(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0109662
(43) 공개일자 2008년12월17일

(51) Int. Cl.
H04B 7/26 (2006.01) *H04L 12/28* (2006.01)
(21) 출원번호 10-2008-0055304
(22) 출원일자 2008년06월12일
심사청구일자 2008년06월12일
(30) 우선권주장
1020070057396 2007년06월12일 대한민국(KR)

(71) 출원인
경희대학교 산학협력단
경기도 용인시 기흥구 서천동 1 경희대학교 국제캠퍼스내
제주대학교 산학협력단
제주특별자치도 제주시 아라1동 1번지 제주대학교
(72) 발명자
허의남
서울특별시 강동구 명일동 354 형인허브빌 아파트 101-1003
이가원
경기도 용인시 기흥구 서천동 294-6 101호
(74) 대리인
특허법인 신성

전체 청구항 수 : 총 12 항

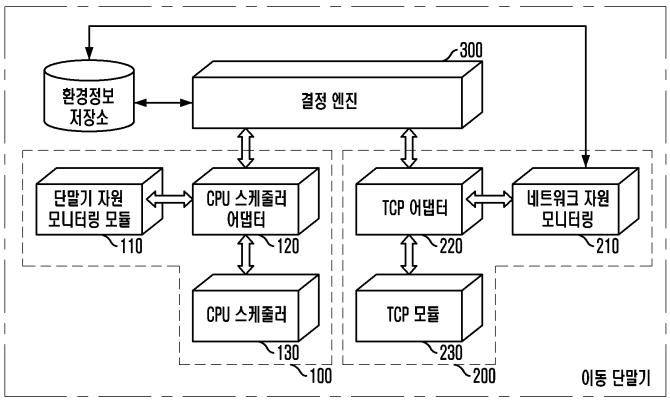
(54) 이동 단말기의 자원 관리 장치 및 그 방법

(57) 요 약

본 발명은 이동 단말기의 자원 관리 장치 및 그 방법에 관한 것으로, 수직적 핸드오프 환경에서 이동 단말기의 자체 자원 및 네트워크 자원을 관리함으로써, 이동통신 사용자에게 항상 서비스 품질을 유지하면서도 넓은 의미의 이동성을 보장하기 위하여 같은 종류의 망뿐만 아니라 이기종의 망으로 이동시 끊김 없이 이기종 망에 빠르게 적응하여 서비스를 제공하기 위한, 이동 단말기의 자원 관리 장치 및 그 방법을 제공하고자 한다.

이를 위하여, 본 발명은 이동 단말기의 자원 관리 장치에 있어서, 패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량을 산출하기 위한 단말기 자원관리 수단; 네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP(Transmission Control Protocol) 소켓의 버퍼 크기를 산출하기 위한 네트워크 자원관리 수단; 및 상기 단말기 자원관리 수단에서 산출한 CPU 할당량과 상기 네트워크 자원관리 수단에서 산출한 TCP 소켓의 버퍼 크기를 할당하기 위한 자원할당 수단을 포함한다.

대 표 도 - 도1



특허청구의 범위

청구항 1

이동 단말기의 자원 관리 장치에 있어서,

패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량을 산출하기 위한 단말기 자원관리 수단;

네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP(Transmission Control Protocol) 소켓의 버퍼 크기를 산출하기 위한 네트워크 자원관리 수단; 및

상기 단말기 자원관리 수단에서 산출한 CPU 할당량과 상기 네트워크 자원관리 수단에서 산출한 TCP 소켓의 버퍼 크기를 할당하기 위한 자원할당 수단

을 포함하는 이동 단말기의 자원 관리 장치.

청구항 2

제 1 항에 있어서,

상기 단말기 자원관리 수단은,

패킷 도착률에 따른 메모리 사용량 및 CPU 처리량을 수집하기 위한 단말기 자원 모니터링 모듈;

상기 단말기 자원 모니터링 모듈에서 수집한 메모리 사용량 및 CPU 처리량 정보를 CPU 스케줄러로 전달하고, 상기 CPU 스케줄러에서 산출한 CPU 할당량을 결정 엔진으로 전달하기 위한 CPU 스케줄러 어뎁터; 및

상기 CPU 스케줄러 어뎁터로부터 전달받은 메모리 사용량 및 CPU 처리량을 이용하여 CPU 할당량을 산출하기 위한 상기 CPU 스케줄러

를 포함하는 이동 단말기의 자원 관리 장치.

청구항 3

제 1 항에 있어서,

상기 네트워크 자원관리 수단은,

네트워크 유효 대역폭 및 패킷 지연시간을 수집하기 위한 네트워크 자원 모니터링 모듈;

상기 네트워크 자원 모니터링 모듈에서 수집한 네트워크 유효 대역폭 정보 및 패킷 지연시간 정보를 TCP 모듈로 전달하고, 상기 TCP 모듈에서 산출한 TCP 소켓의 버퍼 크기를 결정엔진으로 전달하기 위한 TCP 어뎁터; 및

상기 네트워크 자원 모니터링 모듈에서 수집한 네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP 소켓의 버퍼 크기를 산출하기 위한 상기 TCP 모듈

을 포함하는 이동 단말기의 자원 관리 장치.

청구항 4

제 2 항에 있어서,

상기 CPU 스케줄러는,

하기의 [수학식 A]를 이용하여 CPU 할당량(CPU_{alloc})을 산출하는 것을 특징으로 하는 이동 단말기의 자원 관리 장치.

[수학식 A]

$$CPU_{alloc} = CPU_{alloc} \times p(t)$$

$$p(t) = \frac{\lambda_{current}}{\lambda_{previous}}$$

여기서, , $\lambda_{current}$ 는 이전의 패킷 도착률, $\lambda_{previous}$ 는 현재의 패킷 도착률을 의미한다.

청구항 5

제 3 항에 있어서,

상기 TCP 모듈은,

하기의 [수학식 B]를 이용하여 TCP 소켓의 버퍼 크기를 산출하는 것을 특징으로 하는 이동 단말기의 자원 관리 장치.

[수학식 B]

$$S = A_{BWD} \times T$$

여기서, A_{BWD} 는 네트워크의 유효 대역폭, T는 패킷 지연시간을 의미한다.

청구항 6

제 5 항에 있어서,

상기 네트워크의 유효 대역폭은,

하기의 [수학식 C]와 같이 표현되는 것을 특징으로 하는 이동 단말기의 자원 관리 장치.

[수학식 C]

$$A_{BWD} = \frac{t_d}{t_r - t_s}$$

여기서, t_d는 전송된 데이터의 크기, t_r은 'Acknowledgement'를 받았을 때의 시간, t_s는 새로운 데이터 전송이 시작된 시간을 나타낸다.

청구항 7

이동 단말기에서의 자원 관리 방법에 있어서,

패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량을 산출하는 CPU 할당량 산출단계;

네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP(Transmission Control Protocol) 소켓의 버퍼크기를 산출하는 TCP 소켓의 버퍼크기 산출단계; 및

상기 산출한 CPU 할당량과 TCP 소켓의 버퍼 크기를 할당하는 자원 할당단계

를 포함하는 이동 단말기에서의 자원 관리 방법.

청구항 8

제 7 항에 있어서,

상기 CPU 할당량 산출단계는,

패킷 도착률에 따른 메모리 사용량 및 CPU 처리량을 수집하는 단계;

상기 수집한 메모리 사용량 및 CPU 처리량을 이용하여 CPU 할당량을 산출하는 단계; 및

상기 산출한 CPU 할당량을 결정엔진으로 전달하는 단계

를 포함하는 이동 단말기에서의 자원 관리 방법.

청구항 9

제 7 항에 있어서,

상기 TCP 소켓의 버퍼 크기 산출단계는,

네트워크 유효 대역폭 및 패킷 지연시간을 수집하는 단계;

상기 수집한 네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP 소켓의 버퍼 크기를 산출하는 단계; 및

상기 산출한 TCP 소켓의 버퍼 크기를 결정엔진으로 전달하는 단계

를 포함하는 이동 단말기에서의 자원 관리 방법.

청구항 10

제 8 항에 있어서,

상기 CPU 할당량 산출단계는,

하기의 [수학식 D]를 이용하여 CPU 할당량(CPU_{alloc})을 산출하는 것을 특징으로 하는 이동 단말기에서의 자원 관리 방법.

[수학식 D]

$CPU_{alloc} = CPU_{alloc} \times p(t)$

여기서, $p(t) = \frac{\lambda_{current}}{\lambda_{previous}}$, $\lambda_{current}$ 는 이전 패킷 도착률, $\lambda_{previous}$ 는현재 패킷 도착률을 의미한다.

청구항 11

제 9 항에 있어서,

상기 TCP 소켓의 버퍼 크기 산출단계는,

하기의 [수학식 E]를 이용하여 TCP 소켓의 버퍼 크기를 산출하는 것을 특징으로 하는 자원 관리 장치에서의 자원 관리 방법.

[수학식 E]

$$S = A_{BWD} \times T$$

여기서, A_{BWD} 는 네트워크의 유효 대역폭, T는 패킷 지연시간을 의미한다.

청구항 12

제 11 항에 있어서,

상기 네트워크의 유효 대역폭은,

하기의 [수학식 F]와 같이 표현되는 것을 특징으로 하는 자원 관리 장치에서의 자원 관리 방법.

[수학식 F]

$$A_{BWD} = \frac{t_d}{t_r - t_s}$$

여기서, t_d는 전송된 데이터의 크기, t_r은 'Acknowledgement'를 받았을 때의 시간, t_s는 새로운 데이터 전송이 시작된 시간을 나타낸다.

명 세 서

발명의 상세한 설명

기 술 분 야

<1> 본 발명은 이동 단말기의 자원 관리 장치 및 그 방법에 관한 것으로, 더욱 상세하게는 수직적 핸드오프 환경에서 이동 단말기의 자체 자원 및 네트워크 자원을 관리함으로써, 이동통신 사용자에게 항상 서비스 품질을 유지하면서도 넓은 의미의 이동성을 보장하기 위하여 같은 종류의 망뿐만 아니라 이기종의 망으로 이동시 끊김 없이 이기종 망에 빠르게 적응하여 서비스를 제공하기 위한, 이동 단말기의 자원 관리 장치 및 그 방법에 관한 것이다.

배 경 기 술

<2> 이동통신 가입자와 단말기의 수가 급격히 증가함에 따라 무선통신/이동통신 분야가 급격히 성장하고 있다. 이로 인하여, 전자 상거래와 같은 인터넷 프로토콜(IP)에 근거한 서비스 및 다양한 응용(예를 들어, World Wide Web, E-Mail 등)이 활성화된지 오래다.

<3> 이렇게 다양한 무선 시스템이 등장함에 따라 차세대 네트워크에서는 이동 단말이 서로 다른 종류의 네트워크망을 이동하며 서비스를 받을 수 있도록 해야 한다. 이러한 서로 다른 네트워크의 결합은 광대역의 데이터 접근 문제를 감소시키고, 새로운 설비와 이용을 위한 개발 비용의 절감을 도모할 수 있다는 점에서 더욱 현실화될 것으로 기대된다.

<4> 서로 다른 무선 접속 기술 중에, WLAN(Wireless Local Area Network)과 셀룰러 시스템은 지난 2년 사이 널리 보급되기 시작한 기술이다. 이러한 WLAN은 높은 데이터 전송률(11 Mbps for 802.11b, 54 Mbps for 802.11a and 802.11g)과 그에 비해 상대적으로 싼 비용으로 인하여, 주로 공항, 까페, 병원, 학교 등과 같이 무선 인터넷 수요가 많은 공공장소에서 주로 서비스되고 있다.

<5> 최근에는 WLAN 지역에서 VoIP(Voice over Internet Protocol) 및 음성통신도 가능해졌다. 그렇지만 아직 제한된 범위에서 서비스를 제공하고, 많은 방해 요소들로 인한 서비스 품질관련 문제들이 WLAN의 주요 결점으로 부각되고 있다.

<6> 원거리 통신 영역에서 셀룰러 시스템은 이동통신 분야에 있어 가장 대중적인 무선접속 기술이라 할 수 있다. 셀룰러 시스템은 1G, 2G(GSM, DAMPS), 2.5G(GPRS, EDGE), 이어 3G(UMTS) 같이 세대별로 발전해 가고 있다. 1992년에, 국제 전기 통신 연합(ITU)은 3G의 기본적인 특성을 정의하여 IMT-2000(International Mobile Telecommunication for year 2000)을 발표했다.

<7> 이러한 3G의 높은 개발 비용은 전 세계적으로 상용화되는 것을 지체시키는 요소로 작용하고 있다. 반면, 셀룰러 시스템은 낮은 데이터 전송률과 상대적인 자원비용의 제한에 반해 넓은 범위에 서비스가 가능하며 사람들에게 널리 알려진 음성 서비스라는 점에 있어서 유리하다고 할 수 있다.

<8> 이렇게 무선통신과 이동통신의 급속한 성장에 따라, 차세대 네트워크의 형태는 두 가지 통신망이 연동 및 통합되어 서비스가 이루어질 것으로 기대된다. 이동통신의 특성을 살려, 이동성을 더욱 보강하는 등 해당 플랫폼을 제공하기 위하여 WLAN과 셀룰러 시스템의 특성을 결합해야 할 것이다. 이러한 방식은 오버레이 형태로 서로 중첩되는 구조와 같은 방식으로 해결될 수 있을 것이다.

<9> 즉, 이동통신 사용자에게 항상 좋은 품질의 서비스를 제공하면서도 더 넓은 의미의 이동성을 보장하기 위하여 같은 종류의 망뿐만 아니라, 이기종의 망으로 이동시에도 전환이 가능해야 할 것이다. 이것은 이동통신 사용자를 다른 네트워크로 전환 가능하게 해주는 수직적 핸드오프라고 불리는 메커니즘에 의해 달성될 수 있다.

<10> 어플리케이션에서는 노드의 이동 성질로 인한 패킷 지연시간(RTT), 패킷 손실률과 네트워크 매개변수, 가변적인 채널 상태 등으로 서비스 품질의 하락이 나타나게 된다. 이는 전체 네트워크의 효율이 떨어지는 문제로 나타난다. 핸드오프 결정은 이동통신 사용자에게 더 나은 QoS(Quality of Service)를 제공하는 네트워크로 전환하기 위하여 사용된다. 이러한 다른 네트워크로의 전환에 대한 결정은 일정한 품질의 서비스를 사용자에게 꾸준히 제

공하기 위하여 사용자 프로파일, 기기 프로파일 및 어플리케이션 프로파일 등의 정보를 고려해야 한다.

<11> 사용자가 이동하거나, 현재 네트워크 연결 상태가 불량한 경우, 또는 어플리케이션의 서비스 품질이 하락하는 등의 현상이 나타나면, 이동 단말기는 다른 네트워크로 이동해야 할지 현재 네트워크에 남아야 할지를 결정하여 핸드오프를 수행한다.

<12> 현재, 새로운 네트워크 양상을 차용하여 응용 계층에 적용하고자 하는 다양한 수직적 핸드오프 관리 구조가 제안되었지만, 대부분 이동 단말기의 자원 자체에 대한 영향은 고려하고 있지 않다.

<13> 즉, 한 네트워크 환경에서 성격이 다른 네트워크 환경으로의 수직적 핸드오프가 일어날 경우, 높은 데이터 전송률의 네트워크에서 낮은 데이터 전송률의 네트워크로 이동할 수도 있으며, 그 반대로 낮은 데이터 전송률의 네트워크에서 높은 데이터 전송률의 네트워크로 이동할 수도 있다.

<14> 이때, 이동 단말기가 높은 데이터 전송률의 네트워크에서 낮은 데이터 전송률의 네트워크로 이동하는 경우, 자원 분배에 대한 재설정을 해주지 않기 때문에 높은 데이터 전송률에서의 메모리나 CPU 등에 대한 자원 설정이 그대로 적용되어, 상대적으로 데이터 전송률이 낮음에도 불구하고 높은 자원 설정으로 인하여 자원의 낭비를 초래하는 문제점이 있다.

<15> 또한, 이동 단말기가 낮은 데이터 전송률의 네트워크에서 높은 데이터 전송률의 네트워크로 이동하는 경우, 자원 분배에 대한 재설정을 해주지 않기 때문에 낮은 데이터 전송률에서의 메모리나 CPU 등에 대한 자원 설정이 그대로 적용되어, 여유 대역폭이 존재함에도 불구하고 낮은 자원 설정으로 인하여 대역폭의 낭비를 초래하는 문제점이 있다.

발명의 내용

해결 하고자하는 과제

<16> 이러한 문제점을 해결하고자 하는 것이 본 발명의 과제이다.

<17> 따라서, 본 발명은 수직적 핸드오프 환경에서 이동 단말기의 자체 자원 및 네트워크 자원을 관리함으로써, 이동통신 사용자에게 항상 서비스 품질을 유지하면서도 넓은 의미의 이동성을 보장하기 위하여 같은 종류의 망뿐만 아니라 이기종의 망으로 이동시 끊김 없이 이기종 망에 빠르게 적응하여 서비스를 제공하기 위한, 이동 단말기의 자원 관리 장치 및 그 방법을 제공하는데 그 목적이 있다.

<18> 본 발명의 목적들은 이상에서 언급한 목적으로 제한되지 않으며, 언급되지 않은 본 발명의 다른 목적 및 장점들은 하기의 설명에 의해서 이해될 수 있으며, 본 발명의 실시예에 의해 보다 분명하게 알게 될 것이다. 또한, 본 발명의 목적 및 장점들은 특허 청구 범위에 나타낸 수단 및 그 조합에 의해 실현될 수 있음을 쉽게 알 수 있을 것이다.

과제 해결수단

<19> 상기 목적을 달성하기 위한 본 발명의 장치는, 이동 단말기의 자원 관리 장치에 있어서, 패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량을 산출하기 위한 단말기 자원관리 수단; 네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP(Transmission Control Protocol) 소켓의 버퍼 크기를 산출하기 위한 네트워크 자원관리 수단; 및 상기 단말기 자원관리 수단에서 산출한 CPU 할당량과 상기 네트워크 자원관리 수단에서 산출한 TCP 소켓의 버퍼 크기를 할당하기 위한 자원할당 수단을 포함한다.

<20> 또한, 상기 목적을 달성하기 위한 본 발명의 방법은, 이동 단말기에서의 자원 관리 방법에 있어서, 패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량을 산출하는 CPU 할당량 산출단계; 네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP(Transmission Control Protocol) 소켓의 버퍼크기를 산출하는 TCP 소켓의 버퍼크기 산출단계; 및 상기 산출한 CPU 할당량과 TCP 소켓의 버퍼 크기를 할당하는 자원 할당단계를 포함한다.

<21> 또한, 본 발명은 단순히 신호의 세기나 연결 상태에 근거하여 핸드오프를 결정하는 것이 아니라 다른 네트워크로의 핸드오프를 위한 환경적인 지식까지 고려하는 핸드오프 알고리즘 정책을 적용한다.

<22> 또한, 본 발명은 다른 네트워크로의 수직적 핸드오프시 적절한 TCP 버퍼 사이즈 계산을 통해 시스템 리소스의 효율적인 확보 및 응용 프로그램의 실행을 유지하고 효과적인 자원의 적용을 포함한다.

효 과

<23> 상기와 같은 본 발명은, 수직적 핸드오프 환경에서 이동 단말기의 자체 자원 및 네트워크 자원을 관리함으로써, 이동통신 사용자에게 항상 서비스 품질을 유지하면서도 넓은 의미의 이동성을 보장하기 위하여 같은 종류의 망 뿐만 아니라 이기종의 망으로 이동시 끊김 없이 이기종 망에 빠르게 적응하여 서비스를 제공할 수 있는 효과가 있다.

<24> 또한, 본 발명은 단말기 자원 관리 기술을 수직적 핸드오프 상황에 적용함으로써, 사용자에게 서비스 중단을 최소화하고 단말기의 자원 및 네트워크 대역에 대한 낭비를 방지하여 에너지 효율을 높일 수 있는 효과가 있다.

발명의 실시를 위한 구체적인 내용

<25> 상술한 목적, 특징 및 장점은 첨부된 도면을 참조하여 상세하게 후술되어 있는 상세한 설명을 통하여 보다 명확해 질 것이며, 그에 따라 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 본 발명의 기술적 사상을 용이하게 실시할 수 있을 것이다. 또한, 본 발명을 설명함에 있어서 본 발명과 관련된 공지 기술에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에 그 상세한 설명을 생략하기로 한다. 이하, 첨부된 도면을 참조하여 본 발명에 따른 바람직한 실시예를 상세히 설명하기로 한다.

<26> 도 1 은 본 발명에 따른 이동 단말기의 자원 관리 장치에 대한 일실시예 구성도이다.

<27> 도 1에 도시된 바와 같이, 본 발명에 따른 이동 단말기의 자원 관리 장치는, 수직적 핸드오프가 일어나는 동안에 동적으로 이동 단말기의 자원을 관리하는 구조로서, 미들웨어를 전제로 한 모듈 방식에 기초한다.

<28> 일반적으로 핸드오프 결정이 이루어질 때 네트워크 자원 모니터링 모듈이 수집하여 환경정보 저장소(Context Repository-CR)에 저장한 정보를 바탕으로 결정된다. 이때, 환경정보 저장소에 저장되는 정보는, 단말기의 여유 메모리, 어플리케이션, 사용자 정보, 현재 네트워크의 대역 정보 및 통신 연결 상태 등을 포함한다.

<29> 일반적인 핸드오프 절차에서, 결정엔진(Decision Engine-DE)은 서비스 품질 하락을 유발할 수 있는 채널 에러율 및 혼잡과 같은 네트워크 특성이 변하거나, 현재 네트워크의 접속 끊김 등의 네트워크 환경 변수들을 감지하여 핸드오프가 일어나야 하는지의 여부를 결정하는 역할을 한다. 이를 위해 네트워크 정보가 저장되어 있는 환경정보 저장소(Context Repository-CR)를 모니터링하여 핸드오프 여부를 결정한다.

<30> 즉, 결정엔진은 환경정보 저장소(CR)에 저장되어 있는 정보를 모니터링하여, 현재 연결되어 있는 네트워크보다 더 신호가 세고 더 가까우며 더 나은 연결과 통신을 지원할 수 있다고 판단하면 수직적 핸드오프를 결정한다.

<31> 본 발명은 이러한 수직적 핸드오프 결정 시, 수직적 핸드오프 여부를 결정하는 결정엔진(Decision Engine-DE)에서의 핸드오프 여부 결정 결과에 따라 TCP(Transmission Control Protocol) 버퍼의 크기 계산결과를 이동 단말기에 적용할지 여부를 결정하고자 한다.

<32> 이때, 단말기 자원 모니터링 모듈(SRM)은 수직적인 핸드오프가 일어나는 동안에도 이동 단말기의 성능을 유지하기 위하여 단말기의 메모리 사용량과 CPU(Central Processing Unit) 처리량 등의 자원을 모니터하고, 결정엔진에서 핸드오프를 결정하면 CPU 스케줄러는 단말기 자원 모니터링 모듈에서 수집한 자원정보를 CPU 스케줄러 어댑터로부터 전달받아 변화된(핸드오프 된) 네트워크에 어떻게 CPU 스케쥴링을 변경 적용할 것인지 CPU 할당량(할당 시간)을 계산한다.

<33> 마찬가지로, 네트워크 자원 모니터링 모듈(NRM)은 유효한 대역폭에 대한 정보를 모니터링 하고, 핸드오프가 결정됨에 따라 TCP 어댑터는 혼잡 윈도우의 크기를 계산하여 변화된 네트워크의 대역폭에 맞게 TCP 버퍼 크기를 어떻게 변경할 것인가를 결정하게 된다.

<34> 이와 같이 수직적 핸드오프 과정에서, 변화되는 네트워크로의 수직적 핸드오프가 결정되는 경우에 단말기의 설정 역시 변화된 네트워크에 맞게 변경시킨다.

<35> 이하, 도 1을 참조하여 본 발명에 따른 이동 단말기의 자원 관리 장치에 대해 좀 더 상세히 살펴보기로 한다.

<36> 도 1에 도시된 바와 같이, 본 발명에 따른 이동 단말기의 자원 관리 장치는, 수직적 핸드오프 네트워크에서 패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량(할당 시간)을 산출하기 위한 단말기 자원 관리부(100), 수직적 핸드오프 네트워크의 유효 대역폭 및 패킷 지연시간을 이용하여 TCP 소켓의 버퍼 크기를 산출하기 위한 네트워크 자원 관리부(200), 및 상기 단말기 자원 관리부(100)에서 산출한 CPU 할당량과 네트워크 자원 관리부(200)에서 산출한 TCP(Transmission Control Protocol) 소켓의 버퍼

크기를 할당하기 위한 결정엔진(300)을 포함한다.

<37> 여기서, 단말기 자원 관리부(100)는 패킷 도착률에 따른 메모리 사용량 및 CPU 처리량을 모니터링하기 위한 단말기 자원 모니터링 모듈(110), 상기 단말기 자원 모니터링 모듈에서 수집한 메모리 사용량 및 CPU 처리량 정보를 CPU 스케줄러(130)로 전달하고, 상기 CPU 스케줄러(130)로부터의 CPU 할당 시간을 결정 엔진으로 전달하기 위한 CPU 스케줄러 어뎁터(120), 및 상기 CPU 스케줄러 어뎁터(120)로부터 전달받은 메모리 사용량 및 CPU 처리량을 이용하여 CPU 할당량(할당 시간)을 산출하기 위한 CPU 스케줄러(130)를 포함한다.

<38> 또한, 네트워크 자원 관리부(200)는 네트워크 유효 대역폭(수용량) 및 패킷 지연시간(RTT)을 모니터링하기 위한 네트워크 자원 모니터링 모듈(210), 상기 네트워크 자원 모니터링 모듈(210)에서 수집한 네트워크 유효 대역폭 정보 및 패킷 지연시간 정보를 TCP 모듈(230)로 전달하고, 상기 TCP 모듈에서 산출한 TCP 소켓의 버퍼 크기를 결정엔진으로 전달하기 위한 TCP 어뎁터(220), 및 상기 네트워크 자원 모니터링 모듈(210)에서 수집한 네트워크 유효 대역폭 및 패킷 지연시간을 이용하여 TCP 소켓의 버퍼 크기를 산출하기 위한 TCP 모듈(230)을 포함한다.

<39> 또한, 결정엔진(DE)은 핸드오프가 실행되어야 하는지 여부를 판단하기 위해 네트워크 연결 상태 및 신호 세기 등의 환경 정보를 지속적으로 모니터링한다.

<40> 한편, 네트워크에서 응용 프로그램에 의해 요청되는 QoS는 네트워크 레벨 매개변수의 변화로 인해 품질이 하락할 수도 있다. 따라서, 서비스를 지속적으로 유지하기 위해 결정엔진(DE)은 네트워크 레벨 매개변수를 바꿀 수도 있다. 이때, 네트워크 품질에 대한 비중을 정하기 위해서는 사용자의 선호도를 반영하여 매개변수를 할당해야 한다.

<41> 각 매개변수 i의 대표를 P_i라 정의하고, 그에 관계되는 비중을 W_i라고 정의하면, 응용 프로그램의 서비스 품질은 하기의 [수학식 1]과 같이 나타낼 수 있다.

수학식 1

<42>
$$QoS_{est} = \sum w_i p_i$$

<43> 여기서, $QoS_{est} < QoS_{req}$ 이면, 사용자가 원하는 서비스의 질보다 현재의 서비스의 질이 나쁜 것이므로, 더 나은 서비스 품질을 가진 네트워크를 찾아 핸드오프를 수행한다. 이때, QoS는 네트워크의 실적을 대표하는 수치가 될 것이다. 만일, 이 수치가 응용프로그램에서 요구되는 QoS값보다 작을 경우에는 현재 네트워크는 좋은 품질의 서비스를 충분히 제공하지 못하는 것이라 할 수 있다.

<44> 또한, 단말기 자원 모니터링(System Resource Monitoring-SRM) 모듈(에이전트)(110)은, 이동 단말기의 자원 소비량에 대한 정보를 수집한다. 이때, 자원은 지속적으로 감시되는 응용 프로그램의 CPU와 메모리 이용에 대한 정보가 된다. CPU 이용 정보는 수직적 핸드오프시 어플리케이션의 CPU 사용량을 통제하기 위해서 필요하다.

<45> 여기서, 어플리케이션을 i로 표현하고 i*어플리케이션을 의미한다고 하면, CPU 이용은 CPU(A_i)와 같이 나타낼 수 있고, 메모리 이용은 Mem(A_i)과 같이 나타낼 수 있다. 이 정보는 리눅스 'proc' 테이블을 통해 얻을 수 있다.

<46> 또한, 단말기 자원 모니터링 모듈(110)은 핸드오프 프로세스가 일어나는 동안에 지속적인 감시를 통해 CPU 스케쥴러(130)의 스케줄링을 돕는다.

<47> 또한, 네트워크 자원 모니터링(Network Resource Monitoring - NRM) 에이전트(210)는, 프로토콜 스택의 모든 레벨에 있는 네트워크 특성을 감시한다. 지터, 이동 단말기의 이동에 따라 업데이트된 패킷 손실률과 지연율 및 신호 품질을 계산한다. 아울러, 네트워크 자원 모니터링 에이전트는 또한 네트워크의 가능한 대역이 얼마인지 계산한다. 이렇게 계산된 네트워크 유효 대역폭은 TCP 소켓의 버퍼 크기를 산출하는데 이용된다.

<48> 즉, 같은 대역폭을 사용하는 많은 이동통신 이용자들이 있기 때문에 경쟁에 따른 패킷 손실률을 최소한으로 줄이기 위하여, TCP 모듈(230)은 무선 네트워크의 유효한 대역폭에 따라 단말기의 TCP 소켓 버퍼 크기를 설정한다.

<49> 또한, 네트워크 자원 모니터링(NRM) 에이전트(210)는 프레임에서 MAC 레이어를 감시함으로써 유효한 대역폭을

계산한다.

<50> 이하, 도 2를 참조하여 네트워크의 유효 대역폭을 계산하는 과정에 대해 설명하기로 한다.

<51> 도 2에 도시된 바와 같이, t_d는 전송된 데이터의 크기, 다시 말해서 프레임의 수를 나타내고, t_r은 'Acknowledgement'를 받았을 때의 시간을 나타내며, t_s는 새로운 데이터 전송이 시작된 시간을 나타낸다. 이때, t_r-t_s는 전체 전송이 지속되는 간격을 보여준다. 이 간격은 데이터 전송을 위한 다른 후보 노드들과의 경쟁과 채널 혼잡 시간을 포함한다.

<52> 따라서, 네트워크 유효 대역폭은 하기의 [수학식 2]를 통해 계산할 수 있다.

수학식 2

$$A_{BWD} = \frac{t_d}{t_r - t_s}$$

<53>

<54> 도 3 은 본 발명에 따른 TCP 소켓의 버퍼 크기를 산출 과정에 대한 일실시예 설명도이다.

<55> 일반적으로, 이동 단말기가 수직적 핸드오프 시(다른 특징을 가진 네트워크로 이동), 어플리케이션에 의해 서비스가 중단된다. 그러므로, TCP 연결 역시 네트워크 환경의 변화 때문에 영향을 받게 된다.

<56> 따라서, 이동 단말기가 낮은 데이터 전송률의 네트워크에서 높은 데이터 전송률의 네트워크로 이동하면, TCP 연결은 이 변화에 대해서 새로운 유효 대역폭을 이용할 수 있도록 TCP 소켓의 버퍼 크기를 증가시키고, CPU 할당량도 증가시켜 변화된 자원의 사용에 능률적으로 대처해야 한다.

<57> 또한, 이동 단말기가 높은 데이터 전송률의 네트워크에서 낮은 데이터 전송률의 네트워크로 이동하면, TCP 소켓의 버퍼 크기를 감소시키고, CPU 할당량도 감소시켜 자원의 효율성을 높여야 한다.

<58> 이를 위해 도 3에 도시된 바와 같이, TCP 소켓의 버퍼 크기를 산출한다.

<59> 먼저, 네트워크의 수용량을 C bps, 패킷 지연시간(RTT)을 T sec, 현재 네트워크를 N_c, 새로운 네트워크를 N_n이라 하자.

<60> 그러면, 네트워크 대역폭의 완전한 이용을 위해서는 TCP 소켓의 버퍼 크기는 적어도 C×T로 구성되어야 한다. 이때, TCP 혼잡 윈도우 W_c를 사용하여 사용 가능한 네트워크를 최대한 사용할 수 있도록 한다.

<61> 따라서, 소켓 버퍼 사이즈(S)는 C×T로 표현된다.

<62> 여기서, 핸드오프 상황은 두 가지 유형으로 나눌 수 있는데, 하나는 이동 단말기가 높은 데이터 전송률의 네트워크에서 낮은 데이터 전송률의 네트워크로 이동하는 경우이고, 다른 하나는 이동 단말기가 낮은 데이터 전송률의 네트워크에서 높은 데이터 전송률의 네트워크로 이동하는 경우이다.

<63> 첫 번째 경우는 대역폭의 완전한 사용을 위해 윈도우를 더 많이 열어(W_c 증가시킴) TCP 소켓의 버퍼 크기를 증가시킨다.

<64> 두 번째 경우는 낮은 데이터 전송률의 네트워크에서 이전 TCP 소켓의 버퍼 크기를 유지할 경우 버퍼의 낭비를 초래하므로, 이를 방지하기 위해 W_c를 감소시켜야 한다. 이는 하기의 [수학식 3]과 같이 나타낼 수 있다.

수학식 3

$$S = A_{BWD} \times T$$

<65>

<66> 여기서, TCP 소켓의 버퍼 크기(S)는 네트워크 자원 모니터링 모듈(110)에 의해 수집된 네트워크의 유효 대역폭(A_{BWD})에 따라 조절된다.

<67> 결국, 수직적 핸드오프가 결정되면 변화된 네트워크의 대역폭을 구하고, 패킷 지연시간을 구해 TCP 혼잡 윈도우(버퍼)의 크기를 조절한다.

<68> 도 4 는 본 발명에 따른 CPU 스케줄러에서의 CPU 할당량 산출 과정에 대한 일실시예 설명도이다.

<69> 보통, 멀티미디어 응용 프로그램에서는 사용자에게 중단 없는 서비스 제공을 위해서 엄격한 서비스 품질 제공 기준이 요구된다. 그것은 일정 수준에 도달할 때까지 네트워크에서 패킷을 버퍼링함으로써 보증할 수 있다. 그 후에 버퍼링이 일정 수준에 도달하면, 어플리케이션이 재생률이라 불리는 어느 일정한 비율에서 패킷을 사용하여 보여주게 된다. 이동 단말기가 수직적 핸드오프를 할 때 어플리케이션 실행은 수직적 핸드오프가 이루어지고 난 후 데이터 전송률의 변화에 의해 영향을 받는다.

<70> 이때, 이동 단말기가 높은 데이터 전송률의 네트워크에서 낮은 데이터 전송률의 네트워크로 이동한다면(첫 번째 시나리오), 패킷은 더 낮은 비율로 도착할 것이고 낮은 데이터 전송률에서 높은 데이터 전송률 네트워크로 이동한 경우(두 번째 시나리오)에는 더 많은 패킷이 도착할 수 있는데도 낮은 데이터 전송률을 기준으로 패킷을 전송하므로 네트워크 대역 및 단말기 자원의 낭비를 초래한다.

<71> 첫 번째 시나리오의 경우, 응용 프로그램은 패킷의 도착률보다 더 빠른 비율로 데이터를 소모하게 된다. 이것은 응용 프로그램에서 필요로 하는 CPU 할당량보다 더 많이 할당된 경우이다.

<72> 두 번째 시나리오의 경우, 응용 프로그램의 패킷 소비율보다 패킷의 도착률이 더 높다. 따라서, 최종 사용자에게 좋은 품질의 서비스를 꾸준히 제공하기 위해서는 응용 프로그램에서 CPU 자원을 더 요구하게 된다.

<73> 그러므로, 응용 프로그램의 실행 성과를 시간 t의 간격 후에 채워지는 패킷 레벨을 감시함으로써 향상시킬 수 있다. 이 간격을 'epoch'라 부르기로 한다. 최적점 q는 동적으로 변화하는 CPU 할당 변수 'CPU_{alloc}'에 의해 얻어지는 패킷 도착률과 비례하는 버퍼에서 정의된다. 버퍼에 있는 최적점은 패킷의 데이터 전송률이 격렬히 변화하는 수직적 핸드오프 시나리오에서 멀티미디어 어플리케이션이 부드럽게 실행될 수 있도록 보장한다.

<74> 두 가지 매개변수를 버퍼 필 레벨과 CPU 적합 모듈로의 비례 변화로 정의한다. 여기서, 버퍼 필 레벨이란, 시간 t에서 버퍼가 차는 수준을 나타내는 q(t)로서, 하기의 [수학식 4]와 같이 나타낼 수 있다.

수학식 4

$$q(t) = \frac{Current\ buffer\ size\ attimet}{Total\ buffer\ size}$$

<75>

<76> 여기서, 현재의 'epoch'에서 패킷 도착률을 $\lambda_{current}$ 라 하고, 이전의 'epoch'에서 패킷 도착률을 $\lambda_{previous}$ 라 하면, 시간 t에 비례하는 비례변화 p(t)는 하기의 [수학식 5]와 같이 나타낼 수 있다.

수학식 5

$$p(t) = \frac{\lambda_{current}}{\lambda_{previous}}$$

<77>

<78> 상기 [수학식 4]와 [수학식 5]를 이용하여 CPU 스케쥴러의 CPU 할당량을 산출하는 과정은 도 4에 도시된 바와 같다.

<79> 즉, 먼저 패킷의 최초 도착률을 얻는다.

<80> 그리고, 패킷의 현재 도착률을 얻는다.

<81> 이후, 패킷의 최초 도착률과 패킷의 현재 도착률의 비례 변화를 산출한다.

<82> 이후, 현재 버퍼 레벨을 체크하여 그에 비례하게 CPU 자원을 할당한다.

<83> 도 5 는 본 발명에 따른 이동 단말기의 자원 관리 장치에서의 자원 관리 방법에 대한 일실시예 흐름도이다.

<84> 먼저, 단말기 자원 관리부(100)는 수직적 핸드오프 네트워크에서 패킷 도착률에 따른 메모리 사용량 및 CPU(Central Processing Unit) 처리량을 이용하여 CPU 할당량(할당 시간)을 산출한다(501).

<85> 그리고, 네트워크 자원 관리부(200)는 수직적 핸드오프 네트워크의 유효 대역폭 및 패킷 지연시간을 이용하여 TCP 소켓의 버퍼 크기를 산출한다(502).

<86> 이후, 결정엔진(300)은 상기 단말기 자원 관리부(100)에서 산출한 CPU 할당량과 네트워크 자원 관리부(200)에서 산출한 TCP(Transmission Control Protocol) 소켓의 버퍼 크기를 할당한다(503).

<87> 이하, 도 6 내지 도 9를 참조하여 수직적 핸드오프 환경에서 본 발명에 따른 이동 단말기의 자원 관리 장치에 대한 성능 평가 결과에 대해 살펴보기로 한다.

<88> 실험은 시뮬레이션 툴인 'ns-2'를 사용하여 이루어졌으며, 3G 셀룰러 네트워크에서 데이터 전송률은 144kbps, 패킷 지연시간(RTT)은 300msec이고, WLAN의 데이터 전송률은 2Mbps, 패킷 지연시간(RTT)은 100msec라고 설정했다.

<89> 이때, 모의 실험은 낮은 데이터 전송률의 네트워크에서 높은 데이터 전송률의 네트워크로 이동한 상황과 그 반대의 경우에서의 수직적 핸드오프 상황을 고려한다. 이러한 두 가지 핸드오프 상황 동안 최선의 버퍼 크기를 유지하기 위하여 평균에 적합한 CPU 스케줄러의 큐 길이를 설정한다.

<90> 실험에서는 100개의 패킷의 최적 버퍼 사이즈를 고려했다. 동시에 실험 시간이 경과함에 따라 버퍼에 있는 각 패킷의 평균 대기시간도 관찰했다. 하기의 [표 1]은 수직 핸드오프 상황의 가상 매개변수를 나타낸다.

<91> [표 1]

Vertical handoff scenario	Arrival rate before vertical handoff Packets/sec	Arrival rate after vertical handoff Packets/sec	Proportional Change
Low data rate network to High data rate network	10	20	2
	15	20	1.3

<92>

<93> 도 6 은 수직적 핸드오프 수행 후 네트워크 연결의 체증 상태를 나타내는 일예시도로서, 수직적 핸드오프가 일어난 후 본 발명에 따른 자원 관리 장치가 적용된 경우와 적용되지 않은 경우에 있어서 네트워크 연결의 체증 상태를 나타낸다.

<94> 도 6에 도시된 바와 같이, 수직적 핸드오프 후 대역폭이 크게 변화한 경우일수록 네트워크 연결 활용도가 더 낮음을 확실히 볼 수 있다. 여기서, 본 발명의 자원 관리 장치가 적용된 경우 네트워크 연결의 체증이 65%정도 개선되었음을 알 수 있다. 이때, 그래프 상의 큰 변동은 무선 환경에서 나타나는 패킷의 손실률 때문이다.

<95> 도 7 은 수직적 핸드오프가 일어나는 동안 TCP 윈도우의 진행 상태를 나타내는 일예시도로서, 수직적 핸드오프가 일어나는 동안 본 발명에 따른 자원 관리 장치가 적용된 경우와 적용되지 않은 경우에 있어서 TCP 윈도우 진행 상태를 나타낸다.

<96> 도 7에 도시된 바와 같이, 수직적 핸드오프 프로세스가 일어나고 최대 4초 후에는 TCP 윈도우에 있어 최대 수용량에 거의 다다르는 것을 알 수 있다. 그러나,수직 핸드오프 후에 TCP 버퍼 크기를 조정하지 않고 관찰한 경우에, 혼잡 윈도우가 느리게 열리기 때문에 최대 수용량까지 도달하기 위해서는 거의 40초 가까이 소요되는 것을 알 수 있다.

<97> 도 8 은 TCP 연결시 TCP 적합 모듈 사용 여부에 따른 메모리 이용을 나타내는 일예시도이다.

<98> 종래의 경우(a), TCP 소켓의 버퍼가 대역폭 지연 결과에 따라 조정되지 않기 때문에 활용도가 크게 낮음을 알 수 있다. 반면 본 발명의 경우(b), 이동 단말기의 메모리 사용이 40%까지 개선된 것을 알 수 있다.

<99> 도 9 는 총 200초간 TCP 연결 처리량에 대한 일예시도이다.

<100> 이미 WLAN에서 실행되고 있는 다수의 TCP 플로우(flow)에 대해서 실험하였으며, 낮은 데이터 전송량 대역에서

높은 데이터 전송량 대역으로 이동한 경우에 대하여 실험하였다.

<101> 도 9에 도시된 바와 같이, 종래의 경우(a), TCP 연결 처리량은 일정한 수준에 이르기까지 네트워크 변화 후 100초가량 걸리는데 비해, 튜닝한 경우(b)는 변화 후 80초 정도면 일정한 수준의 처리량을 나타내며, 수렴 또한 종래의 경우에 비해 효율적으로 대역을 사용하고 있음을 알 수 있다.

<102> 상기 명시한 바와 같이 수직적 핸드오프에서 동적으로 자원을 관리하여, TCP 레이어와 OS 레이어에서 새로운 네트워크 환경으로 변화된다 하더라도 자원을 효과적으로 이용하기 위함이다. 전송 레이어에서 유효한 대역에 따라 TCP 소켓의 버퍼 크기를 설정함으로써 변화된 네트워크에 맞게 자원이 설정되며, 응용 프로그램의 실행 부분에서는 네트워크 환경 변화에 따른 CPU 스케줄러의 시간 분할에 의해 CPU 자원의 할당이 설정된다.

<103> 실험 결과, 본 발명에 따른 자원 관리 장치가 이동 단말기에 적용되었을 때, 더욱 효율적임을 알 수 있다.

<104> 한편, 전술한 바와 같은 본 발명의 방법은 컴퓨터 프로그램으로 작성이 가능하다. 그리고 상기 프로그램을 구성하는 코드 및 코드 세그먼트는 당해 분야의 컴퓨터 프로그래머에 의하여 용이하게 추론될 수 있다. 또한, 상기 작성된 프로그램은 컴퓨터가 읽을 수 있는 기록매체(정보저장매체)에 저장되고, 컴퓨터에 의하여 판독되고 실행됨으로써 본 발명의 방법을 구현한다. 그리고 상기 기록매체는 컴퓨터가 판독할 수 있는 모든 형태의 기록매체를 포함한다.

<105> 이상에서 설명한 본 발명은, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에게 있어 본 발명의 기술적 사상을 벗어나지 않는 범위 내에서 여러 가지 치환, 변형 및 변경이 가능하므로 전술한 실시예 및 첨부된 도면에 의해 한정되는 것이 아니다.

산업이용 가능성

<106> 본 발명은 이동 단말기들의 자원 관리 등에 이용될 수 있다.

도면의 간단한 설명

<107> 도 1 은 본 발명에 따른 이동 단말기의 자원 관리 장치에 대한 일실시예 구성도,

<108> 도 2 는 본 발명에 따른 네트워크의 유효 대역폭을 계산하는 과정에 대한 일예시도,

<109> 도 3 은 본 발명에 따른 TCP 소켓의 버퍼 크기를 산출 과정에 대한 일실시예 설명도,

<110> 도 4 는 본 발명에 따른 CPU 스케줄러에서의 CPU 할당량 산출 과정에 대한 일실시예 설명도,

<111> 도 5 는 본 발명에 따른 이동 단말기의 자원 관리 장치에서의 자원 관리 방법에 대한 일실시예 흐름도,

<112> 도 6 은 수직적 핸드오프 수행 후 네트워크 연결의 체증 상태를 나타내는 일예시도,

<113> 도 7 은 수직적 핸드오프가 일어나는 동안 TCP 윈도우의 진행 상태를 나타내는 일예시도,

<114> 도 8 은 TCP 연결시 TCP 적합 모듈 사용 여부에 따른 메모리 이용을 나타내는 일예시도,

<115> 도 9 는 총 200초간 TCP 연결 처리량에 대한 일예시도이다.

<116> * 도면의 주요 부분에 대한 부호의 설명

<117> 100 : 단말기 자원 관리부 110 : 단말기 자원 모니터링 모듈

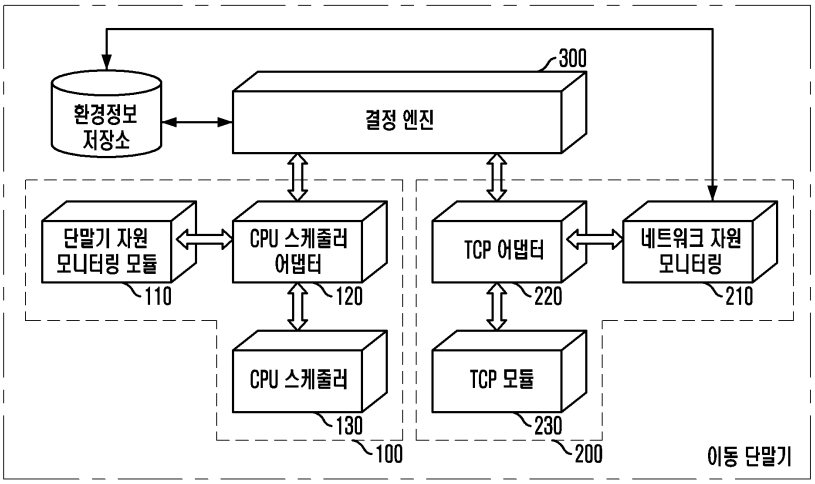
<118> 120 : CPU 스케줄러 어뎁터 130 : CPU 스케줄러

<119> 200 : 네트워크 자원 관리부 210 : 네트워크 자원 모니터링 모듈

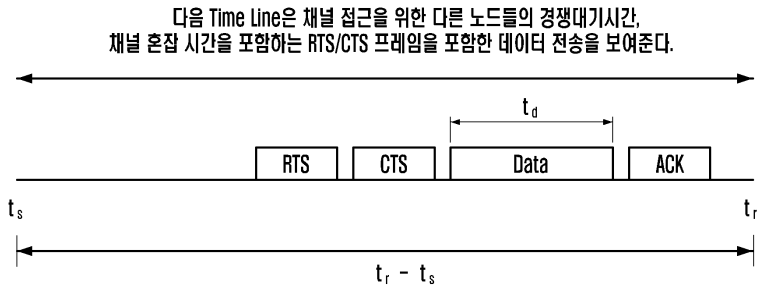
<120> 220 : TCP 어뎁터 230 : TCP 모듈

도면

도면1



도면2



도면3

If (수직적 핸드오프의 결정이 결정 엔진(DE)에 의해 신호되면)
네트워크 N_n의 사용 가능한 대역폭 A_{BWD} 를 구한다.
네트워크 N_n의 패킷 지연시간(RTT) T를 구한다.
$S = A_{BWD}$
Else
네트워크 N_c의 수용량 C를 구한다.
네트워크 N_c의 패킷 지연시간(RTT) T를 구한다.
$S = C \times T$
End if

도면4

$\lambda_{previous} = updateArrivalRate()$ // 패킷의 최초 도착률을 얻는다.

for ever do

$\lambda_{current} = updateArrivalRate()$ // 패킷의 현재 도착률을 얻는다.

$p(t) = \frac{\lambda_{current}}{\lambda_{previous}}$ // 비례 변화를 산출한다.

If $(p(t) < q$ *or* $p(t) > q)$ // 현재 버퍼 레벨을 체크한다.

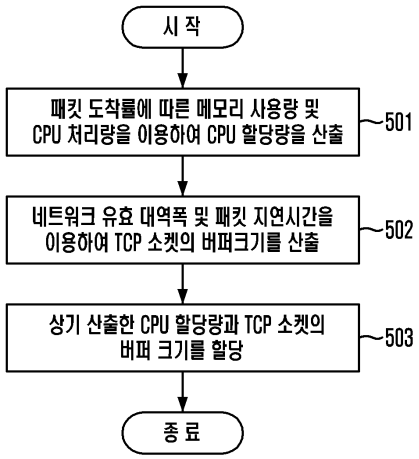
$CPU_{alloc} = CPU_{alloc} * p(t)$ // 그에 비례하여 CPU 리소스를 할당한다.

End if

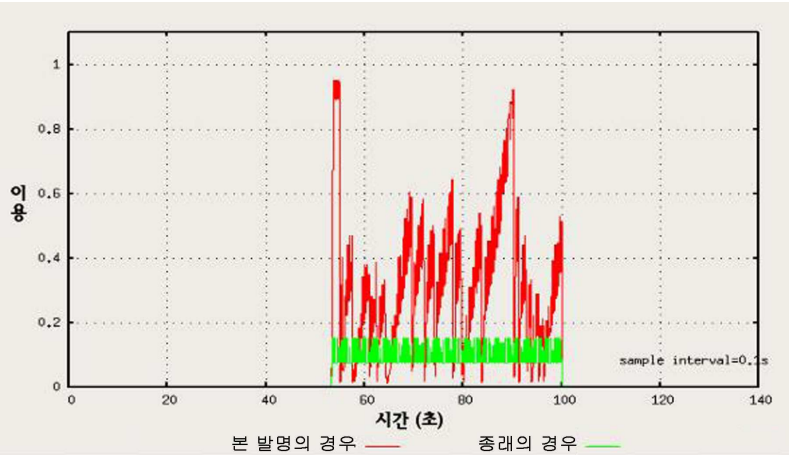
$\lambda_{previous} = \lambda_{current}$

End for

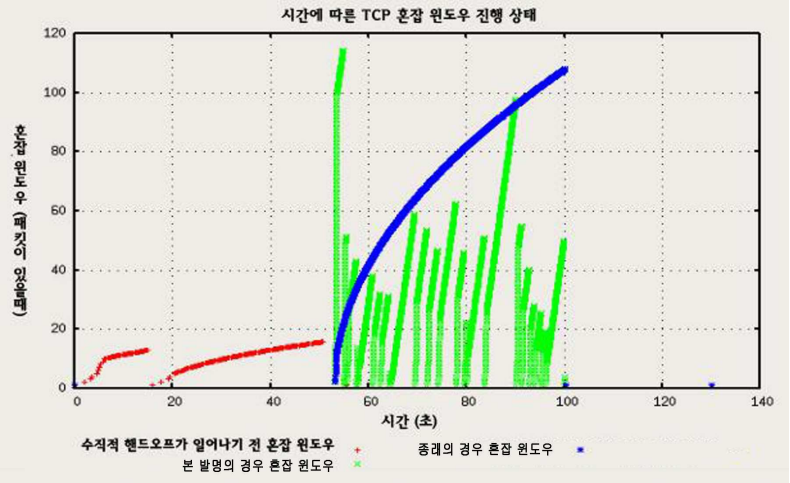
도면5



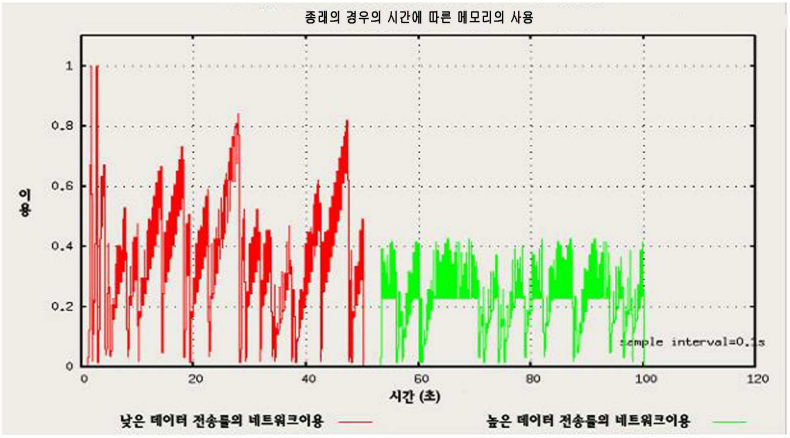
도면6



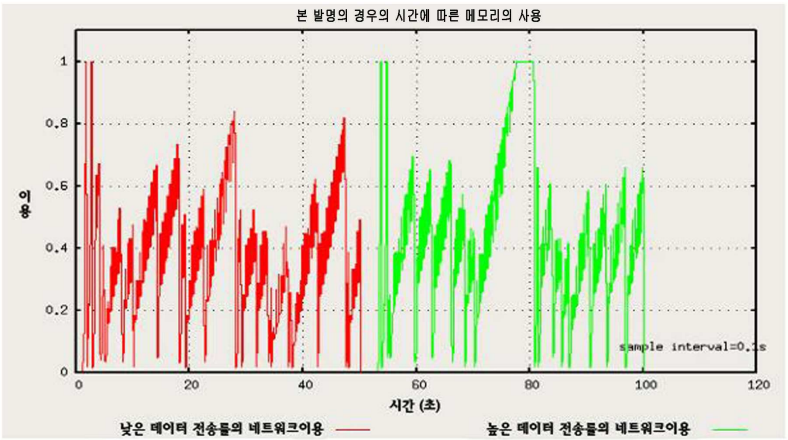
도면7


시간에 따른 TCP 혼잡 윈도우 진행 상태
혼잡 윈도우 (패킷의 갯수)
시간 (초)
수직적 핸드오프가 일어나기 전 혼잡 윈도우
본 발명의 경우 혼잡 윈도우
종래의 경우 혼잡 윈도우

도면8

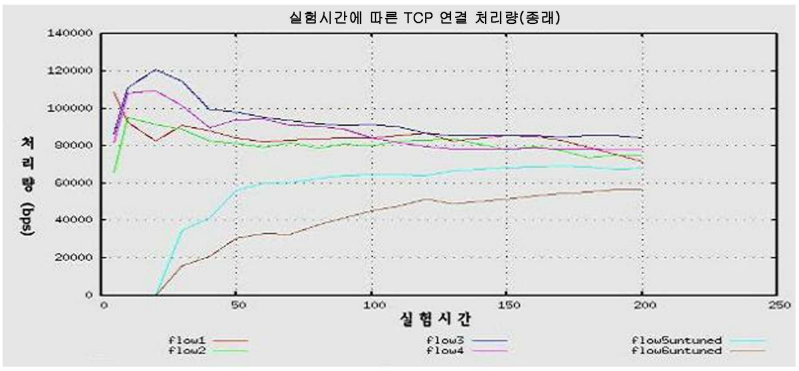

종래의 경우의 시간에 따른 메모리의 사용
이용
시간 (초)
sample interval=0.1s
낮은 데이터 전송률의 네트워크이용
높은 데이터 전송률의 네트워크이용

(a)

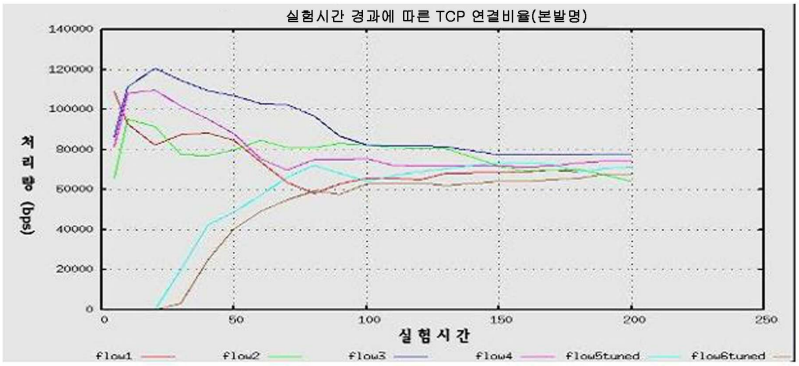

본 발명의 경우의 시간에 따른 메모리의 사용
이용
시간 (초)
sample interval=0.1s
낮은 데이터 전송률의 네트워크이용
높은 데이터 전송률의 네트워크이용

(b)

도면9


실험시간에 따른 TCP 연결 처리량(종래)
처리량 (bps)
실험시간
flow1
flow2
flow3
flow4
flow5untuned
flow6untuned

(a)


실험시간 경과에 따른 TCP 연결비율(본발명)
처리량 (bps)
실험시간
flow1
flow2
flow3
flow4
flow5tuned
flow6tuned

(b)