고속 TCP에서 혼잡으로부터 혼잡 제어창의 빠른 회복 메커니즘

전진영°, 서유화, 강정호, 피영수, 신용태
숭실대학교
{nurnadly°, zzarara, kjho80, coolps, shin}@cherry.ssu.ac.kr

A Study of Quickly Recovering Methods from Low Congestion Windows in High Speed TCP

Jinyoung Jeon°, Youhwa Seo, Jungho Kang, Youngsu Pi, Youngtae Shin
Dept. of Computing, Soongsil University

요 약

미래의 인터넷이 저속 네트워크에서 고속 네트워크로 그 특성이 변하고 있음에 반해 비교적 저속 네트워크에 최적화된 TCP의 성능 개선은 이루어지지 않고 있어 미래 인터넷에서 TCP의 효율적인 동작을 보장하기 어려운 실정이다. 이를 위해 본 논문에서는 기존 AIMD 메커니즘을 수정한 HSTCP와 네트워크로부터 명확한 피드백을 요하는 새로운 프로토콜인 XCP에 대해 살펴보고 고속 네트워크에 적합한 개선된 HSTCP 메커니즘을 제안한다.

1. 서 론

인터넷의 급속한 발전 과정을 네트워크 관점에서 볼 때 혼잡제어(Congestion Control)는 항상 중요한 주제로서 논의되어 왔다. 미래의 인터넷을 구성하는 네트워크는 광 대역 링크들을 비롯하여 긴 지연시간을 갖는 위성통신 등의 다양한 무선 환경으로 구성될 것이 예고되고 있다. 이렇듯 네트워크의 특성이 변하고 있음에 반해 비교적 저속 네트워크에서 최적화된 TCP의 성능 향상은 이루어지지 않고 있어 이에 대한 논의가 필요한 시점이라고 판단된다.

널리 알려진 바와 같이 표준 TCP는 AIMD(Addictive Increase Multiplicative Decrease) 방식으로 혼잡제어 창(Congestion Windows)을 조절하며 세그먼트(Segments)를 전송한다. 이러한 AIMD 방식에서 혼잡제어 창의 역할은 혼잡 제어와 형평성 제어(Fairness Contrcl)에 있다. 그런데 AIMD 방식의 TCP가 혼잡제어 창을 조절하는데 이용하는 기준이 수학적 알고리즘에 의해 결정되는 것이 아니라 많은 실험값을 참고로 설정되었기 때문에 실험 환경이었던 저속 네트워크에 최적화된 것이라고 할 수 있다. 이 때문에 표준 TCP는 미래 인터넷을 구성하는 고속 네트워크 환경에서 효율적인 동작을 보장할 수 없다. 본 논문에서는 이와 관련된 연구들을 소개하고 HSTCP(High Speed TCP) [1]를 바탕으로 낮은 혼잡 제어 창으로부터 좀 더 빠른 회복이 가능한 메커니즘을 제안한다.

본 논문은 2장에서 기존 AIMD 방식을 따르는 HSTCP와 네트워크로부터 명확한 피드백을 요하는 새로운 프로토콜인 XCP(eXplicit Control Protocol)[2][3]에 대해 알아보고 3장에서는 이들을 바탕으로 고속 네트워크에 적합한 수정된 HSTCP 메커니즘을 제안한다. 마지막으로 4장에서는 결론 및 향후 연구방향으로 구성된다.

2. 관련 연구

2.1 HSTCP (High Speed TCP)

현재 표준 TCP의 혼잡제어 메커니즘은 고속 네트워크에서 효율적인 동작을 보장할 수 없다. 대략적으로 TCP 메커니즘에서 RTT당 평균 혼잡 제어창의 크기는 성능(Throughput)으로 볼 수 있는데, 예를 들어 1500 bytes 패킷들을 100ms RTT(Round Trip Time)로 보내기 위한 표준 TCP 연결이 10 Gbps의 성능(Throughput)을 유지하기 위해서는 평균 혼잡제어 창의 크기가 83,333 세그먼트가 필요하고 한번의 혼잡이 발생할 때마다 5,000,000,000 패킷들을 손실하게 된다 [1]. 이러한 시뮬레이션 수치는 매우 비현실적인 것으로 S. Floyd는 이를 해결하기 위하여 기존 AIMD 방식의 틀 위에서 TCP의 응답함수(response function)를 수정한 TCP 메커니즘을 제안하였다 [1]. 수정된 TCP 메커니즘은 *Low_Window*, *High_Window*, *High_P*라는 세 가지 기준값을 이용하여 현재 혼잡제어 창이 Low_Window보다 작거나 같을 때는 표준 TCP 응답 함수를 이용하고 클 때는 수정된 응답함수를 이용하는 방안을 제안하였다. 수정된 응답함수의 혼잡제어 창의 증가는 수식 (1), (2)를 통해 결정된다. S는 상수 값이고 W는 혼잡 제어창이 *Low_Window*보다 클 때의 평균 혼잡 제어창의 크기를 나타낸다. *Low_P*는 *Low_Window*에서의 패킷 손실률을 의미한다.

$$W = (p/Low\_P)^S \times Low\_Window \quad \text{----------(1)}$$

$$S = \frac{(\log High\_Window - \log Low\_Window)}{(\log High\_P - \log Low\_P)} \quad \text{---(2)}$$

이 제안의 특징은 혼잡제어 창을 조절하는데 패킷 손실률이 반영된다는 것이다. 또한 기존의 AIMD 메커니즘을 그대로 이용하기 때문에 형평성(Fairness)에 있어서도 기존 TCP와 수정된 TCP 사이를 제외하면 표준 TCP가 달성한 수준의 형평성을 제공할 수 있다.

2.2 XCP (eXplicit Control Protocol)

XCP는 기존의 ECN(Explicit Congestion Notification)

을 통한 혼잡 제어 방법에서 발전된 제안이다 [2]. 이것의 특징은 ECN이 1 비트를 이용해 혼잡 유/무만을 알렸던 것에 반해 병목 지점의 라우터가 혼잡 정도를 수치로서 송신자에게 알려주는 방식이다. 또한 이 제안은 이용제어(utilization control)와 형평성제어(fairness control)를 분리하여 처리한다. 이용제어는 네트워크의 여분 대역폭과 피드백 지연 시간에 따라 전송 정도를 결정하기 때문에 진동(oscillation)현상을 막고 고 대역폭이나 긴 지연 시간 등을 지니는 네트워크에서 안정성을 제공한다. 그리고 이러한 네트워크의 피드백을 통한 네트워크 자원 이용은 비교적 높은 자원 이용률을 얻을 수 있다. 형평성 제어는 불공정한 자원 이용률을 보이는 흐름(flow)에 대하여 프로토콜이 적절히 대역폭을 재 할당함으로써 이룰 수 있다. 표준 TCP가 개념적으로는 독립적인 효율성(efficiency)과 형평성을 하나의 제어 방법으로서 동시에 얻고자 하여 성능 개선에 어려움에 있었던 것에 비추어 볼 때 이러한 시도는 매우 효과적이라고 할 수 있다.

3. 제안 메커니즘

TCP 메커니즘에서 송신자가 혼잡을 판단하는 기준은 타임아웃과 3개의 중복된 ACK를 받았을 때이다. 이 기준은 모두 패킷이 전송 중 손실되었음을 의미하기 때문에 TCP의 혼잡 제어에서 패킷 손실률을 고려하는 것은 근본적으로 올바른 접근 방향이라고 생각된다. 바로 이러한 점이 모든 라우터와 말단을 수정해야 하는 XCP에 비해 HSTCP가 설득력을 얻고 있는 이유이다. 본 논문에서는 HSTCP가 취한 접근 방향을 토대로 혼잡 상황에서 혼잡 제어창의 빠른 회복을 위한 개선된 HSTCP를 제안한다.

3.1 개선된 TCP 응답 함수(Response Function)

제안하는 응답함수는 S. Floyd가 제안한 HSTCP의 응답 함수를 개선하였다. HSTCP에서는 혼잡이 발생하여 혼잡 제어 창을 초기 값으로 설정한 후 *Low_Window* 보다 작거나 같을 때까지는 표준 TCP 응답함수에 의해 혼잡 제어창이 증가되고 그 이후부터는 수정된 TCP 응답함수에 의해 증가된다. 달성하고자 하는 성능인 10Gbps를 유지하기 위해서는 전술한 바와 같이 83,333 MSS(Maximum Segment Size) 크기의 평균 혼잡 제어창이 요구된다. 본 논문에서는 혼잡 제어창이 *Low_Window*부터 달성하고자 하는 혼잡 제어창의 크기(*High Window*)까지 회복하는 지연 시간을 줄이고자 2단계 회복 메커니즘을 제안한다. 이를 위해 본 논문에서는 *QR_cwnd, QR_aw, QR_a* 3가지 기준값을 이용하였다. 현재 혼잡 제어창(*C_wnd*)이 *Low_Window*보다 크고 *QR_cwnd*보다 작거나 같을 때 까지는 ACK를 받을 때마다 HSTCP의 증가량(*aw*[1])에 *QR_aw*를 더해서 증가하고 *C_wnd*가 *QR_cwnd*보다 크고 *High_Window*보다 작을 때 까지는 HSTCP 증가량(*aw*[1])에 *QR_a* 만큼을 곱해 증가한다. *QR_aw*와 *QR_a*을 위한 수식은 (5), (6)과 같다.

$$aw = High\_Window^2 * 2.0 * High\_P * 2 * bw/(2 - bw) \quad \text{----------(3)[1]}$$

$$bw = \frac{(High\_Decrease - 0.5) * (logw - logW)}{(logW\_1) - log(W) + 0.5} \quad \text{(4)[1]}$$

$$QR\_aw = HS\_Window^{(C\_wnd/High\_Window)} \quad \text{--------(5)}$$

$$QR\_a = High\_Window/C\_wnd \quad \text{----------(6)}$$

(5)식은 첫 번째 단계에서 HSTCP의 응답함수에 의해 증가되는 혼잡 제어 창의 크기보다 약 2배 정도 크게 증가시키게 되고, (6)식는 2단계에서 증가분을 점차 줄이게 된다. 위 과정에서 제안한 혼잡 제어창의 증가 메커니즘을 정리하면 [그림-1]와 같다.

```
if Low_Window < C_wnd  <= QR_cwnd then
    C_wnd = C_wnd + aw + QR_aw
if QR_cwnd < C_wnd <= High_Window then
    C_wnd = C_wnd + aw * QR_a
```

[그림 1] 개선된 혼잡 제어창의 증가 메커니즘

이와 같은 제안은 *C_wnd*가 *QR_cwnd*까지는 ACK당 혼잡 제어창의 증가분 기울기가 증가 상태에 있게 되고 *QR_cwnd*부터 *High_Window*까지는 기울기가 감소 상태에 있게 되는 2단계 회복 상태를 취한다. 이는 결과적으로 혼잡 제어 창이 *Low_Window*에서 달성하고자 하는 *High_Window*까지 증가하는데 걸리는 시간을 줄일 수 있다.

3.2 혼잡 제어창의 증가 시간 분석

개선된 응답 함수와 HSTCP 응답함수의 혼잡 제어창의 증가 시간을 분석하기 위하여 [표-1]을 입력 값으로 하여 (5), (6), (7), (8)식을 통해 증가 시간을 분석하였다.

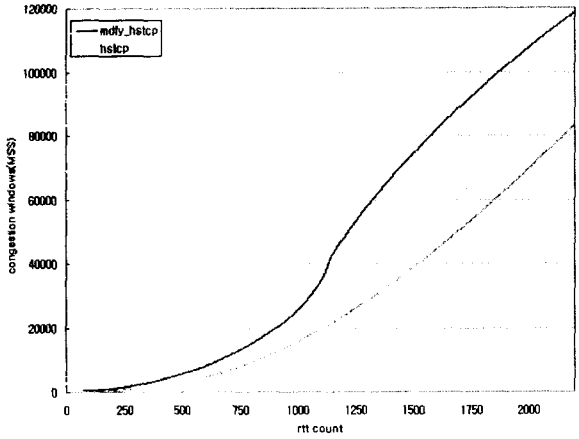
Variable	Value
High_Window	83000 MSS segments
High_P	10^{-7}
High_Decrease	0.1
w	C_wnd
W	38
W_1	*High_Window*
RTT	100 msec
QR_cwnd	*High_Window*/2

[표-1] 분석을 위한 입력 값

[그림-2]는 [표-1]의 입력 값으로 HSTCP의 응답함수와 개선된 응답함수에 의해 증가하는 혼잡 제어 창을 비교한 그래프 이다.

평균 RTT가 100ms이고 고속 TCP에서 혼잡이 발생했을 때 개선된 HSTCP의 메커니즘에 의해 혼잡 제어 창이 증가할 경우 *High_Window*까지 달성하는데 시뮬레이션 수치로서 약 161.1 초가 소요된다. 이는 일반

HSTCP가 219.4 초가 소요되는 것에 비추어 볼 때 좀더 빠른 회복 성능을 보임을 알 수 있다. 이러한 빠른 회복 성능은 자원의 이용률을 높일 수 있어 고속 TCP에 적합하다.



[그림 2] 혼잡 제어창의 증가 그래프 (개선된 응답함수에 의한 증가: mdfy_hstcp, HSTCP의 응답함수: hstcp)

3.3 개선된 HSTCP의 형평성에 대한 고려사항

TCP의 수정이 어려운 점은 수정된 TCP 흐름 간의 형평성뿐만 아니라 표준 TCP 흐름과 수정된 TCP 흐름간의 형평성을 고려해야 하는 것이다. 이는 기존에 사용되고 있는 프로토콜과의 호환성을 위해 매우 중요한 사항이며 많은 네트워크 프로토콜이 지향하는 철학이기도 하다.

그런데 불행히도 표준 TCP 흐름과 수정된 HSTCP간의 형평성을 유지하기란 매우 어려운 일이다. 최악의 경우 수정된 HSTCP 흐름이 이미 진행 중에 있고 표준 TCP의 흐름이 이제 막 시작될 때 수정된 HSTCP의 높은 자원 이용률 때문에 표준 TCP의 자원 이용에 제약이 있을 수 있다. 또한 두 흐름이 모두 혼잡 회피(congestion avoidance)상태에 있다고 해도 두 흐름의 ACK당 혼잡 제어 창의 증가율이 다르기 때문에 이는 형평성에 위배되는 결과를 초래한다. 이는 [표-2]을 통해 잘 나타나 있다.

RTT count	mdfy_HSTCP	Standard TCP
100	284	100
200	930	200
300	1998	300
400	3531	400
500	5565	500
600	8131	600
700	11270	700
800	15054	800
900	19651	900
1000	25378	1000
1100	34299	1100
1200	47899	1200
1300	57612	1300
1400	66338	1400
1500	74332	1500
1600	81749	1600

[표-2] 표준 TCP와 수정된 HSTCP 사이의 형평성

이러한 형평성을 고려할 때 가장 중요시 되는 기준값은 HSTCP에서 *Low_Window* 변수이다. 이 값을 기준으로 표준 TCP의 응답 함수와 수정된 HSTCP의 응답 함수 사이의 작동 여부가 결정되기 때문이다. 이 값이 클 경우는 형평성은 좀 더 유지되겠지만 목표 성능에 도달하기 위한 시간이 좀 더 길어질 것이고 이 값이 작을 경우 목표 성능에는 좀 더 빨리 도달할 수 있지만 형평성에 위배된다. 분명하게 개선된 HSTCP와 표준 TCP 사이의 형평성은 위배되고 있다. 그러나 표준 TCP의 철학이 흐름들 사이에 공정성을 추구하는 것이라고 하여도 실제 네트워크에서 TCP의 공정함이 잘 유지되지 않는다는 점과 수정된 HSTCP가 고속 네트워크에서 얻을 수 있는 성능의 향상을 생각해 볼 때 이러한 형평성의 위배는 관용할만한 수준이다. 그러나 이러한 형평성의 위배는 엄연히 TCP 기본 철학에 위배되기 때문에 이에 대한 연구가 좀 더 수행되어야 한다.

4. 결론 및 향후 연구방향

본 논문에서는 미래 네트워크의 특성이 저속 네트워크에서 고속 네트워크로 변함에 따라 전송 계층 프로토콜인 TCP가 효율적으로 동작하지 못함을 보이고 이를 해결하기 위한 연구들을 살펴보았다. HSTCP는 기존 시스템과의 호환성을 고려한 해결 방안을 제시하였으며, XCP는 네트워크가 제공해주는 명확한 피드백을 바탕으로 한 해결 방안을 제시하였다. 현재 TCP의 말단간의 혼잡 제어 메커니즘은 패킷 손실의 징후를 예측하여 작동되는 메커니즘이기 때문에 본 논문에서는 혼잡 제어에 패킷 손실률이 반영된 HSTCP의 제안을 바탕으로 낮은 혼잡 제어 창으로부터 좀 더 빠른 회복을 위한 메커니즘을 제안하였다. 이를 위하여 QR_cwnd, QR_aw, QR_a의 세 가지 기준값을 이용해 2단계로 혼잡 제어 창을 회복함으로써 좀 더 빠른 회복 성능을 보였다. 그러나 표준 TCP와 수정된 HSTCP 사이의 형평성의 유지를 위한 연구가 좀 더 필요하다.

참고 문헌

[1] S. Floyd, "HighSpeed TCP for Large Congestion Windows", RFC3649, December 2003.

[2] Dina Katabi, Mark Handley and Charlie Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks", August 2002.

[3] Souza E, Agarwal D, "A HighSpeed TCP Study: Characteristics and Deployment Issues", LBNL Technical Report LBNL-53215.