Mininet을 활용한 다양한 TCP 혼잡제어의 플로우 간 성능 수렴 특성 분석

최민성, 김병철, 이재용
충남대학교 정보통신융합학부
emforjsdlek@o.cnu.ac.kr, byckim@cnu.ac.kr, jyl@cnu.ac.kr

Evaluation of the Convergence Characteristics between Flows for Various TCP Congestion Control Schemes using Mininet

Minsung Choi, Byungchul Kim, Jaeyong Lee
Chungnam National Univ. School of Information Communications Convergence Eng

요 약

본 논문은 동일한 TCP 혼잡제어 알고리즘 환경에서 전송 시작 시점 차이가 흐름 간 공정성 수렴에 미치는 영향을 분석한다. Reno, CUBIC, BBRv1, BBRv3을 대상으로 두 흐름을 서로 다른 시점에 시작하는 실험을 Mininet환경에서 수행하였다. 실험 결과, Reno와 CUBIC은 손실 기반 특성으로 인해 수렴까지 장시간이 소요된 반면, BBR 계열은 상대적으로 짧은 시간 안에 공정성에 도달하였다. 그러나 최신 버전인 BBRv3는 BBRv1보다 수렴 속도가 늦어 공정성 확보 측면에서는 오히려 불리한 결과를 보였다. 본 연구는 동일 알고리즘 간 경쟁 상황에서 공정성 수렴 특성을 정량적으로 규명하였다는 점에서 의의가 있다.

Ⅰ. 서 론

인터넷 망에서 다수의 전송 흐름이 동시에 자원 사용은 경쟁하는 상황은 매우 일반적이며, 이 과정에서 혼잡제어 알고리즘은 자원 공유의 효율성과 공정성을 좌우한다. TCP는 인터넷에서 가장 널리 사용되는 전송 프로토콜로서, 다양한 혼잡제어 알고리즘이 제안되어 왔다. 이러한 알고리즘은 링크 대역폭 활용을 극대화하는 동시에 흐름 간의 공정한 자원 배분을 보장하는 것을 목표로 한다.

그러나 실제 네트워크에서는 모든 흐름이 동일한 시점에 시작하지 않고, 서로 다른 시점에서 합류하는 경우가 많다. 이때 선행 흐름이 우위를 차지하고, 후속 흐름은 불리한 조건에서 시작하는 비대칭 현상이 나타난다. 따라서 동일한 알고리즘 간 경쟁 상황에서 전송 시작 시점 차이가 공정성 수렴에 어떤 영향을 주는지 규명하는 것은 실질적인 의미를 가진다.

본 연구는 이러한 문제의식을 바탕으로 Reno[1], CUBIC[2], BBRv1과 BBRv3[3] 네 가지 TCP 혼잡제어 알고리즘을 대상으로, 동일 알고리즘 환경에서 흐름이 서로 다른 시점에 합류할 때 공정성 수렴 특성을 분석하였다.

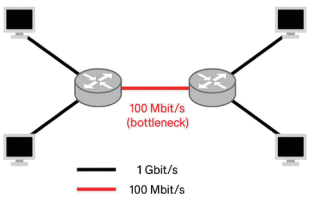
Ⅱ. 기존 TCP 혼잡제어 연구 동향

기존 연구들은 주로 서로 다른 알고리즘 간 경쟁 또는 네트워크 환경 변수(대역폭, 지연, 버퍼 크기 등)가 성능에 미치는 영향을 분석하는 데 집중해 왔다[4][5]. 이러한 연구는 혼잡제어 메커니즘의 특성을 밝히는 데 중요한 기여를 하였으나, 동일한 알고리즘 간 경쟁에서 전송 시작 시점의 차이가 공정성에 미치는 영향을 정량적으로 분석한 연구는 상대적으로 부족하다.

특히 실시간 트래픽이나 대규모 데이터 전송 환경에서는 흐름이 순차적으로 합류하는 경우가 빈번하다. 따라서 시작 시점 차이에 따른 공정성 수렴 과정을 규명하는 것은 TCP 혼잡제어 연구에서 중요한 의미를 갖는다.

Ⅲ. 실험망 구성

본 연구에서는 동일한 TCP 혼잡제어 알고리즘 환경에서 전송 시작 시점 차이가 공정성 수렴에 미치는 영향을 분석하기 위해 <그림1>과 같이 덤벨(dumbbell) 구조 기반의 에뮬레이션 실험을 수행하였다. 병목 링크의 대역폭은 100 Mbit/s, 왕복 지연시간(RTT)은 100 ms, 큐 버퍼 크기는 0.5 BDP로 설정하였다. 실험 환경은 Ubuntu 22.04.5 LTS와 Linux kernel 6.13.7+에서 구축되었으며, Mininet 2.3.1b4을 통해 네트워크 토폴로지를 구성하고 실험하였다[6]. 가상머신 환경은 AMD Ryzen 5 5600X 프로세서(6 cores @ 3.7 GHz)와 4 GB RAM에서 구동되었다.



<그림 1> 덤벨(dumbbell) 구조

Ⅳ. 실험대상 혼잡제어 알고리즘

본 실험에서는 Reno, CUBIC, BBRv1, BBRv3 네 가지 혼잡제어 알고리즘을 대상으로 공정성 수렴 특성을 비교하였다. Reno는 AIMD(Additive Increase, Multiplicative Decrease) 원리를 기반으로 한 전형적인 손실 기반 알고리즘으로, 혼잡윈도우를 선형적으로 증가시키다가 손실이 발생하면 배수 감소를 적용한다[1]. CUBIC은 대역폭이 큰 링크에서도 빠른 확장이 가능하도록 혼잡윈도우를 3차 함수 곡선에 따라 조절하는 방식이며, 현재 리눅스의 기본 혼잡제어 알고리즘으로 널리 사용되고 있다[2]. BBRv1은 손실 이벤트 대신 경로의 대역폭과 최소 RTT를 추정하여 목표 전송률을 설정하는 모델 기반 접근 방식을 채택하며, 손실 기반 방식과 달리 경로 용량을 직접적으로 활용하려는 특성을 가진다. BBRv3는 BBRv1의 불안정성과 불공정성을 개선하기 위해 제안된 최신 버전으로, 대역폭 추정과 프로브 주기 조정이 포함되어 있다[3].

Ⅴ. TCP 혼잡제어의 수렴 성능 평가

실험 절차는 두 개의 흐름을 생성하여 첫 번째 흐름을 시작한 후 30초

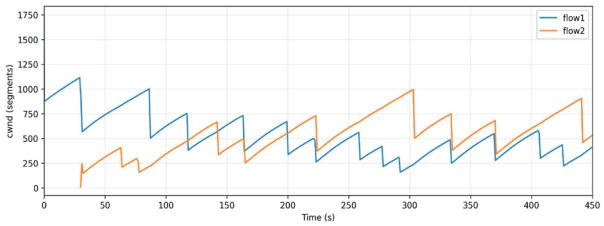
뒤 두 번째 흐름을 합류시키는 방식으로 진행되었다. 이는 선행 흐름이 이미 대역폭을 점유한 상태에서 후속 흐름이 경쟁에 참여하는 상황을 재현한다. 공정성 수렴 시간은 두 혼잡윈도우(cwnd)가 장기 평균값에 근접하며 교차하는 최초 시점으로 정의한다.

실험 결과, Reno의 수렴 시간은 <그림2-a>에서 보듯이 약 90초로 측정되었으며 손실 발생 시 윈도우가 크게 줄어들어 회복이 느리게 진행되었다. CUBIC은 <그림2-b>와 같이 약 40초 내외에서 수렴하였으며, Reno보다 빠른 수렴 특성을 보였으나 여전히 손실 이벤트 이후 회복 과정이 지연되는 양상이 나타났다. BBR 계열 알고리즘은 <그림3>에서와 같이 훨씬 짧은 시간 내에 공정성에 도달하였다. BBRv1(그림3-a)은 약 15초만에 수렴하여 가장 빠른 성능을 보였으며, BBRv3(그림3-b)는 약 25초에서 수렴하여 BBRv1보다 다소 늦었으나 Reno나 CUBIC에 비해서는 월등히 짧았다.

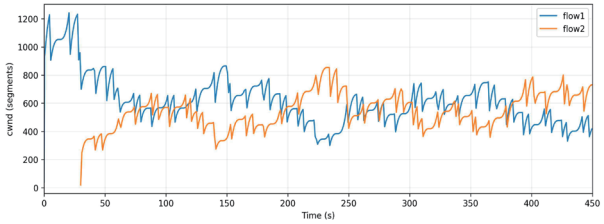
알고리즘별 수렴 시간 비교에서 손실 기반과 모델 기반의 차이는 뚜렷하다. Reno와 CUBIC은 초기 혼잡윈도우 증가가 제한적이고 손실 발생 시 회복이 지연되므로, 후속 흐름이 선행 흐름과의 성능 격차를 줄이기까지 오랜 시간이 소요된다. 반면 BBRv1과 BBRv3는 손실 신호와 무관하게 경로 특성을 추정하여 전송률을 조정하므로, 수십 초 내에 두 흐름이 유사한 성능 수준에 도달한다. 특히 Reno와 BBRv1 사이의 차이는 약 6배로, 혼잡제어 방식의 구조적 차이가 공정성 확보 속도에 직접적인 영향을 미친다는 점을 보여준다.

공정성 수렴 속도는 네트워크 활용성과 서비스 품질에도 연결된다. 후속 흐름의 수렴이 지연될 경우, 스트리밍이나 실시간 응용과 같은 서비스에서는 장시간 동안 자원 불균형이 발생할 수 있다. 반대로 빠른 수렴을 보이는 알고리즘은 전체적인 처리율을 안정적으로 분배하여 사용자 경험의 일관성을 유지할 수 있다. Reno와 CUBIC은 대역폭 활용도가 장기간 비효율적으로 유지될 가능성이 크며, BBR 계열은 보다 빠른 자원 공유를 통해 다양한 응용 환경에서 유리한 결과를 제공할 수 있다.

이러한 결과는 동일한 혼잡제어 알고리즘 환경에서도 전송 시작 시점 차이가 흐름 간 자원 공유에 실질적인 영향을 준다는 점을 보여준다. 손실 기반 알고리즘은 공정성 확보까지 오랜 시간이 소요되어 후속 흐름이 장기간 불리한 상태에 놓이게 된다. 반면, BBR 계열은 짧은 시간 안에 자원이 균등하게 분배되며, 그중 BBRv1은 빠른 수렴을 통해 단기적 이점을, BBRv3는 다소 늦은 수렴을 통해 장기적 안정성을 제공한다. 이러한 비교는 혼잡제어 알고리즘을 선택하거나 개선하는 과정에서 처리율뿐 아니라 공정성 수렴 속도와 안정성 간의 균형을 함께 고려해야 함을 시사한다.

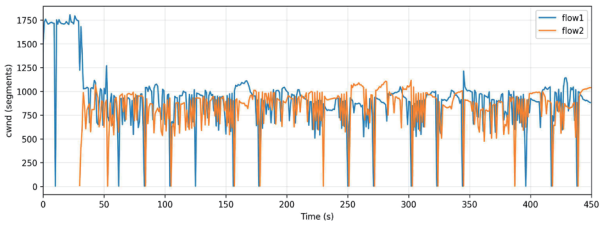


(a) Reno

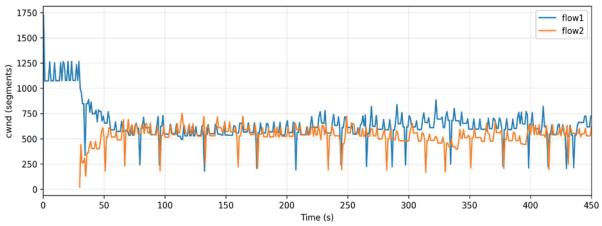


(b) CUBIC

<그림 2> Reno, CUBIC알고리즘의 혼잡윈도우(cwnd) 변화



(a) BBRv1



(b) BBRv3

<그림 3> BBR 알고리즘의 혼잡윈도우(cwnd) 변화

Ⅵ. 결론

본 논문은 동일한 TCP 혼잡제어 알고리즘 간 경쟁에서 전송 시작 시점 차이가 공정성 수렴에 미치는 영향을 분석하였다. Reno와 CUBIC은 손실 기반 특성으로 인해 수렴까지 장시간이 소요되었으며, BBR 계열은 상대적으로 빠른 수렴을 보였다. 특히 BBRv3는 상대 흐름의 성능저하 방지 및 안정성 확보를 위해 설계된 특성으로 인해 BBRv1보다 늦게 균형점에 도달하는 모습을 보였다. 이러한 결과는 동일 알고리즘 환경에서도 전송 시점 차이가 공정성에 중요한 영향을 미치며, 알고리즘별 구조적 차이가 수렴 속도에 직접적으로 작용함을 보여준다. 본 연구는 향후 TCP 혼잡제어 개선과 공정성 확보 방안을 모색하는 데 기초 자료로 활용될 수 있다. 또한, 다양한 RTT·버퍼 조건 및 실제 네트워크 트래픽 환경으로 확장한 추가 연구가 필요하다.

참 고 문 헌

[1] M. Allman, V. Paxson, and W. Stevens, “TCP Congestion Control,” RFC 5681, Sept. 2009.

[2] S. Ha, I. Rhee, and L. Xu, “CUBIC: A New TCP-Friendly High-Speed TCP Variant,” in *Proc. ACM SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64 - 74, 2008.

[3] Google, “BBR: Bottleneck Bandwidth and Round-trip propagation time,” GitHub repository, [Online]. Available: https://github.com/google/bbr

[4] D. Zeynali, E. N. Weyulu, S. Fathalli, B. Chandrasekaran, and A. Feldmann, “Promises and Potential of BBRv3,” Proc. ACM SIGCOMM, 2023.dwd

[5] W. Hong, “Measurement and Analysis of Throughput for BBRv3 on Emulation-Based Testbed,” KREONET Center, KISTI, Oct. 2024.

[6] Mininet, Available: https://mininet.org/, Accessed: Oct. 17, 2025.