(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl.[7]
H04B 7/26

(45) 공고일자 2005년11월03일
(11) 등록번호 10-0526187
(24) 등록일자 2005년10월27일

(21) 출원번호 10-2003-0072784
(22) 출원일자 2003년10월18일

(65) 공개번호 10-2005-0037298
(43) 공개일자 2005년04월21일

(73) 특허권자 삼성전자주식회사
경기도 수원시 영통구 매탄동 416

(72) 발명자 신진현
서울특별시강북구미아2동791-3251

문병인
경기도수원시팔달구망포동동수원엘지빌리지104-1401

조성연
서울특별시동작구신대방1동경남교수아파트103-1704

유혁
서울특별시성북구성북동안암동5가1고려대학교

유시환
서울특별시성북구성북동안암동5가1고려대학교

최진희
서울특별시성북구성북동안암동5가1고려대학교

(74) 대리인 김동진

심사관 : 하은주

(54) 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기위한 조절 방법

요약

본 발명은 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법에 관한 것으로, 현재전송율(cur_bw)과 이전전송율(base_bw)의 차이값(Diff)을 계산하는 제 1단계; 상기 이전전송율의 표준편차(basebw_var)를 계산하여 상한임계값(upper threshold)과 하한임계값(lower threshold)을 설정하는 제 2단계; 상기 차이값과 상기 상하한 임계값의 크기를 비교한 결과에 따라, 상기 이전전송율을 상기 현재전송율로 업데이트하는 제 3단계를 포함하는 것을 특징으로 한다.

대표도

도 5

색인어

느린 시작, 혼잡 회피, 현재 전송율, 이전 전송율

명세서

도면의 간단한 설명

도 1은 모바일 애드 혹 네트워크 시스템을 나타내는 개념도이다.

도 2a는 종래 기술에 따른 TCP-Reno 버전에서의 전송율 변화를 나타내는 도면이다.

도 2b는 종래 기술에 따른 TCP-Vegas 버전에서의 전송율 변화를 나타내는 도면이다.

도 3은 종래 기술에 따른 TCP-Reno 버전과 TCP-Vegas 버전에서의 전송 실패 패킷의 개수를 비교한 도표이다.

도 4는 현재 쓰루풋과 최대 쓰루풋를 비교한 그래프이다.

도 5는 본 발명의 일 실시예에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법을 나타내는 흐름도이다.

도 6은 본 발명의 또다른 일 실시예에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법을 나타내는 흐름도이다.

도 7a 와 7b는 본 발명의 일 실시예에 따른 쓰루풋의 결과를 종래의 TCP-Reno 버전과 TCP-Vegas 버전에 따른 쓰루풋 결과와 비교한 도면이다.

도 8a 와 8b는 본 발명의 일 실시예에 따른 유실된 패킷의 수를 종래의 TCP-Reno 버전과 TCP-Vegas 버전에 따른 유실된 패킷의 수와 비교한 도면이다.

*도면의 주요 부분에 대한 설명

40: 최대 쓰루풋 42: 현재 쓰루풋

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법에 관한 것으로, 더욱 상세하게는 현재전송율(cur_bw)과 이전전송율(base_bw)의 차이값(Diff)을 계산하고, 이전전송율의 표준편차(basebw_var)를 계산하여 상한임계값(upper threshold)과 하한임계값(lower threshold)을 설정한 후, 그 차이값과 상하한 임계값의 크기를 비교한 결과에 따라 혼잡윈도우값(cwnd)을 증가 또는 감소시킨 후 이전 전송율을 현재 전송율로 업데이트함으로써, 애드 혹 네트워크 환경에 적합한 최적의 전송율 조절 기법을 제공하는 것이다.

종래의 공개특허로는 한국공개특허 2001-058119가 있는데, 이는 혼잡윈도우(CWND)의 선형적인 증감과 더불어 지수적 증감을 함께 고려하므로써, TCP- vegas의 수렴속도를 개선하여 TCP의 성능을 향상시킨 전송제어프로토콜 베가스(TCP Vegas)의 혼잡제어방법에 관한 것이다.

또한, 한국공개특허 2003-044465는 χ1, χ2를 α와 β에 추가함으로써 네트워크의 급격한 혼잡을 줄이고 네트워크를 효율적으로 사용할 수 있도록 한 TCP-Vegas를 이용한 전송 제어 프로토콜의 혼잡제어방법에 관한 것이다.

그리고, 외국의 공개 특허로는 미국공개특허 2002-0154602(무선 링크 상에 TCP 성능을 향상시키기 위한 방법)과 일본 공개특허 평11-177618(SACK 정보를 이용하여 cwnd의 삭감량을 변화시키는 경우, 동일 네트워크상에 혼재한 Reno 알고리즘에 의해 대역 확보의 공평성을 얻은 폭주제어 방법) 등이 있다.

모바일 애드 혹 네트워크(Mobile ad hoc network, 이하 MANET이라 명한다.) 환경이란 고정된 인프라 구조없이 필요에 의해 동적으로 형성되는 네트워크가 이동 단말을 중심으로 형성되는 네트워크 환경을 말하는데, 도 1은 애드 혹 네트워크 시스템을 나타내는 개념도를 나타내고 있다. IEEE802.11을 사용하는 모바일 애드 혹 네트워크에서는 통신 채널을 모든 노드가 공유하게 된다. 따라서, 동일한 통신 영역 내에 있는 모든 노드들은 동일한 채널을 통하여 통신을 하게 되어 전송된 데이터가 서로 충돌하거나 유실되기 쉽고, 무선 네트워크의 특성상 링크 계층의 전송 신뢰도가 크게 떨어지게 된다. 송신자는 수신자에게 패킷을 전송하고, 수신자는 상기 전송된 패킷에 대한 ACK-응답 패킷을 송신자에게 전송하게 된다. 이 때, 수신자는 현재까지 수신한 연속적인 패킷의 시퀀스 번호의 최대값을 유지하여 ACK-응답 패킷을 전송하므로 일부 ACK-응답 패킷이 유실되더라도 송신자가 전송하는 양을 줄이지 않도록 한다.

TCP는 TCP/IP 아키텍처 중 전송층(transport layer)에 속하는 프로토콜로서 주로 데이터의 흐름(flow)과 에러(error) 제어를 담당하고 있다. 특히, TCP는 응용 프로그램에 의해 전송될 데이터가 버퍼에 저장되면 윈도우 크기만큼만 전송된다. 이 때, 윈도우의 크기는 목적지 호스트 또는 네트워크 혼잡도(network congestion)의해 결정되고 증가 또는 감소될 수 있다. 특히 TCP에서는 슬라이딩 윈도우 방식을 이용하는데, 이러한 윈도우 개념을 이용함으로써 무선 링크의 사용성(utilization)을 극대화할 수 있게 된다.

종래 기술로서 TCP혼잡 제어 알고리즘은, 초기에 전송율을 단순히 지수적으로 증가시킴으로써 혼잡 상태가 급격히 발생하고 혼잡 상태에서 전송율을 무조건 반으로 줄임으로써 네트워크를 효율적으로 사용하지 못하였다.

한편, 도 2a는 종래 기술에 따른 TCP-Reno 버전에서의 전송율 변화를 나타내는 도면이며, 도 2b는 종래 기술에 따른 TCP-Vegas 버전에서의 전송율 변화를 나타내는 도면이다. 기존의 TCP는 전송량을 조절하는 단계를 크게 두 단계로 구분한다.

첫번째 단계는 사용 가능한 대역폭을 찾는 단계로 느린시작(slow-start) 단계라고 부른다(도면 2a 참조). 사용 가능한 대역폭을 찾기까지 빠른 속도로 전송량을 늘리면서 네트워크의 상태를 확인한다. 전송량을 조금씩 늘리다가 미리 정의된 느린시작 임계값(ssthreshold)을 넘으면, 네트워크의 상태에 따라 전송량을 조절하는 단계로 접어든다. 이 단계를 혼잡회피(congestion avoidance) 단계라고 부르는데, 이 단계에서는 전송량을 천천히 증가시키게 된다.

느린시작(slow-start) 단계에서는 데이터 왕복 전송에 소요되는 시간값(Round Trip Time, 이하 RTT라 명한다)당 이전의 혼잡윈도우(Congestion window, 이하 cwnd라 명한다)를 두 배씩 증가시키게 된다. 한편, 혼잡회피 단계에서는 RTT당 1 패킷을 더 보내도록 cwnd를 증가 시킨다. 이를 나타내는 그림이 도 2a이다.

TCP-Vegas 버전에서는 RTT를 기준으로 최대 사용 가능한 대역폭과 현재 사용 중인 대역폭을 계산한다. 현재 사용 중인 대역폭이 최대 사용 가능한 대역폭 보다 훨씬 작은 경우는 전송률을 증가 시켜서 대역폭을 최대로 사용할 수 있도록 노력하고, 현재 사용 중인 대역폭이 최대 대역폭에 근접한 경우는 전송률을 줄여서 네트워크에 가능한 적은 패킷이 유지되도록 노력한다. 이를 나타내는 도면이 도 2b이다.

그러나, 이러한 종래의 전송 방식은 부득이하게 네트워크에 전송 가능한 양 이상의 데이터를 전송하여 패킷 손실을 발생해야만, 전송률을 조절하기 때문에 큰 부담이 뒤따르게 된다. 특히, 애드 혹 네트워크의 특성상, 전송 가능한 데이터의 양은 유선 네트워크와 달리 상황에 따라 크게 달라진다. 링크 계층에서는 충돌을 회피하기 위한 기법을 가지고 있으며 재전송 메커니즘도 가지고 있기 때문에, 패킷 손실이 발생할 때까지 전송률을 지속적으로 증가시키는 것은 전체 네트워크에 큰 부하를 주게 되며, TCP의 연속된 타임아웃을 유발한다. 이러한 동작은 결국 네트워크 자원 사용 효율을 크게 떨어뜨리는 결과를 초래하게 된다.

한편, 도 3은 종래 기술에 따른 TCP-Reno 버전과 TCP-Vegas 버전에서의 전송 실패 패킷의 개수를 비교한 도표이다. TCP-Vegas 버전과 같은 전송율 조절 방식은 ad hoc 네트워크에서 큰 장점을 가지지만, 주어진 통계자료를 통해 볼 때(도 3 참조), TCP-Vegas 역시 네트워크에서 버려지는 패킷의 수는 큰 차이가 없음을 확인할 수 있다.

이것은 결국 네트워크의 자원을 사용하였으나 전송에 실패하였으므로 네트워크 자원을 효율적으로 사용하고 있지 못하며 더욱이 다른 세션의 전송까지도 방해하는 문제를 낳게 된다. 애드 혹 네트워크에서 백그라운드 트래픽이 존재하는 경우 위에서 지적한 것과 같은 문제는 더욱 심각해 지게 된다. 또한, Vegas버전의 경우 ACK 패킷이 도착하는 시간까지를 RTT로 측정하므로, ACK 패킷의 발생은 기존의 유선 망에서의 수준을 유지하게 되므로 무선 네트워크의 자원사용(link 사용)을 필요이상으로 과도하게 하게 되는 문제가 있었다. 더욱이 이를 기반으로 전송량 조절을 하기 때문에 이것을 고치는 것은 합리적이지 않다.

발명이 이루고자 하는 기술적 과제

본 발명은 상기한 문제점을 해결하기 위해 안출된 것으로서, 본 발명의 목적은 종래의 TCP 전송량 조절 기법을 수정하여 TCP의 네트워크 사용량을 최대화하고, 충돌을 줄이는 기법을 수용함으로써 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법을 제공하는 데 있다.

발명의 구성 및 작용

상술한 목적을 달성하기 위한 본 발명의 일 실시예에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법은, 현재전송율(cur_bw)과 이전전송율(base_bw)의 차이값(Diff)을 계산하는 제 1단계; 상기 이전전송율의 표준편차(basebw_var)를 계산하여 상한임계값(upper threshold)과 하한임계값(lower threshold)을 설정하는 제 2단계; 상기 차이값과 상기 상하한 임계값의 크기를 비교한 결과에 따라, 상기 이전전송율을 상기 현재전송율로 업데이트하는 제 3단계를 포함하는 것을 특징으로 한다.

또한, 상기 이전전송율은 base_bw=cwnd/baseRTT 에 의해서 산출되고, 상기 현재전송율은 cur_bw=cwnd/curRTT 에 의해 산출되고, 상기 차이값은 Diff=cur_bw-base_bw에 의해 산출되며, 상기 cwnd는 혼잡 윈도우값, 상기 baseRTT는 이전 시간에 측정된 데이터 왕복 전송에 소요되는 시간값, 그리고, 상기 curRTT는 현재의 데이터 왕복 전송에 소요되는 시간값인 것을 특징으로 한다.

또한, 이전전송율의 표준편차는, $(basebwvar)=\sqrt{(1-\alpha)(basebwvar)_{before}^{2}+\alpha(Diff)^{2}}$ 에 의해 산출되고, 상기 α 는 가중치 평균의 파라미터이고, $(basebwvar)_{before}$는 이전시간값에서의 표준편차인 것을 특징으로 한다.

또한, 상기 제 3단계는, 차이값이 양수인 경우에는 상한임계값과 크기를 비교하고, 차이값이 음수인 경우는 하한임계값과 크기를 비교한 후 상기 이전전송율을 상기 현재전송율로 업데이트하는 단계인 것을 특징으로 한다.

또한, 상기 제 3단계는, 차이값이 양수이면서 상기 상한임계값보다 큰 경우에는 혼잡윈도우값을 증가시킨후 상기 이전전송율을 상기 현재전송율로 업데이트하고, 상기 차이값이 음수이면서 상기 하한임계값보다 큰 경우에는 혼잡윈도우값을 변동시키지 않고 상기 이전전송율을 상기 현재전송율로 업데이트하는 단계인 것을 특징으로 한다.

한편, 상술한 목적을 달성하기 위한 본 발명의 또 다른 일 실시예에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법은, 현재전송율(cur_bw)과 이전전송율(base_bw)의 차이값(Diff)을 계산하는 제 1단계; 상기 상한임계값(upper threshold)과 하한임계값(lower threshold)을 ±k*basebw_var(k는 1이상인 상수)로 설정하는 제 2단계; 상기 차이값(Diff)이 양수인 경우에 상기 차이값(Diff)이 상기 상한임계값(upper threshold)보다 큰 경우에는 혼잡 윈도우값(cwnd)값을 증가시킨후 상기 이전전송율(base_bw)을 상기 현재전송율(cur_bw)로 업데이트하고, 상기 차이값(Diff)이 음수인 경우에 상기 차이값(Diff)이 상기 하한임계값(lower threshold)보다 작은 경우에는 혼잡 윈도우값(cwnd)을 감소시킨 후 상기 이전전송율(base_bw)을 상기 현재전송율(cur_bw)로 업데이트하는 3단계를 포함하는 것을 특징으로 한다.

또한, 상기 제 3단계는, 상기 혼잡 윈도우값이 느린시작 임계값보다 작을 경우에는 상기 혼잡 윈도우값을 느린시작 임계값으로 조절하는 단계를 포함하는 것을 특징으로 한다.

이하, 본 발명에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법에 대해 도면을 참조하여 상세히 설명한다.

도 4는 현재 쓰루풋과 최대 쓰루풋를 비교한 그래프이다. 도 4에 나타나듯이, 애드 혹 네트워크에서는 시간 변화에 따라 최대 쓰루풋(40)이 크게 변화함을 알 수 있고, 현재 쓰루풋(42)과의 비교를 통해 네트워크 상태 변화를 예측함으로써, TCP송신측에선 전송율을 조절할 수 있는 것이다. 그러므로, 현재쓰루풋(42)을 조절하는 기법이 매우 중요함을 알 수 있다.

도 5는 본 발명의 일실시예에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법을 나타내는 흐름도이고, 도 6은 본 발명의 또 다른 일실시예에 따른 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법을 나타내는 흐름도이다.

TCP의 송신자는 데이터를 한 번에 한 패킷씩 보내지 않고 네트워크가 감당할 수 있을 양만큼을 전송하는데, 이것은 cwnd(congestion window)에 의하여 조절된다.

전송은 두 단계로 나누어져 실행된다. 전송 초기에는 느린시작(slow-start) 단계가 시작된다. 이 때에는 적절한 네트워크의 대역폭을 찾는 것을 목적으로 한다. RTT당 전송량이 미리 정의된 값(ssthresh) 이상이 되면, 네트워크의 상태에 따라 혼잡을 적게 발생시킬 수 있는 상태를 찾기 위하여 혼잡윈도우를 조절하는 혼잡 회피(congestion avoidance) 단계로 접어든다. 초기의 느린 시작 단계의 과정을 설명하는 것이 도 5이며, 혼잡 회피 단계의 과정을 설명하는 것이 도 6이다.

새로운 ACK를 받은 TCP 송신자는 현재전송율을 cur_bw=cwnd/curRTT로 결정하게 된다(curRTT는 현재의 데이터 왕복 전송에 소요되는 시간값이다). 최초의 패킷 교환 시기에는 사용 가능한 대역폭의 초기값인 이전전송율을 base_bw=cwnd/baseRTT (baseRTT는 이전 시간에 측정된 데이터 왕복 전송에 소요되는 시간값)로 계산하고, 초기의 base_bw값이 설정된 이후에는 현재전송율(cur_bw)은 지속적으로 업데이트되며, base_bw의 값은 base_bw의 업데이트가 필요한 경우에 한해, 현재전송율(cur_bw)값으로 base_bw의 값을 정한다. 그리고 난 후, 현재전송률(cur_bw)과 전송률 초기값(base_bw)의 차이인 Diff값을 계산한다(S500, S600). Diff값은 cur_bw에서 base_bw를 뺀 값으로 현재 네트워크의 상태를 파악하기 위하여 사용한다. Diff값은 매 패킷의 교환시에 이루어 지지만, 실제 RTT(Round Trip Time)값이 변화되지 않는 경우가 많으므로, 실제 변화되는 시점은 RTT가 업데이트되는 시점이다.

현재 네트워크의 상태에 대한 정보를 얻고 이 자료를 바탕으로 느린시작(slow-start)단계와 혼잡회피(congestion-avoidance)단계에서 각각 다음과 같은 행동을 취함으로써 네트워크의 부하를 줄이면서도 최대의 대역폭을 사용할 수 있는 구조를 취한다.

느린시작 단계와 혼잡회피 단계에서 상태 변화를 판단하기 위한 두 개의 임계값(threshold)을 설정하는데, 이는 base_bw의 표준편차를 계산하여 양의 표준편차값을 상한임계값(upper-threshold)으로 설정하고, 음의 표준편차값을 하한임계값(lower-threshold)으로 설정한다(S502, S602).

원래 표준편차를 구하는 일은 자료가 많이 요구되며, 또한 자료가 되는 현재전송율이 지속적으로 변화하는 경우는 매 패킷에 대하여 새로이 표준편차를 계산해야 한다. 즉, 평균값이 자료의 변화에 따라 변화하게 되고 이는 편차의 값에 영향을 미치게 되고 오버헤드가 너무 커지게 된다. 따라서, 표준편차를 실제로 구하기 보다는 $(basebwvar)=\sqrt{(1-\alpha)(basebwvar)_{before}^{2}+\alpha(Diff)^{2}}$ 와 같은 근사값을 취한다. 여기서 α 는 가중치 평균의 파라미터이고, $(basebwvar)_{before}$는 이전 시간값에서의 표준편차에 해당한다. 이는, 기존의 TCP-Vegas 버전에서의 α 와 β상황에 따라 조절해야 하는 기준이 모호한 값이라는 문제점을 극복하기 위함이다.

한편, 느린 시작 단계에 대한 과정을 먼저 살펴보면, 상하한 임계값을 설정하고 난 뒤에 Diff값이 양수인지 여부를 판단하게 된다(S504). 만약, 양수인 경우에는 상한 임계값과 크기를 비교한다(S506). Diff값이 상한임계값보다 큰 경우는 현재의 전송상태가 양호하다는 의미이므로 cwnd값을 증가시킨 후에(S508), 이전 전송율(base bw)을 현재전송율(cur bw)로 업데이트하게 된다(S510). Diff값이 상한임계값보다 크지 않을 경우에는 cwnd값을 변화시키지 않고 곧바로 이전 전송율(base bw)을 현재전송율(cur bw)로 업데이트하게 된다(S510).

그리고, Diff값이 양수가 아닌 경우에는 음수인지 여부를 판단하게 되고(S512), 음수인 경우에는 현재의 전송상태가 직전의 전송상태에 비해 양호하지 않다는 의미이므로 하한임계값과 크기를 비교한다(S514). Diff값이 하한임계값보다 큰 경우에는 cwnd값을 변화시키지 않고 곧바로 이전 전송율(base bw)을 현재전송율(cur bw)로 업데이트하게 된다(S510).

Diff값이 하한임계값보다 크지 않거나 Diff값이 영일때는 전송율을 업데이트하지 않고 전송을 종료할 것인지 여부를 판단한다(S516). 계속 전송을 할 필요가 있다면 일단 현재의 전송율을 유지하게 된다(S518). 그리고 난 후, 다시 처음의 S500 단계로 피드백해서 Diff값을 계산하게 된다(S500).

이전전송율인 base_bw를 현재전송율인 cur_bw로 업데이트하는 것이 중요한 이유는, 느린시작 단계는 적절한 base_bw를 찾는 과정이므로 느린 시작 단계에서 설정한 base_bw값이 혼잡회피 단계에서는 네트워크의 혼잡상태를 판단하는 기준이 되기 때문이다.

즉, 느린 시작 단계에서는 현재의 Diff값이 상하한 임계값 안의 범위에서 움직이면 cwnd값을 변화시킴이 없이 base_bw값을 업데이트해 주며, 특히 Diff가 양수이면서 상한임계값을 넘는 경우만 가용성(utilization)을 끌어올리기 위하여 cwnd값을 증가시킨 후 업데이트를 하는 것이다.

느린시작 단계에선 상한임계값을 + basebw_var, 하한임계값을 -basebw_var로 정하였지만, 혼잡회피 단계일 때는 k배를 한 값, 즉, ±k*basebw_var (k는 1이상인 상수)를 상하한 임계값으로 함으로써 전송량의 증감폭을 줄인다. 도 6의 실시예에선 k값이 1인 경우를 나타내고 있다.

다음으로, 혼잡 회피 단계의 경우를 살펴보면, 느린 시작 단계와 마찬가지로 Diff값을 계산한 후(S600), 상하한 임계값을 설정하게 된다(S602).

그리고, Diff값이 양수인지 여부를 판단한 후(S604), 양수인 경우는 상한 임계값과 크기를 비교한다(S606). 상한 임계값보다 큰 경우는 현재의 전송상태가 양호하다는 의미이므로 cwnd값을 증가시킨 후(S608) 이전 전송율(base_bw)을 현재전송율(cur_bw)로 업데이트하게 된다(S610). Diff값이 상한임계값보다 크지 않을 경우에는 cwnd값을 변화시키지 않고 곧바로 이전 전송율(base_bw)을 현재전송율(cur_bw)로 업데이트하게 된다(S610).

그리고, Diff값이 양수가 아닌 경우에는 음수인지 여부를 판단하게 되고(S612), 음수인 경우에는 현재의 전송상태가 직전의 전송상태에 비해 양호하지 않다는 의미이므로 하한임계값과 크기를 비교한다(S614). 만약, 하한임계값보다 작을 경우는 현재의 전송상태가 매우 불량한 상태이므로 cwnd값을 감소시킨 후(S616), 전송율을 업데이트하게 된다(S610). 그러나, 하한임계값보다 작지 않을 경우는 cwnd값을 변화시키지 않고 곧바로 업데이트를 한다(S610).

업데이트 단계를 거친 후에는 cwnd값이 느린시작 임계값(slow-start threshold) 아래로 떨어지는지 여부를 판단한다(S618). 또한, Diff값이 양수도 음수도 아닌 영이 될 경우도 cwnd값이 느린시작 임계값 아래로 떨어지는지 여부를 판단한다(S618).

만약, cwnd값이 느린시작 임계값보다 작은 경우는 패킷을 보낼 공간이 많이 남아있다는 의미이므로 cwnd 값을 느린시작 임계값에 맞춘 후(S620), 전송종료 여부를 판단하게 된다(S622). cwnd값이 느린시작 임계값보다 작지 않을 경우에도 전송종료 여부를 판단한 후(S622) 계속 전송을 해야 할 필요가 있는 경우는 다시 S600단계부터 시작하게 된다.

즉, 혼잡회피 단계에서는 상한 및 하한의 양 임계값을 초과하는 경우에만 cwnd 크기를 조절하여 전송량을 조절함으로써, 네트워크에 과부하를 주지 않도록 조절하는 것이다.

도 7a 와 7b는 본 발명의 일 실시예에 따른 쓰루풋 결과를 종래의 TCP-Reno 버전과 TCP-Vegas 버전에 따른 쓰루풋 결과와 비교한 도면이다. 도 7a는 백그라운드 트래픽이 없을 경우이고, 도 7b는 백그라운드 트래픽이 존재할 경우의 비교 결과이다. 본 발명에 따른 시뮬레이션 결과를 보면, 백그라운드 트래픽이 존재하든 존재하지 않든 종전의 TCP-Reno나 Vegas보다 쓰루풋이 더 높게 나타난다는 것을 알 수 있다.

도 8a 와 8b는 본 발명의 일 실시예에 따른 유실된 패킷의 수를 종래의 TCP-Reno 버전과 TCP-Vegas 버전에 따른 유실된 패킷의 수와 비교한 도면이다. 마찬가지로, 도 8a는 백그라운드 트래픽이 없을 경우이고, 도 8b는 백그라운드 트래픽이 존재할 경우의 비교 결과이다. 본 발명에 따른 시뮬레이션 결과를 보면, 백그라운드 트래픽이 존재하든 존재하지 않든, 종전의 TCP-Reno나 Vegas보다 유실된 패킷의 수가 더 적게 나타난다는 것을 알 수 있다.

본 발명에서는 전송율의 향상 또는 감소가 생겼다고 해서 바로 cwnd 값을 변화시키는 것이 아니라, 일반적으로 발생하는 편차값의 범위내에 Diff값이 존재한다면 cwnd값을 변화시킴이 없이 이전전송율을 현재전송율로 업데이트하면서 전송상태를 개선시켜 나가는 것이다.

발명의 효과

상기한 바와 같은 발명에 따르게 되면, 현재 상태에서의 최적의 전송량을 찾기 위한 방식을 제안하여 도 4와 같이 네트워크 상태에 따른 최적의 전송량을 추적할 수 있는 네트워크 전송량 예측을 하고, 이를 기반으로 전송율을 조절하여 애 드 혹 네트워크의 전송량을 증가시키며, 무선랜을 이용한 애드 혹 네트워크의 성능 저하현상을 막는 효과가 있다.

또한, 네트워크의 패킷 유실은 더 적어지고, 쓰루풋은 높은 상태를 유지함으로써 기존의 방법보다 네트워크의 자원 사용이 더 효율적이며, 특히, 백그라운드 트래픽이 증가할 때도 마찬가지의 효과를 나타낸다.

(57) 청구의 범위

청구항 1.

전송 제어 프로토콜(TCP)에서 전송율을 조절하는 방법에 있어서,

현재전송율(cur_bw)과 이전전송율(base_bw)의 차이값(Diff)을 계산하는 제 1단계;

상기 이전전송율의 표준편차(basebw_var)를 계산하여 상한임계값(upper threshold)과 하한임계값(lower threshold)을 설정하는 제 2단계;

상기 차이값과 상기 상하한 임계값의 크기를 비교한 결과에 따라, 상기 이전전송율을 상기 현재전송율로 업데이트하는 제 3단계를 포함하는 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

청구항 2.

제 1항에 있어서,

상기 이전전송율은 base_bw=cwnd/baseRTT 에 의해서 산출되고, 상기 현재전송율은 cur_bw=cwnd/curRTT에 의해 산출되고, 상기 차이값은 Diff=cur_bw-base_bw에 의해 산출되며, 상기 cwnd는 혼잡 윈도우값, 상기 baseRTT는 이전 시간에 측정된 데이터 왕복 전송에 소요되는 시간값, 그리고, 상기 curRTT는 현재의 데이터 왕복 전송에 소요되는 시간값인 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

청구항 3.

제 1항에서, 상기 이전전송율의 표준편차는, $(basebwvar)=\sqrt{(1-\alpha)(basebwvar)^2_{before}+\alpha(Diff)^2}$ 에 의해 산출되고, 상기 α 는 가중치 평균의 파라미터이고, $(basebwvar)_{before}$는 이전시간값에서의 표준편차인 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

청구항 4.

제 1항 내지 제 3항 중 어느 한 항에 있어서, 상기 제 3단계는,

차이값이 양수인 경우에는 상기 차이값을 상한임계값과 크기를 비교하고, 차이값이 음수인 경우는 상기 차이값을 하한임계값과 크기를 비교한 후 상기 이전전송율을 상기 현재전송율로 업데이트하는 단계인 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

청구항 5.

상기 제 4항에 있어서, 상기 제 3단계는,

상기 차이값이 양수이면서 상기 상한임계값보다 큰 경우에는 혼잡윈도우값을 증가시킨후 상기 이전전송율을 상기 현재전송율로 업데이트하고, 상기 차이값이 음수이면서 상기 하한임계값보다 큰 경우에는 혼잡윈도우값을 변동시키지 않고 상기 이전전송율을 상기 현재전송율로 업데이트하는 단계인 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

청구항 6.

제 1항에 있어서, 상기 제 2단계는,

상기 상한임계값 및 하한임계값을 ±k*basebw_var(k는 1이상인 상수)로 설정하는 단계이고, 상기 제 3단계는,

상기 차이값이 양수이면서 상기 상한 임계값보다 큰 경우에는 혼잡 윈도우값을 증가시킨 후 상기 이전전송율을 상기 현재전송율로 업데이트하고, 상기 차이값이 음수이면서 상기 하한 임계값보다 작은 경우에는 혼잡윈도우값을 감소시킨 후 상기 이전전송율을 상기 현재전송율로 업데이트하는 단계인 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

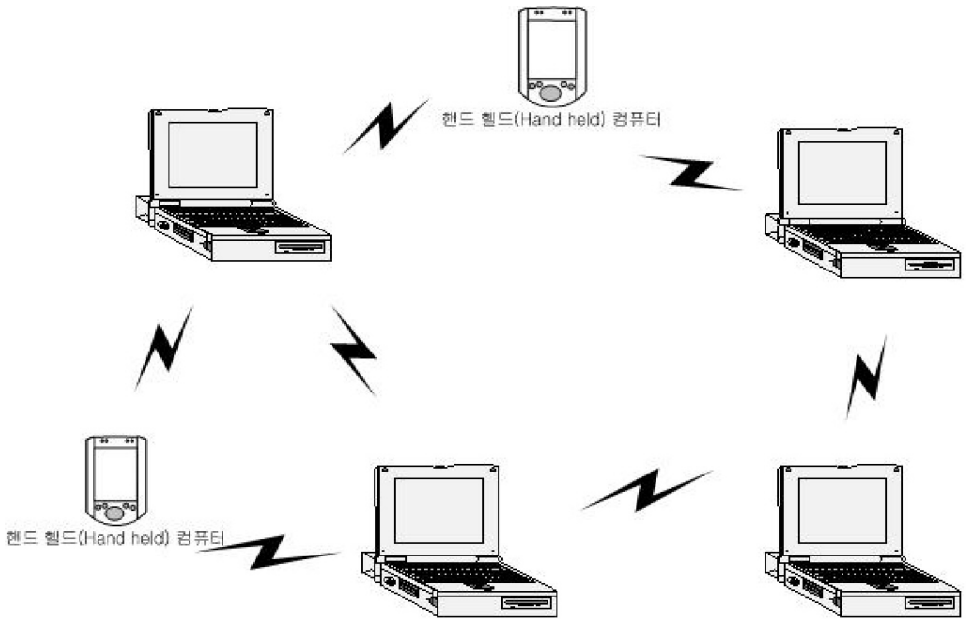
청구항 7.

제 6항에 있어서, 상기 제 3단계는,

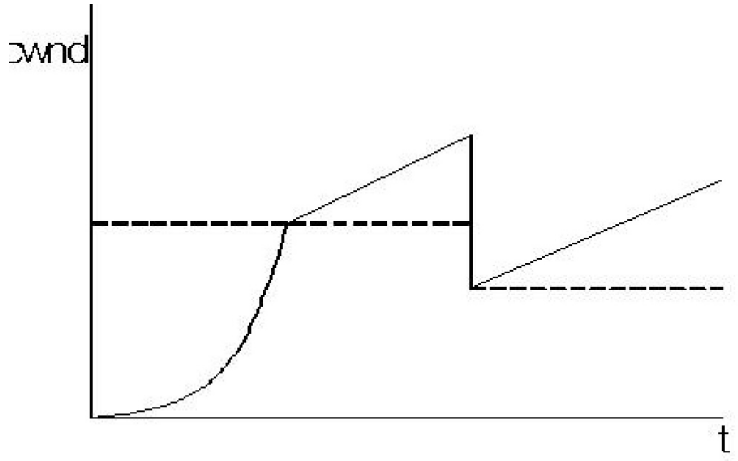
상기 혼잡 윈도우값이 느린시작 임계값보다 작을 경우에는 상기 혼잡 윈도우값을 느린시작 임계값으로 조절하는 단계를 포함하는 것을 특징으로 하는 모바일 애드 혹 네트워크 환경에서 최적의 전송율을 찾기 위한 조절 방법.

도면

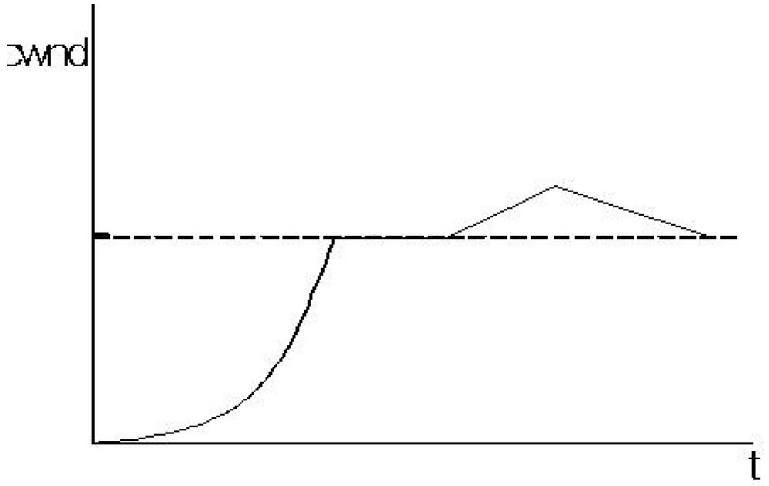
도면1



도면2a



도면2b

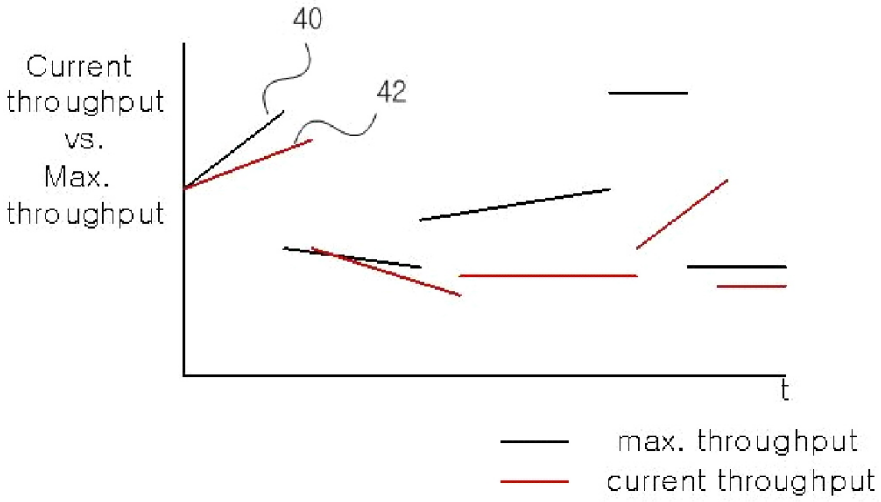


도면3

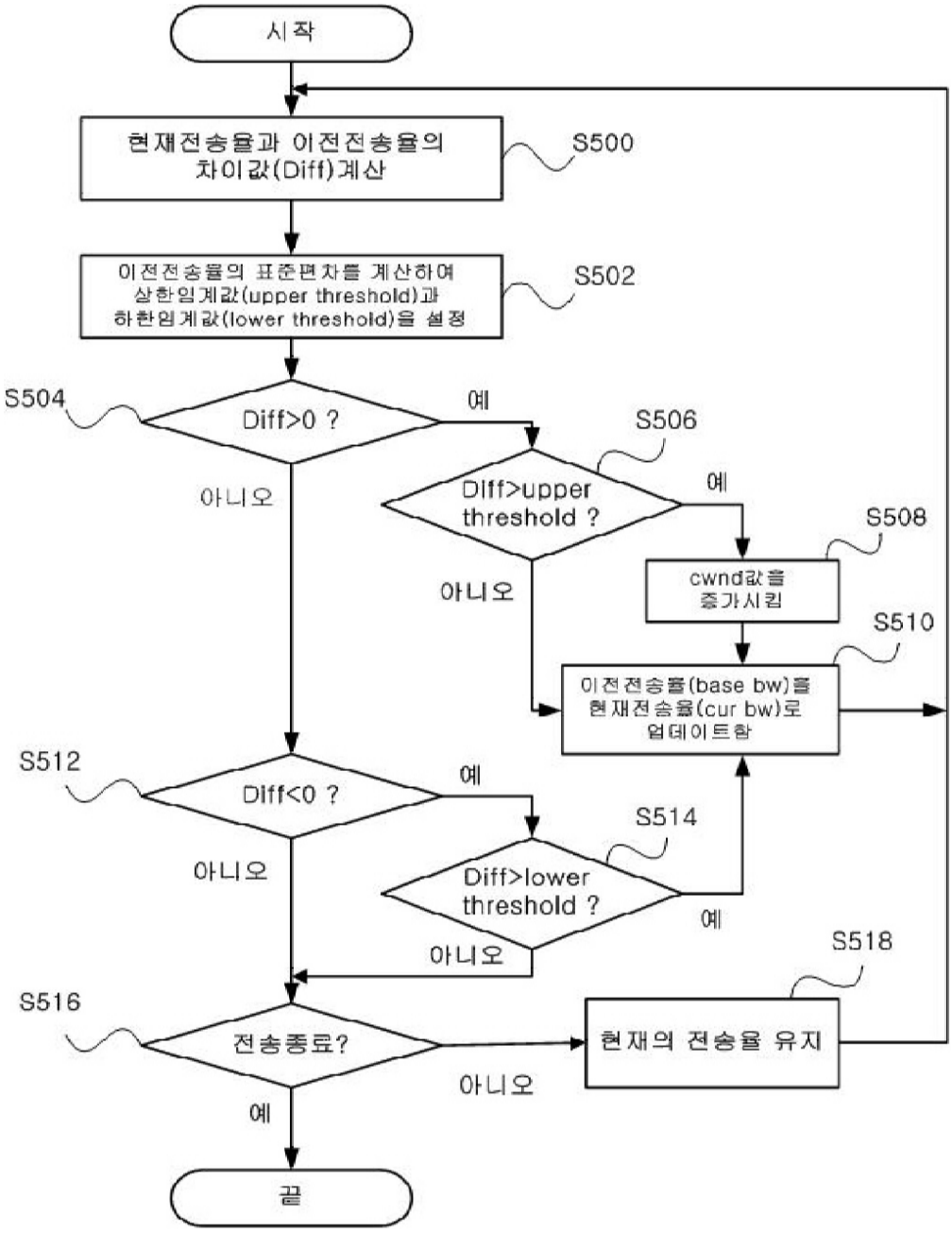
흡 수	1	2	3	4	5	6	7	8	9
Reno	18	215	10933	8926	9645	11029	13746	13609	15278
Vegas	1	125	10008	11161	12295	13745	13284	12934	14024

트레이스 파일 분석을 통해 얻은 전송 실패 패킷의 개수

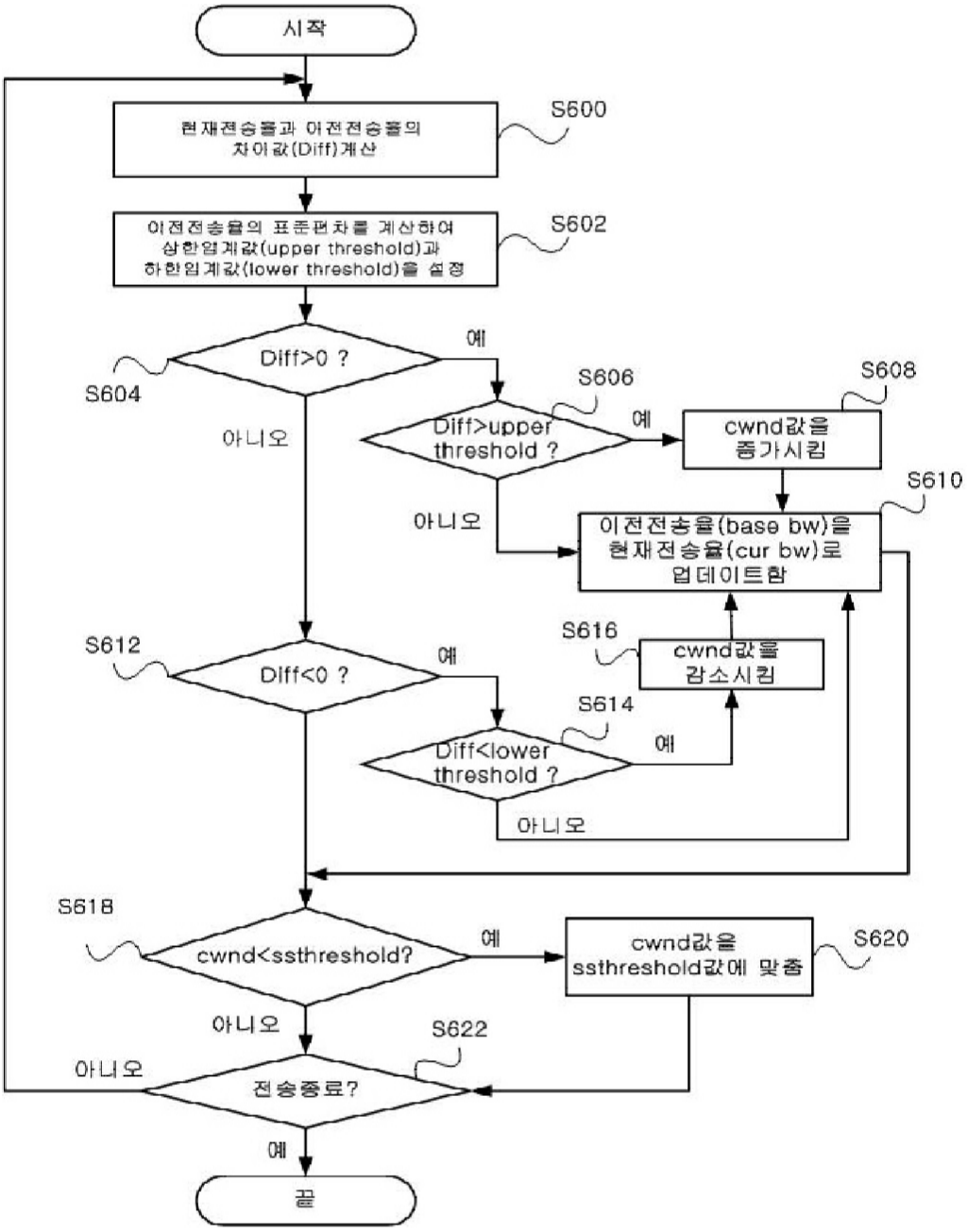
도면4


40
42
Current throughput vs. Max. throughput
t
max. throughput
current throughput

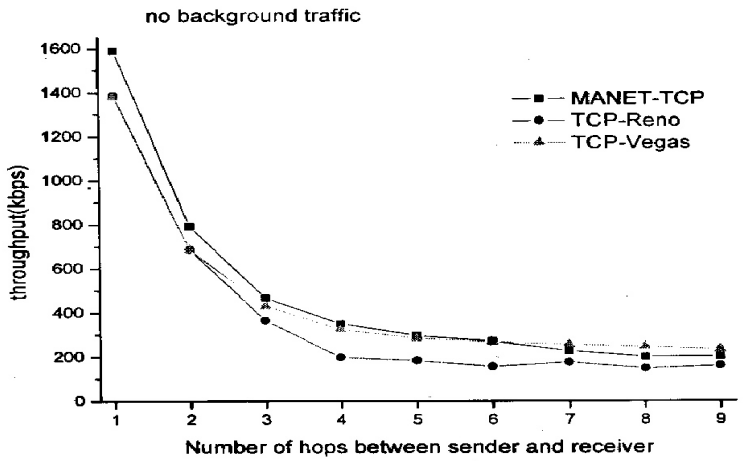
도면5


시작
현재전송율과 이전전송율의 차이값(Diff)계산
S500
이전전송율의 표준편차를 계산하여 상한임계값(upper threshold)과 하한임계값(lower threshold)을 설정
S502
S504
Diff>0 ?
예
아니오
S506
Diff>upper threshold ?
예
아니오
S508
cwnd값을 증가시킴
S510
이전전송율(base bw)를 현재전송율(cur bw)로 업데이트함
S512
Diff<0 ?
예
아니오
S514
Diff>lower threshold ?
예
아니오
S516
전송종료?
아니오
예
S518
현재의 전송율 유지
끝

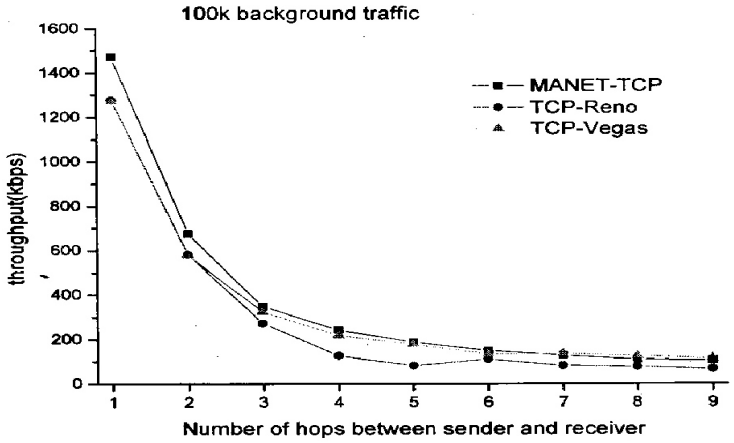
도면6


시작
현재전송률과 이전전송률의 차이값(Diff)계산
S600
이전전송률의 표준편차를 계산하여 상한임계값(upper threshold)과 하한임계값(lower threshold)을 설정
S602
Diff>0 ?
S604
예
아니오
Diff>upper threshold ?
S606
예
cwnd값을 증가시킴
S608
아니오
이전전송율(base bw)을 현재전송율(cur bw)로 업데이트함
S610
Diff<0 ?
S612
예
cwnd값을 감소시킴
S616
Diff<lower threshold ?
S614
예
아니오
cwnd<ssthreshold?
S618
예
cwnd값을 ssthreshold값에 맞춤
S620
아니오
전송종료?
S622
아니오
예
끝

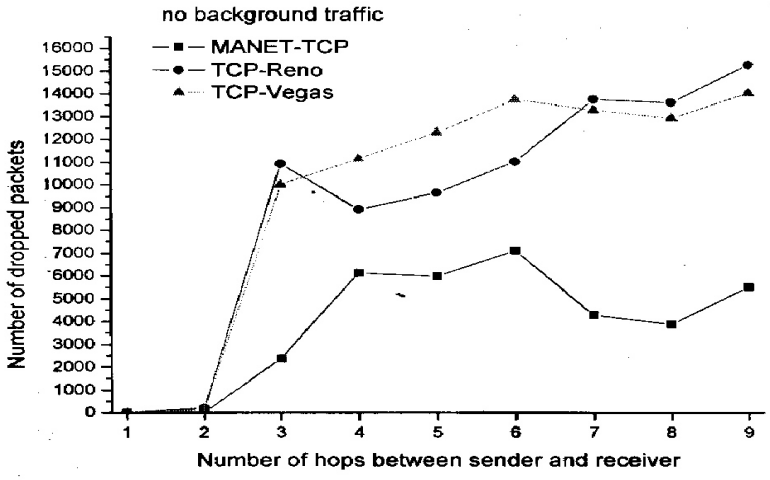
도면7a



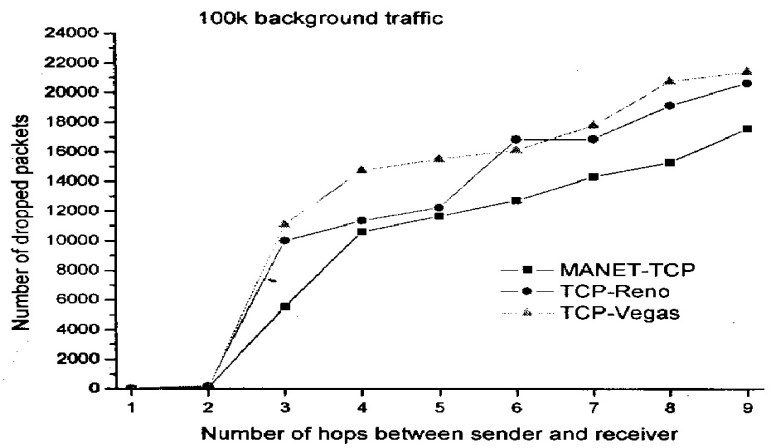
도면7b



도면8a



도면8b


100k background traffic
Number of dropped packets
24000
22000
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
1
2
3
4
5
6
7
8
9
Number of hops between sender and receiver
MANET-TCP
TCP-Reno
TCP-Vegas