C. Hoe, "Improving the start-up behavior of a congestion control scheme for TCP," ACM SIGCOMM '96, pp. 270-280, August 1996.
- [3] Z. Wang and J. Crowcroft, "A new congestion control scheme: slow start and search (Tri-S)," ACM Computer Communication Review, vol. 21, issue 1, pp. 32-43, January 1991.
- [4] E. Lee, K. Chung, "Congestion Control Scheme for Efficient Multimedia Transmission in Broadband Wireless Networks", J. Korea Inst. Inf. Commun. Eng., vol. 18, no. 7 : 1599~1609 Jul 2014.
- [5] C. Yoo, D. Kim, "The Study on New Wireless TCP-Westwood Algorithm having Available Bandwidth Estimation within Allowable Range", Journal of Digital Contents Society, vol. 15, no. 2, pp. 147-154, Apr 2014.
- [6] H. Kang, L. Anh, C. Hong, "Multi-path TCP Congestion Control for Streaming Services in Wireless Network", Journal of KIISE, vol. 38, no. 2, pp. 118-123, Apr 2012.
- [7] Y. Zang, et al, "NS-2 Application of Network Simulator", Hongrunc Publishing Company, Jul 2008.
- [8] NS-2 Manual. [Online]. Available: <http://www.isi.edu/nsnam/ns/>, Nov 2011.
- [9] S. Floyd, T. Henderson, and A. Gurtov, "The New Reno Modification to TCP's Fast Recovery Algorithm," IETF RFC 6582, April 2004.
- [10] C. Yeon, "TCP Congestion Control Based on RTT and Timeout", MS degree of hanyang university, Feb 2004.
- [11] Seong Cheol. Kim, Jain. Raj, "Relation between Buffers and RTT", Journal of KICS, vol. 15, no. 1, pp. 894-897, 1996.
- [12] A. Behrouz, "Data Communications and Networking, 5th Edition", McGraw-Hill Education, Oct 2013.
- [13] K. Kim, N. Kim, "An Enhanced Wireless TCP protocol based on Explicit Error Notification", Journal of KICS, vol. 26, no. 12B, pp. 1656-1664, 2001.
- [14] M. Park, H. Choo, "Performance Analysis of Wireless TCP Algorithm in Networks with Long Delay", Korean Society for Internet Information, vol. 8, no. 1, pp. 41-44, Jun 2007.

임 민 기

2009년 3월 - 현재 : 강원대학교 IT
대학 전기전자공학부 학부과정

관심분야 : 차세대 이동통신 및 차세대 무선 네트워크

김 동 회

2005년 : 고려대학교 전파공학과(공학박사)

1989년 1월 ~ 1997년 1월 : 삼성전자 전임연구원

2000년 8월 ~ 2005년 8월 : 한국전자통신연구원 선임
연구원

2006년 3월 ~ 현재 : 강원대학교 IT대학 전기전자공학
부 전자통신학과 교수

관심분야 : 이동통신 및 무선 네트워크 등