스위칭이더넷에서 주기적 메시지에 대한 경성 실시간 통신을 위한 메시지 스케쥴링 알고리즘

(Message Scheduling Algorithm for Hard Real-time Communications of Periodic Messages on a Switched Ethernet)

김 명 균 † 이 희 찬 ††
(Myung-Kyun Kim) (Hee-Chan Lee)

요 약 본 논문에서는 스위칭이더넷에서의 주기적 메시지에 대해 경성 실시간 통신을 위한 메시지 전송 모델을 제안하고, 각 메시지들을 마감시간 안에 전송하기 위한 메시지 스케쥴링 알고리즘을 제안한다. 제안하는 스케쥴링 알고리즘은 중앙노드 없이 동작하는 분산 알고리즘이고 스위치에 새로운 기능의 추가 없이 송신자와 수신자 노드 사이에서 동작한다. 제안한 알고리즘은 새로운 주기적 메시지에 대한 전송요청이 오면 이미 스케쥴 되어 있는 주기적 메시지에 영향을 미치지 않고 송신 링크와 수신 링크에서 스케쥴링 가능한지 검사를 하고, 스케쥴링이 가능하면 전송 스케쥴을 생성한다. 본 논문에서 제안하는 스케쥴링 알고리즘은 스위칭이더넷에서 주기적 메시지에 대해 마감시간 내 전송을 보장하고, 새로운 메시지에 대한 동적인 추가가 용이하여 경성 실시간 시스템에서 유연한 메시지 전송 기법을 제공한다.

키워드 : 실시간 통신, 스위칭이더넷, 메시지 스케쥴링, 산업용 통신, 실시간 시스템

Abstract This paper proposes a message transmission model for hard real-time communications of periodic messages on a switched Ethernet and also proposes an algorithm to schedule the messages to be transmitted within their deadlines. The proposed scheduling algorithm is a distributed one and is performed by the source and the destination nodes without the modification of the operational features of the standard Ethernet switch. When a new periodic message needs to be transmitted, it is first checked whether it can be scheduled on both the transmission and the reception links without affecting the already-scheduled messages, and a feasible schedule is made for the new message if it is schedulable. The proposed scheduling algorithm guarantees the transmission of periodic messages within their deadline and allows flexible message transmission on a hard real-time switched Ethernet through the dynamic addition of new periodic messages during run-time.

Key words : Real-time communication, switched Ethernet, message scheduling, industrial communications, real-time systems

1. 서 론

실시간 분산 제어 시스템은 오늘날 프로세스 제어, 공장 자동화, 자동차, 로봇 등 여러 산업 환경에서 광범위하게 사용되고 있다[1]. 스위칭이더넷은 큰 대역폭을 제공하고, 네트워크 트래픽의 미세 분할(micro-segmentation), 양방향 통신 등 실시간 통신을 위한 많은 특징을 제공한다[2,3]. 그러나 스위칭이더넷은 출력단에서의 메시지 충돌로 인해 경성 실시간을 보장하지는 못한다. 따라서 스위칭이더넷에서 경성 실시간 메시지 전송을 지원하기 위해서는 스위치의 출력 버퍼에서의 오버런(overrun)을 막기 위해 네트워크의 트래픽 양을 조절할 수 있도록 하는 기능이 필요하다[4-7]. Loeser 등[5]과 Kweon 등[6]은 Traffic shaping 기술을 이용하여 네트워크 상의 트래픽 양을 제한함으로써 실시간 통신을 제공하는 방법을 제시하였다. 하지만 이들의 방법은 메시지의 최대 지연시간이 보장됨을 보여주지만 각 메시지들이 만족해야하는 명확한 마감시간을 고려하지 않았다. Hoang 등[4]와 Yiming 등[7]은 스위칭이더넷 이더넷에서 EDF(Eariliest Deadline FIrst) 스케쥴링 기반의 경

· 본 연구는 산업자원부 · 울산광역시 지원 울산대학교 네트워크 기반 자동화연구센터의 지원에 의한 것입니다.
† 종신회원 : 울산대학교 컴퓨터정보통신공학부 교수
mkkim@ulsan.ac.kr
†† 학생회원 : 울산대학교 컴퓨터정보통신공학부
dinoguy@gmail.com
논문접수 : 2006년 5월 22일
심사완료 : 2006년 8월 17일

성 실시간 시스템을 제안하였다. 그렇지만 메시지 스케쥴링을 위해 노드뿐 아니라 스위치도 EDF 정책에 따라 메시지를 스케쥴링해야 하므로 노드와 스위치의 MAC 계층 상위에 EDF 스케쥴링을 지원하는 RT(Real-Time) 계층의 추가가 필요하게 된다. Pedreiras 등[8]은 이더넷에서 동적인 실시간 메시지 요구를 지원하기 위해 FTT-ethernet 라는 융통성있는 메시지 전송 모델을 제안하였다. FTT-ethernet은 마스터-슬레이브 모델에 기반하여 동기화된 메시지 전송을 사용한다. 그러나 이들의 방법에서는 슬레이브들은 새로운 메시지 전송요구가 발생할 경우 이를 마스터에 전송하고, 마스터에서 메시지에 대한 스케쥴링을 수행하므로 이더넷의 분산 메시지 전송 모델의 장점을 감소시키고 마스터에 장애가 발생할 경우 네트워크 전체가 마비되는 문제점이 존재한다.

본 논문에서는 스위칭이더넷에서 주기적 메시지에 대해 경성 실시간 통신을 보장하기 위한 메시지 스케쥴링 알고리즘을 제안한다. 제안된 알고리즘은 분산 알고리즘으로 메시지 송신자와 수신자 사이의 동작에 의해 스케쥴링을 수행하고, 기존 스위치에 대한 기능추가 없이 동작한다. 본 논문에서 스위칭이더넷은 Pedreiras 등[8]에서 제안한 것과 유사한 동기화된 메시지 전송 모델을 사용하지만, 새로운 메시지의 전송가능여부(feasibility check) 및 메시지 전송 스케쥴을 추가하기 위해 분산된 방법을 사용한다. 한 노드에서 새로운 주기적 메시지에 대한 요청이 발생할 경우, 이미 스케쥴되어 있는 주기적 메시지들에는 영향을 미치지 않고 송신 링크와 수신 링크에서 스케쥴링이 가능한지를 검사하고, 가능하다면 새로운 메시지에 대한 전송 스케쥴을 생성한다. 이러한 새로운 메시지에 대한 진입제어(admission control)를 위해 송신자와 수신자 사이에 제어 메시지를 교환한다. 본 논문에서 제안하는 스케쥴링 알고리즘은 스위칭이더넷에서 주기적 메시지에 대해 마감시간 내 전송을 보장하고, 새로운 메시지에 대한 동적인 추가가 용이하여 경성 실시간 시스템에서 유연한 메시지 전송을 제공한다.

본 논문의 순서로는 2장에서는 스위칭이더넷에서 메시지 전송 모델에 대해서 기술하고, 3장에서는 스위칭이더넷에서 새로운 주기적 메시지에 대한 전송가능여부 검사(feasibility check)와 마감시간을 보장하기 위한 메시지 스케쥴링 알고리즘에 대해서 기술하며, 4장에서는 제안한 메시지 스케쥴링 알고리즘에 대한 실험결과에 대해 기술하고, 마지막 5장에서 결론과 향후 연구에 대해 제시한다.

2. 스위칭이더넷에서의 메시지 전송 모델

본 논문에서 스위칭이더넷의 각 노드는 그림 1과 같이 송신링크(TL)와 수신링크(RL)로 구성된 양방향 링

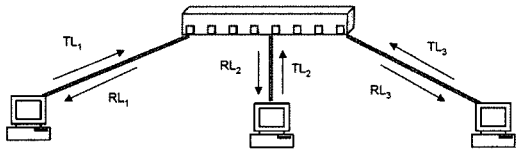


그림 1 스위칭이더넷의 예

크로 스위치와 연결되어 있다. 모든 노드는 동기화 되어 있고 송신링크와 수신링크는 매크로 사이클(Macro Cycle)의 연속으로 구성되어 있다. 매크로 사이클은 그림 2에서와 같이 기본적인 메시지 전송 단위인 기본 사이클(Elementary Cycle: EC)로 나누어져 있다. 각 EC는 주기 사이클(Periodic Cycle: PC)과 비주기 사이클(Aperiodic Cycle: AC)로 나누어지고, PC는 주기적 메시지의 전송을 위해 사용되고 AC는 비주기적 메시지의 전송을 위해 사용된다. 모든 주기적 메시지는 메시지의 마감시간 안에 전송되는 것을 보장하기 위하여 송신링크와 수신링크에 전송되기 전에 메시지의 전송을 위한 대역폭이 할당되어져야만 한다. 새로운 주기적 메시지의 전송을 위한 네트워크 대역폭을 할당하기 위해 진입제어 프로세스가 수행되며 이를 위해 송신자와 수신자는 두 개의 진입제어 메시지(Periodic_msg_req와 Periodic_msg_rep 메시지)를 교환한다. 이 진입제어 메시지들은 AC 구간에서 송신자와 수신자 사이에 전송되어지며, 이 제어 메시지가 해당 AC 구간에서 전송이 완료될 수 있도록 비주기 메시지들 중 가장 높은 우선순위를 가지게 된다. PC의 길이(PL)와 AC의 길이(AL)는 시스템의 주기적 메시지와 비주기적 메시지의 비율에 따라 결정된다. 본 논문에서는 주기적 메시지에 대해서만 고려한다.

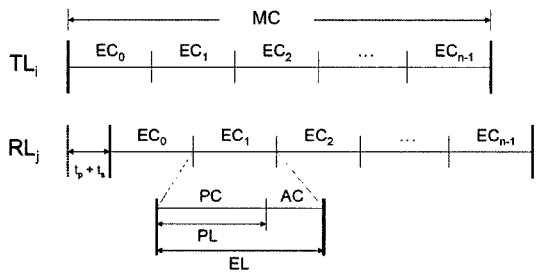


그림 2 메시지 전송 모델

노드 N_i에서 노드 N_j로 전송되는 주기적 메시지 SM_{ij}는 $\{P_{ij}, D_{ij}, C_{ij}\}$의 실시간 전송 요구사항을 가지며, 여기에서 P_{ij}, D_{ij}, C_{ij}는 각각 메시지의 주기, 마감시간, 그리고 메시지의 길이를 나타낸다. 본 논문에서는 $D_{ij} = P_{ij}$ = k*EL(단 k는 양수정수이며, EL은 EC의 길이를 나타냄)이며, 모든 주기적 메시지의 요구사항이

사전에 알려져 있으며, 하나의 MC를 구성하는 EC의 수 M은 각 메시지들의 주기들에 대한 최소공배수(Least Common Multiple), 즉 M = LCM{P_{ij}/EL}개의 EC로 구성된다. 자동차 네트워크 또는 프로세스 제어 네트워크와 같은 실시간 제어 네트워크들에서는 각 실시간 태스크들의 메시지 전송 요구사항들이 미리 알려지게 된다. 따라서 본 논문에서는 실시간 태스크들에서 요구하는 주기적인 메시지들에 대한 실시간 전송 요구사항이 모두 알려져 있다고 가정하며, 이에 따라 M은 전체 시스템 동작 전에 정해지고 고정된 값을 갖는다. 본 시스템에서 스위치에 연결되어 있는 모든 노드들은 동기화되고, 모든 송신링크의 매크로 사이클 또한 동기화 되어 있다. 수신링크의 출력버퍼가 비어있을 때, 송신링크의 EC_0에서 전송되는 첫 번째 메시지는 송신링크에서의 전파지연시간(t_p) 더하기 스위치내에서의 스위칭지연시간(t_s) 후에 수신노드의 수신링크에 도착하게 된다. 그러므로 그림 2와 같이 수신링크에서의 MC의 시작은 송신링크에서의 시작 시간보다 (t_p+t_s) 시간 후에 시작한다.

p = P_{ij}/EL 이고 0≤k≤M/p-1일 때, (k*p)번째 EC부터 ((k+1)*p-1)번째 EC까지의 EC들을 주기 P_{ij}인 주기적 메시지의 k번째 주기구간(period boundary)이라고 한다. 스케쥴링이 된 기존의 메시지를 전송하는 도중 주기 P_{ij}를 갖는 새로운 메시지가 추가되면, 송신자는 해당 메시지를 그 메시지의 다음 주기구간(period boundary)부터 전송을 시작한다.

3. 스위칭이더넷에서 주기적 메시지의 실시간 스케쥴링

스위칭이더넷에서 각 노드는 하나의 MC내의 각 EC에서 전송되어야 하는 메시지들의 순서를 포함하고 있는 '메시지스케쥴'을 유지하고 있다. 스케쥴된 메시지의 전송 도중 하나의 송신자에서 새로운 주기적 메시지의 전송이 필요하면, 해당 EC내의 AC에서 새로운 메시지에 대한 전송가능여부 검사와 메시지 스케쥴링을 위해 송신자와 수신자는 진입제어 프로세스를 수행하게 되며, 만약 새로운 메시지가 전송 가능하다면, 그 메시지는 메시지스케쥴에 추가되고, 해당 메시지의 다음 주기구간부터 전송되게 된다. 실시간 전송 요구사항{P_{ij}, D_{ij}, C_{ij}}를 갖는 새로운 주기적 메시지 SM_{ij}에 대한 진입제어 프로세스는 다음과 같다.

(i) 노드 N_i는 송신링크 TL_i에서의 메시지 전송가능여부를 검사한 후 가능하다면 AC 구간에서 노드 N_j에게 Periodic_msg_req 제어메시지를 전송한다.

(ii) 노드 N_i로부터 Periodic_msg_req 메시지를 수신하면, 노드 N_j는 RL_j의 전송가능여부를 판단한 뒤 Periodic_msg_rep 메시지에 결과를 노드 N_i에 전송한다.

(iii) 노드 N_i는 Periodic_msg_rep 메시지를 수신하면, 메시지를 해독하여 메시지 SM_{ij}가 송신링크 TL_i와 수신링크 RL_j에서 모두 전송가능하면 메시지스케쥴에 SM_{ij}를 추가한다.

3.1 송신링크에서 전송가능여부검사

실시간 전송 요구사항{P_{ij}, D_{ij}, C_{ij}}을 갖는 메시지 SM_{ij}의 스케쥴링은 하나의 MC내의 각 주기구간에서 메시지 SM_{ij}을 전송가능한 EC를 하나씩, 전체적으로 M/p(여기서, p = P_{ij}/EL) 개의 EC를 찾는 것이다. 새로운 메시지가 전송가능하면 송신자는 송신링크의 해당 EC들의 메시지스케쥴 맨 뒤에 그 메시지를 추가한다. 송신링크에서의 전송가능여부 검사를 위해 각 노드는 송신링크의 각 EC에 대한 현재 전송트래픽량에 대한 정보를 유지하고 있다.

그림 3은 송신링크 TL_3에서의 전송가능여부 검사에 대한 예를 보여주고 있다. 이 예제에서 MC는 6개의 EC로 구성되어 있고, EC의 길이는 1.2, PC의 길이는 1.0 (AC의 길이는 0.2)이다. 그림 3(a)는 현재 TL_3의 각 EC들에서 전송되고 있는 메시지스케쥴을 보여준다. 여기에서 $T_{3,i}$은 TL_3의 EC_i에서의 현재 전송트래픽량을 나타내며, T_3 = {$T_{3,i}$: 0< i <5}이다. 현재 상태에서 노드 N_3이 실시간 요구사항이 {2, 2, 0.4}인 새로운 메

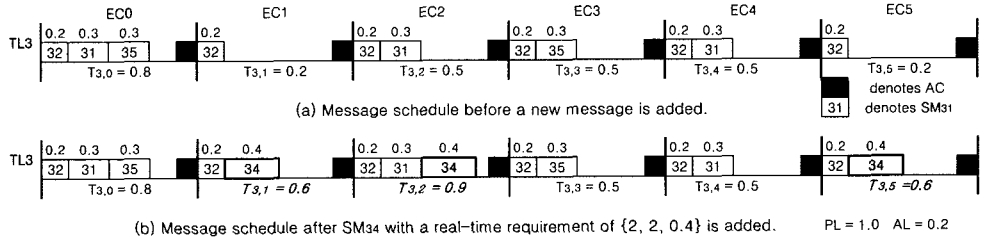


그림 3 송신링크에서의 전송가능여부 검사 예

시지 SM_{34}를 노드 N_4로 전송하고자 하면, 본 논문의 알고리즘은 먼저 메시지 SM_{34}의 각 주기구간에서 현재 전송트래픽량이 가장 작은 EC들을 찾고, 해당 EC들에서 메시지 SM_{34}이 전송가능 한 지를 검사한다. 그림 3(a)의 예제에서 SM_{34}의 주기가 2이므로 각 주기구간에서 현재 전송트래픽량이 가장 작은 EC들은 {EC_1, EC_2, EC_5} (혹은 {EC_1, EC_3, EC_5})가 된다. {EC_1, EC_2, EC_5}내의 모든 EC들은 PC가 C_{34} (0.4) 보다 큰 여유공간을 갖고 있으므로 메시지 SM_{34}을 해당 EC내에서 전송가능하다. 따라서 SM_{34}은 송신링크 TL_3에서 전송가능하고 {EC_1, EC_2, EC_5}에 속하는 각 EC의 메시지 스케쥴에 추가되게 된다. 그림 3(b)는 메시지 SM_{34}가 추가된 후의 TL_3의 메시지스케쥴을 보여주고 있다.

송신노드는 새로운 메시지가 송신링크에서 전송 가능하다면, 수신링크에서의 전송가능여부를 검사하기 위해 수신노드에게 Periodic_msg_req 제어메시지를 전송한다. Periodic_msg_req 제어메시지에는 수신노드가 수신링크에서 전송가능여부를 검사하기 위해 필요한 정보, 즉 새로운 메시지에 대한 실시간 요구사항, 송신링크의 각 EC에서의 현재 전송트래픽량들에 대한 정보를 포함하여 전송한다. 수신노드는 Periodic_msg_req 메시지를 받으면, 수신링크에서 전송가능여부를 검사하고 그 결과를 Periodic_msg_rep 메시지를 통해 송신노드로 전송하게 된다. 송신노드는 Periodic_msg_rep 메시지를 받으면, 메시지를 해독하여 새로운 메시지가 송신링크과 수신링크 모두에서 전송가능하다면, 메시지스케쥴에 새로운 메시지를 추가하게 된다. 다음 알고리즘 1은 송신링크 TL_i에서 새로운 메시지 SM_{ij}에 대한 전송가능여부를 검사하는 알고리즘이다.

```
// 알고리즘 1 : TL_i에서 새로운 메시지 SM_ij에 대한
전송가능여부 검사 알고리즘
// 메시지 SM_ij의 실시간 전송 요구사항 : {P_ij, D_ij, C_ij}
// MC 길이 : M
1.  p = P_ij/EL ;
2.  for (k = 0; k ≤ (M/p - 1); k++) {
3.      T_min = min {T_i,k*p+m};
                0≤m≤p-1
4.      if (T_min + C_ij > PL) {      // Not feasible
5.          메시지 SM_ij에 대한 전송요구를 거절;
6.          return;
7.      }
8.  }
9.  Periodic_msg_req[SM_ij, {P_ij, D_ij, C_ij}, T_i] 메시지를
노드 N_j에 전송;
10. return;
```

3.2 수신링크에서의 전송가능여부 검사

송신노드의 각 EC에서 전송되어진 모든 메시지는 (t_p +t_s) 시간 후에 스위치의 수신노드의 수신링크 출력 큐에 도착하게 된다. 같은 출력 큐에 도착한 메시지들은 큐에 저장되고 FIFO (First-In, First-Out)의 순서로 수신링크를 통해 전송되어 진다. 새로운 메시지가 수신링크의 EC에서 전송가능하기 위해서는 그 메시지가 이미 스케쥴 되어 있는 다른 메시지들에 영향을 미치지 않고 해당 EC내에서 전송이 종료될 수 있어야 한다. 수신링크의 전송가능여부 검사를 위해 각 노드는 수신링크의 MC내의 각 EC들에 대해 전송완료예상시간 (expected transmission finishing time)을 유지한다. 수신링크 RL_j의 EC_k에서의 전송완료예상시간($R_{j,k}$)은 현재 EC_k에서 노드 N_j로 전송되는 모든 주기적 메시지가 수신링크 RL_j를 통해 스위치에서 전송이 완료되는 예상시간을 의미한다. 새로운 메시지 SM_{ij}가 EC_k에 추가될 경우, 이 전송완료예상시간 $R_{j,k}$는 $R_{j,k}$ 현재값, C_{ij} (메시지 SM_{ij}의 길이), 그리고 $T_{i,k}$(송신링크 TL_i에서 SM_{ij}가 전송되는 시간)으로부터 구해질 수 있다.

예를 들어, 노드 N_j가 노드 N_i로부터 Periodic_msg_req[SM_{ij}, {P_{ij}, D_{ij}, C_{ij}}, T_i] 메시지를 수신하게 되면, R_j, T_i, 그리고 C_{ij}를 이용하여 SM_{ij}의 수신링크 RL_j에서의 전송가능여부를 검사하게 된다. RL_j의 EC_k에서 SM_{ij}가 전송가능하기 위해서는,

$$\max\{R_{j,k},\ T_{i,k}\} + C_{ij} \le \mathrm{PL}$$

을 만족하여야 한다. 새로운 메시지 SM_{ij}가 EC_k에 송신 및 수신링크 모두에서 전송가능하면, 송신노드는 SM_{ij}를 EC_k에 전송하도록 메시지스케쥴에 추가하게 되고, 수신노드는 수신링크 RL_j의 EC_k에 대한 전송완료예상시간 $R_{j,k}$를 다음과 같이 갱신하게 된다.

$$RL_j = \max\{R_{j,k},\ T_{i,k}\} + C_{ij}$$

그림 4는 수신링크 RL_j에서의 전송가능여부 검사와 전송완료예상시간이 갱신되는 예를 나타내고 있다. 그림 4의 (a)와 (b)는 수신 가능한 상태를 나타내는 예를 나타내고, (c)의 예에서는 새로 추가할 메시지 SM_{ij}에 대해

$$\max\{R_{2,1},\ T_{5,1}\} + C_{52} = 0.8 + 0.3 = 1.1 > \mathrm{PL}$$

이므로 해당 EC내에서 전송완료를 보장할 수 없다.

다음 알고리즘 2는 수신노드가 송신노드로부터 Periodic_msg_req 메시지를 수신하였을 때 수신링크상에서의 전송가능여부를 검사하는 알고리즘이다.

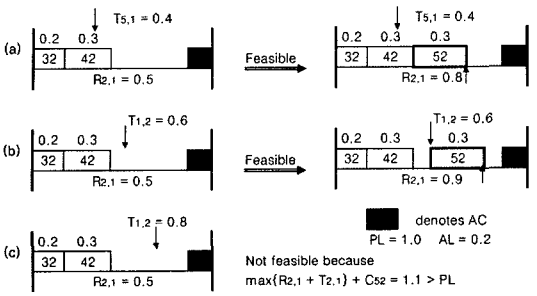


그림 4 RL_2의 EC_1에서의 메시지 SM_{52}의 전송가능여부 검사 예

```
// 알고리즘2 : RL_j에서 SM_ij에 대한 전송가능성 검사
// 메시지 SM_ij의 실시간 전송 요구사항 : {P_ij, D_ij, C_ij}
// MC 길이 : M
// next(S): 집합 S의 다음 원소를 반환
1.  p = P_i,j/EL ;   S =∅;
2.  for (k = 0; k ≤ (M/p - 1); k++) {
3.      R_min = max {R_j,k*p, T_i,k*p};
4.      imin = k * p;   m = 1;
5.      while ( m ≤ p - 1) {
6.          if (R_min > max {R_j,k*p+m, T_i,k*p+m}) {
7.              R_min = max {R_j,k*p+m, T_i,k*p+m};
8.              imin = k * p + m;
9.          }
10.         m = m + 1;
11.     }
12.     if (R_min + C_ij > PL) {   // Not feasible
13.         Periodic_msg_rep[SM_ij, NULL] 메시지를
노드 N_i에 전송;
14.         return;
15.     } else
16.         S = S ∪ {imin};
17. }
18. Periodic_msg_rep[SM_ij, S] 메시지를 노드 N_i에 전송;
19. while ((k = next(S)) ≠ NULL)      // R_j 갱신
20.     R_j,k = max {R_j,k, T_i,k} + C_ij;
21. return;
```

3.3 메시지스케줄 갱신

노드 N_j는 노드 N_i로부터 Periodic_msg_rep[SM_{ij}, S]를 수신하면 메시지를 해독하여 수신링크 RL_j에서의 전송가능여부 검사 결과를 확인하게 된다. 만약 S가 NULL이면 SM_{ij}는 RL_j에서 전송가능하지 않음을 나타내므로 메시지 SM_{ij}의 전송요청은 거절된다. 만약 S가 NULL이 아니면, 메시지 SM_{ij}는 S에 속한 EC들에서 전송가능함을 나타낸다. 메시지 SM_{ij}가 전송가능하면 노드 N_i는 S에 속한 EC들의 메시지테이블에 SM_{ij}을 추가하고, 해당 EC의 현재 전송트래픽량 (T_i)을 갱신한다. 이에 따라 노드 N_i는 새롭게 추가된 메시지 SM_{ij}를 메시지의 다음 주기구간부터 전송을 시작하게 된다. 다음 알고리즘 3은 송신노드가 수신노드로부터 Periodic_ msg_ rep 메시지를 수신하였을 때 수행하는 알고리즘이다.

```
// 알고리즘 3: TL_i의 메시지테이블 갱신
// 메시지 SM_ij의 실시간 전송 요구사항 : {P_ij, D_ij, C_ij}
// MC 길이 : M
// next(S): 집합 S의 다음 원소를 반환
// Update_msg_schedule(SM_ij, k): EC_k의
// 메시지스케줄에 SM_ij를 추가하는 함수
1.  if (S == NULL) {
2.      메시지 SM_ij의 전송요구를 거절;
3.      return;
4.  }
5.  p = P_i,j/EL ;
6.  while ((k = next(S)) ≠ NULL) {
7.  T_i,k = T_i,k + C_ij; // 현재 전송트래픽량 (T_i) 갱신
8.  Update_msg_schedule(SM_ij, k);
9.  }
10. return;
```

4. 메시지 스케줄링 알고리즘의 실험 및 결과

제안된 스케줄링 알고리즘을 구현하기 위해 고정밀 POSIX 타이머 기능을 가지는 리눅스 2.6.10[11]을 사용하였고, Cisco Catalyst WS-C1912C-EN 스위치를 사용하여 100 Mbps의 대역폭을 가진 네트워크에서 실험을 진행 하였다. Cisco Catalyst WS-C1912C-EN 스위치는 Cut-through 스위칭 모드로 동작할 경우 100 Mbps의 포트에서 7μs의 스위칭 지연 시간을 가진다. 실험에서 사용한 스위칭이더넷은 하나의 스위치와 5개의 노드로 구성되어 있으며, 각 노드들은 100 Mbps의 속도로 스위치와 전이중 방식으로 연결되어 있다.

본 실험에서는 스위칭이더넷상의 노드들의 MC들의 동기화를 위해 하나의 노드를 마스터노드로 두고 마스터 노드는 각 MC마다 SYNC 메시지를 브로드케스트하여 MC의 시작을 하려주도록 하였다. 그림 5는 실험에서 사용한 네트워크 메시지 전송모델을 나타낸다. 각노드는 SYNC 메시지를 수신함으로써 MC의 시작시간을 알 수 있다. 네트워크에서 Mater노드는 SYNC 메시지만을 전송하고 다른 메시지 전송에는 관여하지 않았고, 다른 동기화 기법을 사용한다면 마스터 노드는 필요하지 않다[8,9]. SYNC 메시지는 64바이트(프레임 최소길이)의 크기를 가지고, M(한 MC내의 EC의 수), EL, PL, AL에 대한 정보를 가지고 있다. 본 실험에서는 그림 5와 같이 M = 6, EL = 1 ms, PL = 0.8 ms, AL = 0.2ms의 값을 가진다.

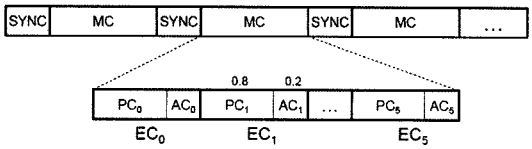


그림 5 실험에 사용된 메시지 전송 모델.

본 실험에서 각 노드는 메시지 길이가 [20, 80] μs 사이의 값을 갖고 주기는 {1, 2, 3} 중 하나의 주기를 갖는 30개의 메시지를 임의로 생성하고 이들을 메시지버퍼에 저장하고 있다. 그림 6은 본 실험에 의해 생성한 메시지에 대한 메시지 길이 및 주기 값에 관한 그래프이다.

스케쥴링 알고리즘은 이미 스케쥴된 메시지들을 전송하는 도중 새로 발생하는 주기적 메시지들을 동적으로 추가할 수 있도록 설계되었다. 이러한 메시지의 동적추가 기능을 보이기 위하여 실험을 다음과 같이 두 단계로 나누어 수행하였다.

(i) 각 노드는 첫 MC가 시작되기 전 메시지버퍼에 저장된 메시지 중 처음 15개의 메시지에 대해서 먼저 진입제어 프로세스를 수행한다.

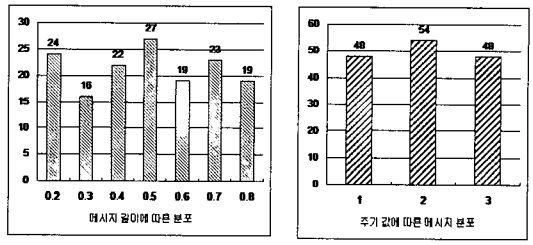
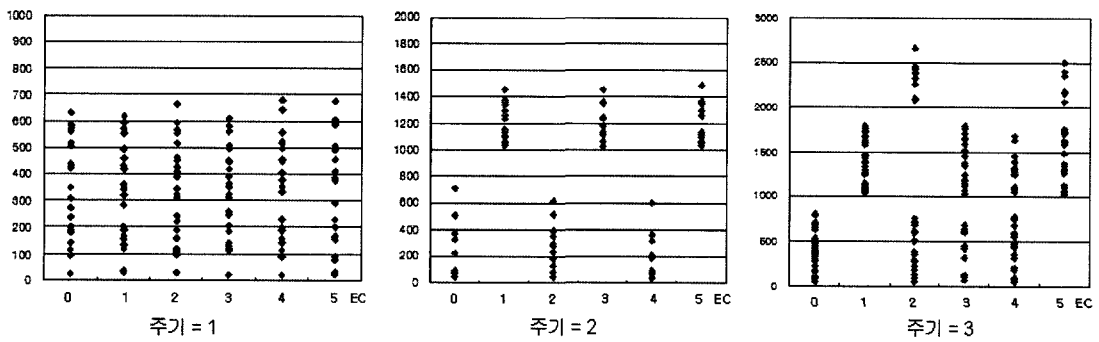


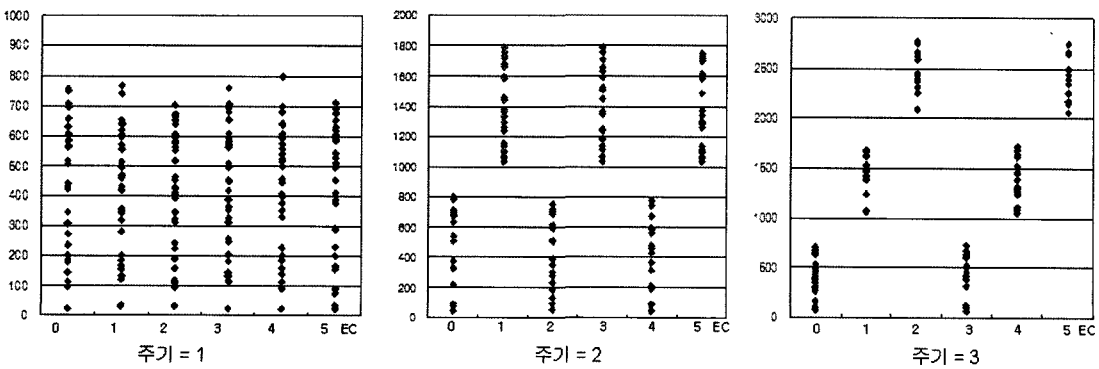
그림 6 실험에 사용한 메시지의 길이 및 주기에 대한 분산

(ii) 각 노드는 다음 MC에서 (i) 단계에서 스케쥴된 메시지들을 전송하는 도중 나머지 15개의 메시지들에 대해 순서대로 진입제어 프로세스를 수행하였다. (ii) 단계의 진입제어 프로세스는 진입제어 프로세스에 실패하는 첫 번째 메시지가 발생할 때까지 수행하였다.

스케쥴된 메시지가 마감시간내에 전송됨을 보이기 위하여 각 메시지들에 대한 응답시간(response time)을 측정하였다. 스케쥴된 각 메시지들에 대한 새로운 인스턴스 (instance)는 각 메시지의 주기구간 시작에서 생성된다고 가정하였다. 각 메시지의 응답시간은 메시지가 생성된 시점에서 메시지의 전송이 완료되는 시점까지의



(a) 1단계에서 스케쥴된 메시지들에 대한 응답기간



(b) 2단계에서 스케쥴된 메시지를 포함한 전체 메시지들에 대한 응답시간

그림 7 스케쥴된 메시지들에 대한 응답시간

시간을 의미한다. 그림 7은 스케쥴된 각 메시지의 주기에 따른 스위칭이더넷에서의 응답시간을 나타내고 있다. 그림 7(a)는 단계 (i)에서 스케쥴된 처음 15개의 메시지들에 대한 응답시간을 나타내고, 그림 7(b)는 단계 (ii)를 통해 다음 MC 구간에서 동적으로 추가된 메시지들을 포함한 전체 메시지들에 대한 응답시간을 나타내고 있다. 그림 7에서 스케쥴된 모든 메시지는 메시지의 마감시간 안에 전송되고 있음을 확인할 수 있다.

제안된 메시지 스케쥴링 알고리즘의 네트워크 활용율(Utilization)를 구하기 위하여 단계 (ii)에서 첫 번째 진입제어 발생하였을 때의 네트워크 활용율을 측정하였다. 네트워크 활용율은 다음과 같이 정의 된다.

$$U = \frac{\sum -of-all-the-messages-transmitted-\in -a-MC}{M * PL}$$

본 실험에서 사용한 메시지들에 대해 네트워크 활용율은 U = 0.80으로 측정되었다.

5. 결론 및 향후 계획

실시간 분산 제어 시스템은 프로세스 제어, 공장 자동화, 자동차산업 등 여러 산업 분야에 광범위하게 사용되고 있다. 그러한 응용들에서 각 테스크는 반드시 정해진 마감시간 안에 수행되어야 하고 테스크들 사이의 통신 또한 테스크의 실시간 요구사항을 만족하도록 마감시간 안에 전송되어야 한다. 사무실 네트워크 환경에서 가장 많이 사용되어지고 있는 스위칭이더넷은 실시간통신을 위한 좋은 특징을 가지고 있다. 그러나 스위칭이더넷은 출력단에서의 메시지 충돌로 인해 산업용 응용프로그램들이 필요로 하는 경성 실시간 통신 요구사항을 만족하기 위해서는 네트워크상에서의 트래픽양을 조절하는 기법이 요구된다.

본 논문에서는 스위칭이더넷에서 주기적 메시지의 경성 실시간 통신을 위한 동기화된 전송 모델을 제안하고 실시간성을 갖는 주기적 메시지들에 대해 동적 추가가 가능한 메시지 스케쥴링 알고리즘을 제안하였다. 제안된 스케쥴링 알고리즘은 스케쥴링을 위한 중앙노드가 필요없이 동작하는 분산 스케쥴링 알고리즘이며 기존 이더넷 스위치에 대한 기능 수정없이 송신노드와 수신노드 사이에서 수행된다. 제안된 메시지 스케쥴링 알고리즘은 스케쥴된 주기적 메시지들이 마감시간 안에 전송됨을 보장하고, 이미 스케쥴된 메시지들을 전송하는 도중 새로운 메시지들에 대한 추가가 가능하여, 실시간 요구사항이 동적으로 변하는 산업용 응용프로그램들을 위해 필요한 유연한 경성 실시간 통신 기법을 제공한다.

참 고 문 헌

[1] G. C. Buttazzo, Hard real-time computing systems: Predictable scheduling algorithms and applications, 2nd ed., Springer, 2005.

[2] Y. Song, "Time-constrained communication over switched Ethernet," In Proc. FeT'01, pp. 138-143, 2001.

[3] J. Jasperneit and P. Neumann, "Switched ethernet for factory communication," 8th IEEE Int'l Conf. on Emerging Technologies and Factory Automation, Antibes, France, 2001.

[4] H. Hoang, et. al., "Switched realtime ethernet with earliest deadline first scheduling- protocols and traffic handling," In Proc. of Int'l Workshop on Parallel and Distributed Real-Time Systems, Fort Lauderdale, FL, USA, 2002.

[5] J. Koeser andg H. Haerti, "Low-latency hard real-time communication over switched Ethernet," In Proc. ECRTS, 2004.

[6] S. K. Kweon, K. G. Shin, and G. Workman, "Achieving real-time communication over Ethernet with adaptive traffic shaping," In Proc. IEEE Real-Time Technology and Applications Symposium, pp.90-100, 2000.

[7] A. Yiming and T. Eisaka, "Support industrial hard real-time traffic with switched Ethernet," In ICESS, 2005.

[8] P. Pedreiras, L. Almeida, and P. Gai, "The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency," In Proc. ECRTS, pp.134-142, 2002.

[9] Kopez H. and Ochsenreiter W.: Clock synchronization in distributed real-time systems. IEEE Tr. on Computers C-06, 8 (1987), pp.833-839. 13.

[10] IEEE: 1588 IEEE standard for a precision clock synchronization protocol for networked measurement and control systems. Tech. Rep., IEEE Instrumentation and measurement Society (2002).

[11] Linux with high resolution POSIX timers: http://sourceforge.net/projects/high-res-timers.

김 명 균

1980년~1986년 서울대학교 컴퓨터공학과(학사). 1986년~1988년 한국과학기술원 전산학과(석사). 1990년~1996년 한국과학기술원 전산학과(석사). 1989년~1998년 전주우석대학교 교수. 1998년~현재 울산대학교 컴퓨터정보통신공학부 교수. 관심분야는 실시간통신, 산업용통신, 홈네트워크

이 희 찬

2000년~2004년 울산대학교 컴퓨터정보통신공학부(학사). 2004년~현재 울산대학교 컴퓨터정보통신공학부(석사과정). 관심분야는 실시간통신, 산업용통신, 홈네트워크