분산 컴퓨팅 환경에서 실시간 메시지 통신을 위한 TTP/C 프로토콜

(A Protocol of TTP/C(timed token protocol with concession) for Real-Time Messages in Distributed Computing Environment)

오 성 흔 † 최 중 섭 † 양 승 민 ††
(Sung-Heun Oh) (Joong-Sup Choi) (Seung-Min Yang)

요 약 분산 실시간 시스템에서의 노드간 주고 받는 메시지는 크게 동기적 메시지와 비동기적 메시지로 나눌 수 있다. 동기적 메시지는 샘플링된 음성이나 화상 데이타와 같이 주기적으로 발생되는 메시지로 전달시간 제약을 가지므로 종단간 마감시간을 보장해 주어야 한다. 비동기적 메시지는 비주기적인 메시지로 일정하지 않게 발생하며 일반적으로 전달 시간 제약사항이 엄격하지 않다.

본 논문에서는 분산 환경에서 동적으로 발생하는 동기적 메시지에 대해 높은 마감시간 보장성을 제공하는 실시간 프로토콜 TTP/C(timed token protocol with concession)을 제안한다. TTP/C는 비동기적 메시지나 긴급하지 않은 동기적 메시지의 전송 대역폭을 양보함으로써 긴급한 동기적 메시지의 마감시간을 보장한다. 또한 TTP/C는 노드에서 전송되는 동기적 메시지의 발생시간이 주기와 다르게 일정치 않더라도 높은 보장성을 갖는다. 모의 실험을 통하여 TTP/C가 기존의 TTP에 비해 동기적 메시지의 마감시간 보장성에 대해 높은 성능 향상을 가져옴을 보인다.

Abstract Messages in distributed real-time systems are categorized into two groups: synchronous messages and asynchronous messages. Synchronous messages, such as sampled audio and image data, are generated periodically with delivery time constraints. Protocols should guarantee the end-to-end deadlines for such messages. Asynchronous messages are non-periodic and may arrive in a random way with no strict time constraints.

In this paper, we propose TTP/C(timed token protocol with concession), an extension of TTP protocol, to achieve higher timeliness guarantee for synchronous messages in distributed real-time systems. In TTP/C, a node concedes the allocated bandwidth to other nodes with urgent synchronous messages to be sent provided that the node has no urgent messages. TTP/C works very well even if the synchronous messages are generated with some jittering by nodes. The simulation results show the improved performance of TTP/C protocol for guaranteeing synchronous messages deadlines compared to the existing TTP protocols.

1. 서 론

분산 실시간 시스템에서 여러 노드에 분산되어져 있는 태스크들은 공동의 목적을 달성하기 위해 협력하며 수행된다. 이렇게 노드에 분산된 태스크의 수행이 마감시간 내에 이루어지기 위해서 태스크들 사이에서 이루어지는 실시간 메시지 전송은 제한된 시간 내에 이루어져야 한다. 전송 매체가 여러 노드에 의해 공유되는 네트워크의 경우 MAC(medium access control) 프로토콜은 실시간 메시

· 본 연구는 한국과학재단 특정기초 연구과제 지원(과제번호: 96-0101-07-01-3)으로 수행되었습니다.
† 비 회 원 : 숭실대학교 컴퓨터학과
honestly@realtime.soongsil.ac.kr
jschoi@realtime.soongsil.ac.kr
†† 종신회원 : 숭실대학교 컴퓨터학과 교수
yang@computing.soongsil.ac.kr
논문접수 : 1999년 11월 22일
심사완료 : 2000년 3월 13일

지가 마감시간 내에 전송되도록 충분한 전송 대역폭과 예측 가능한 접근 방법을 제공해야 한다[1][2].

예측 가능한 실시간 메시지 전송을 위해 몇 가지 MAC 프로토콜들이 제안되었다[1][3][4][5]. 노드가 연속적으로 토큰을 받는 시간의 간격이 제한(bound)되어 노드가 전송 매체에 예측 가능하게 접근할 수 있도록 하는 TTP(timed token protocol)는 많은 내장형 실시간 응용에서 사용되는 Fiber Distributed Data Interface(FDDI), IEEE 802.4, High-Speed Data Bus and High-Speed Ring Bus(HSDB/HSRB)와 Survivable Adaptable Fiber Optic Embedded Network(SAFENET)과 같은 표준 프로토콜로 통합되었다[1]. TTP는 메시지 발생의 규칙성에 따라 동기적 메시지(synchronous message)와 비동기적 메시지(asynchronous message)로 나눈다. 동기적 메시지는 샘플링된 음성 또는 화상 데이타와 같이 주기적으로 발생되는 메시지로 전달 시간 제약사항을 가질 수 있다. 비동기적 메시지는 비주기적인 메시지로 일정하지 않게 발생하며 일반적으로 전달 시간 제약사항을 갖지 않는다. TTP를 사용하는 네트워크의 초기화 시, 예상 토큰 순환 시간을 나타내는 TTRT(target token rotation time)라는 프로토콜 파라미터가 결정된다. 각 노드는 동기적 용량(synchronous capacity) H_i 를 할당받는데 이는 $TTRT$ 시간의 한 부분으로 포함된다. 동기적 용량은 노드가 토큰을 받을 때 마다 동기적 메시지를 전송할 수 있는 최대 시간을 말한다. 비동기적 메시지는 토큰이 예상보다 일찍 노드에 도착했을 때 전송된다. 각 노드의 동기적 용량을 결정하는 동기적 대역폭 할당(SBA 또는 synchronous bandwidth allocation) 기법에 대한 많은 연구가 이루어졌다[1][2][6][7][8].

TTP에서 동기적 메시지가 마감시간 내에 전송되기 위해서는 해당 주기의 시작시간부터 마감시간까지 일정한 횟수 이상의 토큰을 받아 동기적 메시지를 전송해야 한다. 만약 전송할 동기적 메시지가 해당 주기의 시작시간에 발생하지 않고 지연되어 발생하는 경우 노드는 비동기적 메시지만 전송한 후 다음 노드로 토큰을 전달한다. 결과적으로 동기적 메시지를 전송하기 위한 토큰 이용의 횟수가 줄어들어 마감시간 내에 동기적 메시지 전송을 완료할 수 없는 경우가 발생한다. 따라서 지연되어 발생된 메시지는 긴급한 전송을 필요로 하지만 기존의 TTP는 긴급한 전송 방법을 지원하지 않는다. 노드에서의 메시지 지연 발생을 방지하는 것은 매우 어려운 일이므로 동적인 분산 실시간 시스템에서 보다 높은 동기적 메시지의 마감시간 보장성을 갖기 위해서는 시스템의 상태에 동적으로 대처하여 동기적 메시지를 전송하는 프로토콜이 필요하다.

본 논문에서는 분산 실시간 시스템의 동적인 상황에 맞추어 동기적 메시지를 전송함으로써 보다 높은 동기적 메시지의 마감시간 보장성을 갖는 TTP/C(timed token protocol with concession)를 제안한다. TTP/C는 시스템의 동기적 메시지가 긴급한 전송을 요구할 때 자신 또는 다른 노드가 가진 비동기적 메시지의 전송 대역폭과 다른 노드의 긴급하지 않은 동기적 메시지의 전송 대역폭을 양보받아 사용함으로써 긴급한 동기적 메시지의 마감시간을 보장한다. 물론 자신의 전송 대역폭을 양보하는 동기적 메시지의 마감시간은 TTP/C에 의해 보장된다. 본 논문에서는 TTP/C 기법을 제안하고 실험을 통해 TTP에 비해 높은 동기적 메시지의 마감시간 보장성을 가짐을 보인다.

본 논문의 구성은 다음과 같다. 2장에서 시스템 모델로써 네트워크 모델과 메시지 스트림 모델을 정의한다. 3장에서는 TTP/C를 제안하고 동기적 메시지에 대한 TTP/C의 마감시간 보장 특성을 보인다. 4장에서 모의 실험을 통하여 TTP와 제안한 TTP/C의 동기적 메시지에 대한 마감시간 보장성을 비교한다. 5장에서 결론 및 앞으로의 연구 방향을 기술한다.

2. 시스템 모델

2.1 네트워크 모델

네트워크 분석을 용이하게 하기 위해 G. Agrawal[1]의 가상 네트워크 모델(virtual network)을 사용한다. 실제 네트워크 상에서 각 노드는 한 개 이상의 동기적 메시지 스트림과 비동기적 메시지 집합을 갖는다. 가상 네트워크 모델을 이용하면 다수의 동기적 메시지 스트림을 갖는 실제 노드는 노드 당 한 개의 동기적 메시지 스트림을 갖는 여러 개의 가상노드로 변형된다. 예를 들어 2개의 동기적 메시지 스트림을 갖는 실제 노드는 한 노드에 한 개의 동기적 메시지 스트림을 갖는 2개의 가상 노드로 변형시킨다. 이렇게 n 개의 동기적 메시지 스트림을 갖는 네트워크를 실제 노드의 개수에 상관없이 n 개의 가상 노드로 변형함으로써 분석을 용이하게 할 수 있다. 실제 노드는 다수 개의 동기적 메시지 스트림과 비동기적 메시지 집합을 갖는데 실제 노드에서 파생된 가상 노드들 중에서 마지막 가상 노드만 추가적으로 비동기적 메시지 집합을 갖는다. 즉, 마지막 가상 노드를 제외한 나머지 가상 노드는 비동기적 메시지 집합을 가지지 않고 한 개의 동기적 메시지 스트림 만을 갖는다. 또한 같은 실제 노드에서 파생된 가상 노드들 사이에서 토큰 오버헤드는 존재하지 않는다. 토큰 오버헤드는 실제 노드 간에만 존재하므로 상위 실제 노드의 마지막 가상 노드와 해당 실제 노드의 처음 가상 노드 사이에만 존재한다.

2.2 메시지 스트림 모델

가상 노드는 한 개의 동기적 메시지 스트림과 비동기적 메시지 스트림 집합을 갖는다. 가상 노드 i 의 동기적 메시지 스트림 S_i 는 (P_i, C_i, D_i)로 표기한다. C_i 는 메시지의 크기로써 이 메시지를 전송하는데 필요한 최악전송시간이다. P_i 는 동기적 메시지 스트림의 주기이다. D_i 는 동기적 메시지의 상대적 마감시간이다. 시간 t 에 존재하는 동기적 메시지의 잔여 전송 시간을 $E_i(t)$ $(E_i(t) \le C_i)$라고 하고 마감시간까지 남은 시간을 $D_i(t)$ 라고 한다.

본 논문에서는 네트워크 시스템에 n 개의 동기적 메시지 스트림이 존재하고 n 개의 가상 노드가 존재하고 이 가상 노드들은 링 방식으로 연결되어 있다고 가정한다.

$$S = \{S_1, S_2, \ldots, S_n\}$$

$$S_i = (P_i, C_i, D_i)$$

메시지가 전송될 때 한 개 이상의 패킷으로 나누어져 전송되는데 모든 패킷의 마감시간은 메시지의 마감시간과 같고 만일 한 개의 패킷이라도 마감시간을 놓치거나 손실되면 마감시간을 맞추지 못한 것으로 가정한다.

토큰이 링을 따라 한번 회전하는데 걸리는 토큰 오버헤드를 τ 라고 하는데 이 τ 는 노드간의 지연시간, 버퍼의 오버헤드로 인한 지연시간, 매체 전달 시간, 각 노드가 토큰을 처리하는 시간 등을 포함한다. τ 는 의 $TTRT$ 시간에서 모든 동기적 메시지의 동기적 용량 H_i 의 합을 제외하고 남은 시간이다.

동기적 메시지 스트림 집합이 실행가능하기 위해 다음의 두 가지 제약사항을 만족해야 한다[1].

1) 프로토콜 제약사항(protocol constraints)

모든 노드의 동기적 용량 H_i 의 합은 $TTRT$ 에서 τ 를 뺀 값보다 클 수 없다. 즉,

$$\sum_{i=1}^{n} H_i \le TTRT - \tau \quad (1)$$

이다.

2) 마감시간 제약사항(deadline constraints)

모든 동기적 메시지는 마감시간 이전에 전송되어져야 한다. 동기적 대역폭 할당 기법에 의해 구간 $[t, t+D_i]$ 사이에서 동기적 메시지가 전송될 수 있도록 할당된 최소의 전송시간을 X_i 라고 한다. 동기적 메시지의 마감시간을 보장하기 위해 다음의 식을 만족해야 한다.

$$X_i \ge C_i \quad (2)$$

X_i 는 구간 $[t, t+D_i]$ 사이에서 노드 i 가 토큰을 받는 개수와 H_i 의 함수이다.

3. TTP/C

본 장에서는 먼저 TTP/C의 기본 개념을 살펴본다. 그리고 TTP/C가 전송하는 동기적 메시지의 상태를 $E_i(t)$와 $D_i(t)$에 따라 분류한다. 마지막으로 TTP/C의 동작을 자세히 설명하고 동기적 메시지의 마감시간 보장 측면에서 TTP/C의 특성을 살펴본다.

3.1 TTP/C의 기본적 개념

동기적 메시지 스트림은 각 주기마다 스트림의 실체(instance)인 메시지를 한 개씩 발생시킨다. 이론적으로 노드는 동기적 메시지 스트림의 각 주기의 시작시간에 맞추어 메시지를 발생해야 하지만 실제 시스템의 경우 계산 수행 또는 I/O 수행 등의 균형 잡히지 못한 자원 배분으로 인해 동기적 메시지를 각 주기의 시작 시간에 발생시키기 어렵다. 동기적 메시지가 주기의 시작시간에서 멀리 떨어져 발생될수록 그만큼 마감시간에 가까워지기 때문에 마감시간 보장을 위해 메시지의 전송은 긴급하게 이루어져야 한다. 동기적 메시지의 마감시간 보장성을 높이기 위해 네트워크 프로토콜은 이러한 동적인 상황에 맞추어 동기적 메시지를 전송해야 한다. 동기적 메시지는 주기의 시작시간 이후에 발생될 수도 있지만 주기의 시작시간 이전에 발생될 수도 있다. 그러나 일찍 발생한 메시지를 전송하면 burst 현상을 초래하게 된다[4]. 본 논문에서는 주기의 시작시간 이전에 메시지가 발생되더라도 즉시 전송하지 않고 전송을 미루어 다음 주기의 시작시간 이후 토큰을 받았을 때 메시지를 전송한다고 가정한다. 동기적 메시지 스트림의 각 주기의 시작시간과 해당주기에 발생하는 메시지의 실제 발생시간과의 차이를 발생 지연시간(generation delay)이라고 정의한다. 동기적 메시지의 발생시간이 해당 주기의 시작시간보다 빠르거나 같다면 발생 지연시간은 0 이다.

본 논문에서 제안하는 TTP/C는 분산 실시간 시스템의 동적인 상황에 적응하여 동기적 메시지를 전송하는 프로토콜이다. 이 프로토콜은 긴급함을 표시할 수 있는 토큰을 이용한다. 토큰을 받은 노드는 메시지 상태에 따라 적절한 전송 방법을 결정하여 동기적 메시지를 전송한다 TTP/C의 전송 방법은 다음의 두 가지 정보에 의해 결정된다.

1) 토큰을 받은 노드가 마감시간 보장을 위해 반드시 동기적 메시지를 보내야 하는가에 대한 동기적 메시지의 상태,
2) 이전 노드 중 긴급한 동기적 메시지를 가진 노드의 존재 여부.

TTP/C의 기본적인 개념은 비동기적 메시지의 전송 대역폭 또는 긴급하지 않은 동기적 메시지의 전송 대역

폭을 긴급한 동기적 메시지를 위해 양보함으로써 동기적 메시지의 마감시간 보장성을 높이는 것이다. TTP/C는 동기적 메시지의 마감시간을 보장하는 조건 하에 동기적 메시지의 전송 대역폭을 양보한다. 긴급한 메시지를 가지고 있는 노드는 다른 노드에게 자신의 긴급함을 나타내기 위해 토큰에 긴급함을 표시하여 전달한다. 긴급함이 표시된 토큰을 받은 노드는 이전 노드 중에 긴급한 동기적 메시지를 가진 노드가 존재함을 알고 비동기적 메시지를 전달하지 않고 긴급한 노드에게 비동기적 전송 대역폭을 양보한다. 만약 노드가 가지고 있는 동기적 메시지를 당장 전송하지 않아도 TTP/C에 의해 마감시간이 보장된다면 동기적 메시지를 위한 전송 대역폭도 양보한다.

TTP/C에서 메시지의 긴급함을 표시하기 위해 토큰에 urgent_counter 필드를 추가한다. Urgent_counter는 긴급한 메시지를 갖는 노드의 개수를 기록한다. 각 노드는 자신이 urgent_counter를 증가시켰는지 분별하기 위해 urgent라는 불린(boolean) 변수를 갖는다. 노드가 긴급한 동기적 메시지를 가진 경우 TTP/C 프로토콜은 urgent 변수를 TRUE 시키고 urgent_counter 필드를 1 만큼 증가시켜 다음 노드로 전달하여 긴급한 동기적 메시지가 있음을 알린다. urgent 변수가 TRUE인 노드가 토큰을 받았을 때 노드가 토큰의 긴급함을 해제하기 위해서는 urgent 변수를 FALSE시키고 urgent_counter를 1 만큼 감소시킨다. 논문 진행의 용이함을 위해 노드의 urgent 변수를 TRUE시키고 urgent_counter를 증가시켜 토큰을 전달하는 것을 노드가 긴급한 토큰을 전달한다라고 표현한다. 만약 단순히 토큰을 전달한다라고 표현하면 이전 노드에게 받은 토큰의 urgent_counter과 노드의 urgent를 변경하지 않고 단순하게 전달함을 의미한다. urgent_counter 필드가 1 이상인 토큰을 긴급한 토큰이라고 표현한다. 문장의 문맥이 명확하지 않은 경우 명시적으로 표현한다.

3.2 TTP/C에서 동기적 메시지의 상태 분류

본 절에서는 TTP/C를 제안하기 이전에 노드가 가지고 있는 동기적 메시지의 상태를 $E_i(t)$와 $D_i(t)$에 따라 분류한다.

동기적 용량 H_i 가 주어 졌을 때 동기적 메시지를 전송하기 위해 노드 i 가 받아야 하는 토큰의 최소 개수 δ_i 는 다음과 같다.

$$\delta_i = \lceil \frac{C_i}{H_i} \rceil$$

δ_i 는 1보다 크거나 같은 정수이다. H_i 는 노드 i 가 토큰을 받았을 때 전송할 수 있는 최대 전송 시간으로 동기적 대역폭 할당 기법에 의해 결정된다.

시간 t 에 받은 토큰을 포함하여 다음의 식을 만족하는 v_i 개수 만큼 토큰을 받아야만 메시지를 모두 전송할 수 있다.

$$v_i = \lceil \frac{E_i(t)}{H_i} \rceil$$

초기적으로 $E_i(t)=C_i$ 가 된다. $\lceil \frac{E_i(t)}{H_i} \rceil > \frac{E_i(t)}{H_i}$ 이면 v_i-1번은 H_i 만큼 전송하고 마지막 한번은 $E_i(t)-(v_i-1)\cdot H_i$ 만큼 전송하게 된다. 그러나 $\lceil \frac{E_i(t)}{H_i} \rceil = \frac{E_i(t)}{H_i}$ 이면 v_i 번 H_i 만큼 전송한다.

TTP/C에서 노드가 긴급한 토큰을 받으면 비동기적 메시지는 전송하지 않고 자신에게 할당된 동기적 대역폭 H_i 만큼 동기적 메시지를 전송한다. 노드가 긴급한 토큰을 전달하면 모든 노드는 할당된 동기적 용량 H_i 만큼만 사용하기 때문에 $TTRT$ 시간 이내에 다음 토큰을 받을 수 있다. 노드가 동기적 메시지의 전송을 완료할 때까지 긴급한 토큰을 계속 전달하는 경우 현재 받은 토큰을 제외하고 v_i-1번 더 토큰을 받을 수 있는 최대시간은 $(v_i-1)\cdot TTRT$이다. 시간 t 에 토큰을 받았을 때 동기적 메시지의 전송을 완료하기 위해 긴급한 토큰을 계속 전달할 경우 동기적 메시지의 전송을 완료할 수 있는 TTP/C의 최소전송완료시간 $W_{\min}(t)$은 $(v_i-1)\cdot TTRT + E_i(t)-(v_i-1)\cdot H_i$ 가 된다. 그림 1은 긴급한 토큰을 전달할 때 $W_{\min}(t)$의 예를 보여준다.

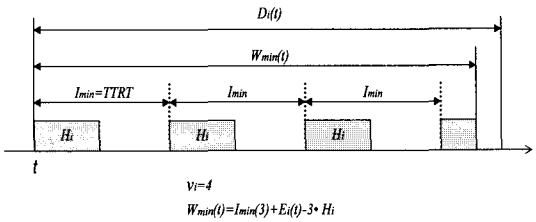


그림 1 긴급한 토큰을 계속 전달할 때 $W_{\min}(t)$의 예

반면 노드가 긴급한 토큰을 보내지 않고 단순히 토큰을 전달할 때 $2TTRT-H_i$ 시간 이내에 다음 토큰을 받을 수 있다[1]. 노드가 긴급하지 않은 토큰을 계속 전달한다고 할 때 현재 받은 토큰을 제외하고 v_i-1번 더 토큰을 받을 수 있는 최대시간은 $v_i\cdot TTRT-H_i$ 이다[1]. 시간 t 에 토큰을 받았을 때 동기적 메시지의 전송을 완료하기 위해 긴급하지 않은 토큰을 계속 전달할 경우 동기적 메시지의 전송을 완료할 수 있는 TTP/C의 최대전송완료시간 $W_{\max}(t)$

는 $v_i \cdot TTRT + E_i(t) - v_i \cdot H_i$ 가 된다.

위의 $W_{min}(t)$과 $W_{max}(t)$를 정리하면 다음과 같다.

$$W_{min}(t) = (v_i - 1) \cdot TTRT + E_i(t) - (v_i - 1) \cdot H_i$$
$$W_{max}(t) = v_i \cdot TTRT + E_i(t) - v_i \cdot H_i$$
$$\text{또는 } W_{max}(t) = W_{min}(t) + TTRT - H_i$$

따라서 긴급한 토큰을 계속 전달하는 경우 전송시간 C_i인 동기적 메시지의 전송을 완료할 수 있는 시간 W_{min}은 $(\delta_i - 1) \cdot TTRT + C_i - (\delta_i - 1) \cdot H_i$ 이다. 긴급한 토큰을 전달하지 않고 단순히 토큰을 전달하는 경우 전송시간 C_i 인 동기적 메시지의 전송을 완료할 수 있는 시간 W_{max}는 $\delta_i \cdot TTRT + C_i - \delta_i \cdot H_i$ 또는 $W_{min} + TTRT - H_i$ 이다.

W_{min}과 W_{max}의 값은 동기적 메시지의 최악전송시간 C_i 를 기준으로 하였기 때문에 전송을 완료할 수 있는 최악 시간을 나타낸다. 그러나 동기적 메시지의 평균 전송시간은 C_i 보다 작기 때문에 동기적 메시지의 전송을 완료하기 위해 받아야 하는 토큰의 개수는 δ_i 보다 작을 수 있으며 전송 완료시간도 W_{min}과 W_{max}보다 작을 수 있다. 예를 들어 MPEG 비디오 스트림의 경우 압축 알고리즘에 발생되는 동기적 메시지의 크기는 일정하지 않다. 그러므로 실제 발생되는 메시지의 크기가 작은 경우 δ_i 보다 적은 수의 토큰으로 동기적 메시지의 전송이 가능하다. TTP/C는 실제 생성된 동기적 메시지의 정보에 따라 전송방법을 결정하므로 효율적으로 네트워크의 전송 대역폭을 사용할 수 있다.

시간 t 에 토큰을 받았을 때 노드는 이전 노드 중 긴급한 동기적 메시지를 가진 노드의 존재 유무에 따라 동기적 메시지를 전송할 것인지 전송하지 않을 것인지 결정하게 된다. 동기적 메시지를 전송하였는지 하지 않았는지에 따라 다음 토큰을 받는 시간 t'에서의 $W_{min}(t')$과 $W_{max}(t')$는 다음과 같이 변한다.

- 시간 t 에 메시지를 전송했을 때

$$W_{min}(t') = W_{min}(t) - TTRT$$
$$W_{max}(t') = W_{max}(t) - TTRT$$

- 시간 t 에 메시지를 전송하지 않았을 때

$$W_{min}(t') = W_{min}(t)$$
$$W_{max}(t') = W_{max}(t)$$

시간 t 에 노드가 토큰을 받았을 때 동기적 메시지가 TTP/C에 의해 마감시간 내에 전송되기 위한 조건은 다음과 같다.

$$D_i(t) \geq W_{min}(t)$$

동기적 메시지가 위의 식을 만족하면 TTP/C 프로토콜은 긴급한 토큰을 전달함으로써 동기적 메시지의 마감시간을 보장할 수 있다.

시간 t 에 노드가 토큰을 받았을 때 동기적 메시지의 상태를 다음의 네 가지 상태로 나눈다.

- 상태 1 : $W_{max}(t) + TTRT \leq D_i(t)$
- 상태 2 : $W_{max}(t) \leq D_i(t) < W_{max}(t) + TTRT$
- 상태 3 : $W_{min}(t) \leq D_i(t) < W_{max}(t)$
- 상태 4 : $D_i(t) < W_{min}(t)$

상태 1 또는 2에 있는 동기적 메시지는 긴급하지 않은 메시지이다. 상태 3 또는 4에 있는 동기적 메시지는 긴급한 메시지이다. 동기적 메시지의 마감시간이 TTP/C 프로토콜에 의해 보장되기 위해서 동기적 메시지는 상태 1, 2, 또는 3에 존재해야 한다.

3.3 TTP/C

TTP/C에서 사용하는 각 매개변수는 다음과 같다.

- $TTRT$: 한번의 토큰 회전동안 모든 노드가 자신에게 할당된 동기적 대역폭을 전송했을 때의 최악 토큰순환 시간이다.
- H_i : 노드가 토큰을 받았을 때 동기적 메시지를 전송할 수 있는 최대 전송 대역폭이다. 이 값은 동기적 대역폭 할당 기법에 의해 결정된다.
- 노드 i 의 토큰 회전 타이머(token rotation timer) TRT_i : 이 타이머는 $TTRT$ 로 초기화 되고 0이 될 때까지 계속 감소된다.
- 노드 i 의 토큰 소유 타이머(token holding timer) THT_i : 이것은 노드 i 가 비동기적 메시지를 전송할 수 있는 시간을 조절하기 위해 사용되는 타이머이다. 이 타이머는 노드가 비동기적 메시지를 전송하는 동안 감소된다.
- 노드 i 의 LC_i (late counter) : 이것은 노드 i 가 최근에 토큰을 받은 시간 이후에 TRT_i 가 만료됐던 횟수를 기록하는 카운터이다.
- urgent : 노드에 긴급한 메시지가 있음을 가리키는 변수, 초기값은 FALSE 이다.
- urgent_counter : 토큰이 링을 따라 돌면서 긴급한 메시지를 가진 노드의 개수를 기록하는 필드이다. 긴급한 메시지를 가진 노드는 이 필드 값을 증가시킨다.

TTP/C는 TTP와 같이 기본적으로 다음과 같은 수행을 한다.

- $THT_i \leftarrow 0$, $LC_i \leftarrow 0$, $TRT_i \leftarrow TTRT$ 를 수행하여 각 노드를 초기화 한다.
- 초기화 이후 TRT_i 는 항상 감소된다.
- 만일 노드에 토큰이 도착하기 전에 TRT_i 가 0이 되

면 TRT_i 를 $TTRT$ 값으로 고치고 LC_i 를 증가시켜 토큰의 도착이 늦었음을 표시한다. LC_i 가 1을 초과하게 되면 링 복원 프로세스(ring recovery process)가 시작된다.

- 토큰을 받은 노드는 각 노드의 타이머에 조절되어 메시지를 전송한다. 전송되고 있는 패킷은 전송을 마칠 때까지 중지될 수 없다. 토큰을 받았을 때 $LC_i = 0$ 이면 토큰은 노드에 일찍 도착한 것이고 $LC_i \geq 1$ 이면 토큰은 노드에 늦게 도착한 것이다.
- 토큰이 일찍 도착했을 때 즉, $LC_i = 0$ 일 때는 $THT_i \leftarrow TRT_i$, $TRT_i \leftarrow TTRT$ 를 수행한다.
- 토큰이 늦게 도착했을 때 즉, $LC_i \geq 1$ 일 때는 $LC_i \leftarrow 0$을 수행한다.

TTP/C의 pseudo-code는 그림 2와 같다. 그림 2에서 '현재 메시지'는 노드가 토큰을 받았을 때 전송을 시작해야 하거나 전송을 완료하지 않은 메시지이다. '다음 메시지'는 다음에 준비될 동기적 메시지이다.

TTP/C 프로토콜은 토큰을 받았을 때 전송할 동기적 메시지가 존재한다면 다음의 네 가지 중 하나를 수행한다.

- 메시지의 상태가 1 또는 2이고 이전 노드 중 긴급한 메시지를 가진 노드가 존재하지 않으면 동기적 메시지를 전송하고 가능하면 비동기적 메시지를 전송한 후 토큰을 전달한다. (urgent_counter가 0일 경우 이전 노드 중 긴급한 메시지를 가진 노드는 없다. 그리고 urgent_counter가 1이고 urgent가 TRUE일 때 토큰을 가진 노드가 유일하게 긴급한 메시지를 가진 노드이므로 이전 노드 중 긴급한 메시지를 가진 노드는 없다.)
- 메시지의 상태가 1이고 이전 노드 중 긴급한 메시지를 가진 노드가 존재하면 동기적, 비동기적 메시지를 전송하지 않고 토큰을 전달한다. 이 경우 긴급한 토큰을 전송하지 않더라도 TTP/C는 전송대역폭을 양보한 동기적 메시지의 마감시간을 보장할 수 있다.
- 메시지의 상태가 2이고 이전 노드 중 긴급한 메시지를 가진 노드가 존재하면 동기적, 비동기적 메시지를 전송하지 않고 urgent를 TRUE시키고 urgent_counter를 증가시켜 토큰을 전달한다(이미 urgent가 TRUE이면 urgent_counter를 증가시키지 않는다). 이 경우 전송 대역폭을 양보한 동기적 메시지의 마감시간을 보장하기 위해 긴급한 토큰을 전달해야 한다.
- 메시지의 상태가 3 또는 4일 때 동기적 메시지만 전송하고 urgent를 TRUE시키고 urgent_counter를 증가시켜 토큰을 전달한다. 상태 4에 있는 동기적 메시지의 마감시간은 충분한 전송 대역폭을 양보 받으면 보장될 수 있다. 그러므로 상태 4에 있다고 해서 전송을 포기하는 것이 아니라 일단 동기적 메시지를 전송한 후 긴급한 토큰을 전달하여 다른 노드가 충분한 전송 대역폭을 양보해 주기를 요청한다.

```
Loop
  토큰을 받음;
  if( 전송할 메시지가 없음 ) {
    if( 다음 메시지의 상태가 1)
      if( urgent == TRUE )
        urgent = FALSE, urgent_counter --;
      else
        if( urgent == FALSE )
          urgent = TRUE, urgent_counter++;
  } else {
    if( 현재 메시지의 상태가 1 또는 2) {
      if( ((urgent == TRUE) and (urgent_counter == 1)) or (urgent_cou  nter == 0)) {
        최대 H_i 시간 만큼 동기적 메시지를 전송;
        if( (현재 메시지 전송이 완료되지 않음) or ((현재 메시지의 전송 완료 ) and (다음 메시지의 상태가 1))) {
          if( urgent == TRUE )
            urgent = FALSE, urgent_counter --;
          if( 토큰이 일찍 도착 )
            THT_i가 만료될 때까지 비동기적 메시지를 전송;
        } else // 현재 메시지의 전송이 완료되고 다음 메시지의 상태가 2, 3, 또는 4 일 때
          if( urgent == FALSE )
            urgent = TRUE, urgent_counter++;
      } else if( ( 메시지의 상태가 2) and ( urgent==FALSE ) )
        urgent = TRUE, urgent_counter++;
    } else {
      최대 H_i 시간만큼 동기적 메시지를 전송;
      if((현재 메시지의 전송 완료) and (다음 메시지의 상태가 1)) {
        if( urgent == TRUE )
          urgent = FALSE, urgent_counter --;
      } else // 현재 메시지의 전송이 완료되지 않았거나 다음 메시지의 상태가 2, 3, 또는 4일 때
        if( urgent == FALSE )
          urgent = TRUE, urgent_counter++;
    }
  }
  토큰을 다음 노드로 전송;
Forever;
```

그림 2 TTP/C 프로토콜

3.4 TTP/C의 마감시간 보장성

TTP/C는 현재 전송 중인 동기적 메시지의 마감시간을 보장하기 위해 토큰을 처리할 뿐만 아니라 다음 주기에 준비될 동기적 메시지를 고려하여 토큰을 처리한다. 현재 전송 중인 동기적 메시지를 전송 완료하였을 때 단순히 토큰을 전달하지 않고 다음 주기에 준비될 동기적 메시지의 상태를 고려하여 토큰을 적절히 처리하여 전달한다. 그리고 노드가 토큰을 받았을 때 전송할 동기적 메시지가 존재하지 않더라도 준비될 동기적 메시지를 고려하여 토큰을 처리하여 다음 노드로 전달한다. 토큰이 도착하였을 때 전송할 동기적 메시지가 존재하지 않는 경우는 다음과 같다.

(1) 메시지의 주기가 이미 시작되었지만 노드가 아직 전송할 동기적 메시지를 준비하지 못한 경우,

(2) 현재 주기의 동기적 메시지가 모두 전송되었고, 아직 다음 주기가 되지 않은 경우

다음에 준비될 동기적 메시지가 발생될 실제 시간은 알 수 없지만 동기적 메시지 스트림 정보를 통해 준비될 동기적 메시지의 v_i 와 마감시간을 알 수 있다. 즉, W_{min} , W_{max} 와 $D_i(t)$를 알 수 있다. 이 정보를 통해 TTP/C는 다음에 준

비될 동기적 메시지의 마감시간 보장을 위해 토큰에 긴급함을 표시해야 하는지를 결정한다. 토큰을 받은 시간을 t라고 할 때 긴급함을 표시하지 않고 토큰을 전달하게 되면 다음 토큰의 최악도착시간은 $t+2TTRT-H_i$ 가 된다. 그러므로 $W_{\min}(t)+2TTRT-H_i \le D_i(t)$을 만족한다면 즉, 다음에 준비될 동기적 메시지가 상태 1에 있다면 긴급함을 표시하지 않고 토큰을 전달한다. 그러나 $W_{\min}(t)+2TTRT-H_i > D_i(t)$인 경우 즉, 다음에 준비될 동기적 메시지의 상태가 1이 아닌 경우에는 긴급함을 토큰에 표시하여 전달한다. 이와 같이 TTP/C 프로토콜은 현재 전송 중인 동기적 메시지 뿐만 아니라 다음에 준비될 동기적 메시지를 위해 토큰을 적절히 처리함으로써 보다 높은 동기적 메시지 마감시간 보장성을 갖는다.

상태 1, 2, 또는 3에 있는 동기적 메시지의 마감시간이 TTP/C에 의해 보장되기 위해서 이 동기적 메시지는 결코 TTP/C에 의해 상태 4로 전이되지 않고 상태 1, 2, 또는 3에 존재해야 한다. 다음 정리 1은 상태 1, 2, 또는 3에 있는 동기적 메시지에 대해서 TTP/C는 항상 마감시간을 보장할 수 있음을 보여 준다.

정리 1. 상태 1, 2, 또는 3에 있는 동기적 메시지의 마감시간은 TTP/C에 의해 보장된다.

증명 :

시간 t 에 노드에 토큰이 도착했을 때 동기적 메시지가 $D_i(t) \ge W_{\min}(t)$를 만족하면 동기적 메시지는 상태 1, 2, 또는 3에 있다. TTP/C가 동기적 메시지의 상태에 따라 적절한 수행을 한 후에 다음 토큰을 받는 최악시간 t'에 동기적 메시지의 상태가 최소한 상태 3에 있기 위한 다음의 식 (1)을 만족한다면 TTP/C는 최소한 상태 3에 있는 동기적 메시지의 마감시간을 보장해 줄 수 있다.

$$D_i(t') \ge W_{\min}(t') \tag{1}$$

시간 t 에 노드에 토큰이 도착했을 때 TTP/C의 수행은 다음의 네 가지 경우로 나뉜다.

경우 1) 동기적 메시지가 상태 1 또는 2에 있고 이전 노드에 긴급한 메시지를 가진 노드가 존재하지 않을 때

이 경우 TTP/C 프로토콜은 동기적 메시지를 전송하고 토큰이 일찍 도착한 경우라면 비동기적 메시지를 전송한다. TTP/C 프로토콜에 의해 t'는 $t+2TTRT-H_i$이다. $D_i(t')=D_i(t)-2TTRT+H_i$ 이다. 노드는 H_i 만큼 동기적 메시지를 전송하기 때문에 시간 t'에 $E_i(t')=E_i(t)-H_i$, $W_{\min}(t')=W_{\min}(t)-TTRT$, $W_{\max}(t')=W_{\max}(t)-TTRT$, $W_{\max}(t')=W_{\min}(t')+TTRT-H_i$ 이다.

상태 1과 상태 2를 만족하는 식 $D_i(t) \ge W_{\max}(t)$은 식 (2)와 같이 변경된다.

$$\begin{aligned} D_i(t') &\ge W_{\max}(t)-2TTRT+H_i \\ &\ge W_{\max}(t')-TTRT+H_i \\ &\ge W_{\min}(t') \end{aligned} \tag{2}$$

이 경우 식 (2)에 의해서 식 (1)은 만족한다.

경우 2) 동기적 메시지가 상태 1에 있고 이전 노드에 긴급한 메시지를 가진 노드가 존재할 경우

이 경우 TTP/C는 동기적, 비동기적 메시지를 전송하지 않고 단순히 토큰을 전달한다. 긴급한 토큰을 전달하지 않고 자신도 동기적 메시지를 H_i 만큼 전송하지 않기 때문에 TTP/C에 의해 t'는 $t+2TTRT-H_i$ 이다. $D_i(t')=D_i(t)-2TTRT+H_i$ 이다. 동기적 메시지를 H_i 만큼 전송하지 않았기 때문에 $E_i(t')=E_i(t)$, $W_{\min}(t')=W_{\min}(t)$, $W_{\max}(t')=W_{\max}(t)$, $W_{\max}(t')=W_{\min}(t')+TTRT-H_i$이다.

상태 1의 식 $D_i(t) \ge W_{\max}(t)+TTRT$는 식 (3)과 같이 변경된다.

$$\begin{aligned} D_i(t') &\ge W_{\max}(t)-TTRT+H_i \\ &\ge W_{\max}(t')-TTRT+H_i \\ &\ge W_{\min}(t') \end{aligned} \tag{3}$$

이 경우 식 (3)에 의해서 식 (1)은 만족한다.

경우 3) 동기적 메시지가 상태 2에 있고 이전 노드에 긴급한 메시지를 가진 노드가 존재할 경우

이 경우 TTP/C는 동기적, 비동기적 메시지를 전송하지 않고 urgent 필드를 TRUE로 바꾸고 urgent_counter를 증가시켜 다음 노드로 토큰을 전달한다. 이 토큰을 받는 다음 노드들은 긴급한 토큰을 받기 때문에 동기적 메시지만 전송하게 된다. 또한 자신도 동기적 메시지를 H_i 만큼 전송하지 않기 때문에 TTP/C에 의해 t'는 $t+TTRT-H_i$ 이다. $D_i(t')=D_i(t)-TTRT+H_i$ 이고 $E_i(t')=E_i(t)$이다. 동기적 메시지를 H_i 만큼 전송하지 않았기 때문에 시간 t'에 $W_{\min}(t')=W_{\min}(t)$, $W_{\max}(t')=W_{\max}(t)$, $W_{\max}(t')=W_{\min}(t')+TTRT-H_i$ 이다. 식 $D_i(t) \ge W_{\max}(t)$은 식 (4)와 같이 변경된다.

$$\begin{aligned} D_i(t') &\ge W_{\max}(t)-TTRT+H_i \\ &\ge W_{\max}(t')-TTRT+H_i \\ &\ge W_{\min}(t') \end{aligned} \tag{4}$$

이 경우 식 (4)에 의해서 식 (1)은 만족한다.

경우 4) 동기적 메시지가 상태 3에 있을 때

동기적 메시지가 상태 3에 있을 때 TTP/C는 동기적 메시지만 전송한 후 긴급한 토큰을 전달한다. 이 토큰을 받는 모든 노드는 긴급한 토큰을 받기 때문에 동기적 메시

지만 전송하게 된다. TTP/C에 의해 t'는 $t+TTRT$ 이다. 또한 $D_i(t')=D_i(t)-TTRT$ 이다. 동기적 메시지를 H_i 만큼 전송하였기 때문에 시간 t'에 $E_i(t')=E_i(t)-H_i$, $W_{\min}(t')=W_{\min}(t)-TTRT$, $W_{\max}(t')=W_{\max}(t)-TTRT$, $W_{max}(t')=W_{min}(t')+TTRT-H_i$이다. 상태 3의 식 $D_i(t)\geq W_{\min}(t)$는 식 (5)과 같이 변경된다.

$$D_i(t') \geq W_{\min}(t)-TTRT \geq W_{min}(t') \quad (5)$$

이 경우 식 (5)에 의해서 식 (1)은 만족한다.

위의 네 가지 경우에서 살펴본 바와 같이 상태 1, 2, 또는 3에 있는 동기적 메시지는 TTP/C에 의해 결코 상태 4로 전이되지 않는다. 그러므로 상태 1, 2, 또는 3에 있는 동기적 메시지의 마감시간은 TTP/C 프로토콜에 의해서 보장된다.■

상태가 4인 동기적 메시지는 자신의 비동기적 메시지의 전송 대역폭을 양보하거나 다른 노드가 전송 대역폭을 양보해 주지 않으면 마감시간 내에 전송 될 수 없다. 그러나 자신의 비동기적 메시지의 전송 대역폭을 양보하거나 다른 노드가 전송 대역폭을 양보해 주어 충분하게 전송 대역폭을 양보 받으면 이 상태 4에 있는 동기적 메시지는 상태 1, 2, 또는 3으로 전이할 수 있다. 만약 동기적 메시지가 상태 1, 2, 또는 3으로 일단 전이되기만 하면 정리 1에 의해 이 동기적 메시지는 TTP/C에 의해 마감시간 내에 전송된다.

4. 성능 평가

본 장에서는 모의 실험을 통하여 동기적 메시지에 대한 TTP와 TTP/C의 마감시간 보장성을 비교한다. 비교 척도로는 노드의 개수의 변화와 동기적 메시지의 발생 지연시간 변화에 따라 동기적 메시지의 마감시간을 놓치는 비율을 사용하였다.

실험에서 쓰이는 동기적 메시지 스트림으로 MPEG 표준으로 압축된 디지털 비디오 신호를 사용하였다. 각 MPEG 비디오 채널은 평균 1.5 Mbps의 신호 비율을 가지며 초당 30 프레임의 비율로 전송된다. 프레임의 발생 주기는 33 msec이다. 각 비디오 프레임은 다음 프레임이 발생하기 이전에 전송되어야 한다. MPEG 압축 알고리즘은 8개의 프레임마다 한 개의 I(Intracoded) 프레임을 발생시키는데 이 프레임은 다른 프레임에 비해서 평균 3배 정도 큰 크기를 갖는다. 본 실험에서 I 프레임의 크기는 100 Kbits ~ 150 Kbits사이에서, I 프레임이 아닌 다른 프레임의 크기는 25 Kbits ~ 75 Kbits사이에서 uniform distribution을 갖는다고 가정한다[5]. 동기적 메시지의 마감시간은 다음 주기의 동기적 메시지의 발생시간과 같다고 가정한다. 각 프로토콜은 동기적 메시지가 마감시간 내에 전송하지 못하면 전송을 포기하고 동기적 메시지를 버린다고 가정한다. 비동기적 메시지는 항상 충분하게 준비되어 있어서 노드가 각 프로토콜에 의해서 전송할 수 있는 양만큼 항상 전송할 수 있다고 가정한다.

TTP와 TTP/C에서 사용되는 $TTRT$ 와 H_i 는 같은 기법을 통해 정해 진다. $TTRT$ 는 $\frac{P_{\min}}{2}$로 16.5 msec라고 정한다. 동기적 대역폭 할당 기법은 지역 동기적 대역폭 할당 기법인 비례적 할당 기법(proportional allocation scheme)을 사용한다[1]. 비례적 할당 기법은 동기적 용량 H_i 를 $\frac{C_i}{P_i}(TTRT-\tau)$로 할당한다.

본 실험에서 네트워크 속도는 100 Mbps이고 τ 는 노드의 개수에 비례한다고 가정한다. 노드의 개수가 n 일 때 τ 는 $n\cdot 11\mu$sec 이다. 이러한 가정에서 프로토콜 제약사항인 $\sum_{i=1}^{n} H_i \leq TTRT-\tau$ 를 만족하는 노드의 최대 개수는 이론적으로 22개가 된다. 노드의 수가 23개 이상이면 $\sum_{i=0}^{23}\frac{1.5msec}{33msec}(TTRT-\tau) > (TTRT-\tau)$를 만족하게 되어 프로토콜 제약사항을 만족하지 못하게 되므로 TTP와 TTP/C는 동기적 메시지의 마감시간을 보장할 수 없게 된다. 실험에서 사용한 노드의 개수는 1개부터 30개로 변화를 주어 실험하였다.

본 실험에서는 메시지의 발생 지연시간을 변화시키면서 실험 하였다. 동기적 메시지 스트림의 발생은 normal distribution을 갖는다고 가정한다. 표준편차로는 주기에 대한 비율로 정하였으며 평균은 주기의 시작시간으로부터 '주기×표준편차'가 지난 발생 지연시간으로 하였다. 예를 들어 주기 33msec의 '주기×10%'의 표준편차를 갖는 동기적 메시지 스트림은 주기의 시작시간으로부터 3.3 msec 지난 발생 지연시간을 평균으로 하고 3.3을 표준편차로 갖는 normal distribution을 갖는다. 실험 시간은 MPEG 비디오 스트림이 5분간 전송되는 것으로 하였다. 5분 동안 각 노드는 약 9,900개의 동기적 메시지를 발생시킨다.

그림 3, 그림 4는 노드의 개수의 변화에 따라 동기적 메시지의 I 프레임이 마감시간을 놓치는 비율을 보인다. I 프레임이 다른 프레임보다 중요한 이유는 I 프레임이 아닌 다른 프레임의 압축을 풀기 위해서 I 프레임의 정보가 필요하기 때문이다. 그리고 I 프레임은 상대적으로 큰 크기를 갖기 때문에 다른 프레임에 비해 마감시간을 놓칠 확률이 크다. 이런 이유로 인해 본 실험에서는 I 프레임 만을 고려한다. 그림 3, 그림 4의 실험에서 동기적 메시지 스트림의 발생은 주기의 시작시간에서 각각 '주기×0%', '주

기×5%', '주기×10%', '주기×20%'만큼 지난 발생 지연시간을 동기적 메시지의 발생 평균시간으로 갖고 표준편차를 '주기×0%', '주기×5%', '주기×10%', '주기×20%'로 갖는 normal distribution을 갖는다.

그림 3에서 볼 수 있듯이 발생 지연시간이 0%일 때 TTP는 노드가 12개부터 I 프레임의 마감시간을 놓치기 시작한다. TTP는 주기의 시작시간 바로 전에 토큰이 도착했을 때 노드에 전송할 동기적 메시지가 없으면 비동기적 메시지만 전송하고 토큰을 다음 노드에게 전달하기 때문에 다음 번에 토큰이 도착하면 이미 동기적 메시지의 마감시간을 맞출 수 없게 되는 경우가 발생한다. 그러나 TTP/C는 다음 주기에 준비될 동기적 메시지를 고려하여 토큰을 처리하여 전달하기 때문에 TTP보다 높은 I 프레임의 마감시간 보장성을 갖는다. 노드의 수가 23개 이상이더라도 TTP/C는 TTP보다 여전히 좋은 성능을 보인다. TTP/C에서 긴급한 동기적 메시지를 가진 노드는 자신 또는 다른 노드의 비동기적 메시지의 전송 대역폭 뿐만 아니라 급하지 않은 동기적 메시지를 가진 노드의 전송 대역폭까지 양보 받아 사용할 수 있기 때문에 TTP보다 높은 마감시간 보장성을 갖는다. 노드의 수가 23개 이상일 때 TTP/C가 I 프레임의 마감시간을 맞추지 못하는 이유는 긴급한 토큰을 전달한다 하더라도 23개 이상의 노드가 자신이 할당 받은 H_i 를 사용하기 때문에 각 노드가 연속된 토큰을 받을 수 있는 최악시간이 $TTRT$ 를 초과하기 때문이다.

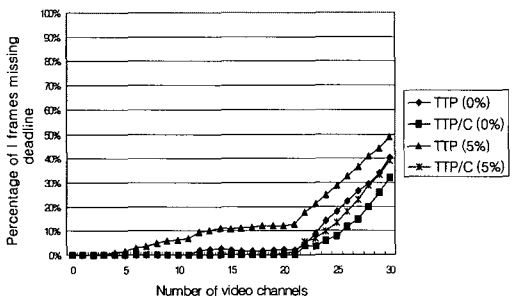


그림 3 발생 지연시간이 주기의 0%, 5%일 때 노드 개수의 변화에 따른 I 프레임이 마감시간을 놓치는 비율

그림 3, 그림 4에서 볼 수 있듯이 발생 지연시간이 길어질수록 I 프레임이 마감시간을 놓치는 비율은 증가한다. 두 프로토콜이 I 프레임의 마감시간을 맞추지 못하는 이유는 ① 동기적 메시지가 마감시간을 보장할 수 없을 정도로

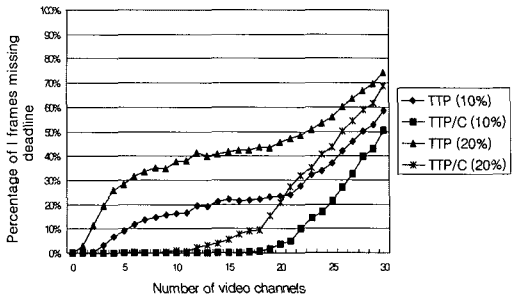
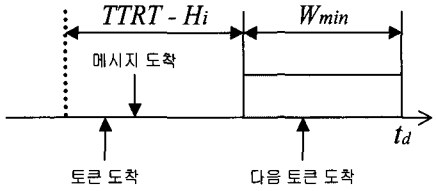


그림 4 발생 지연시간이 주기의 10%, 20%일 때 노드 개수의 변화에 따른 I 프레임이 마감시간을 놓치는 비율

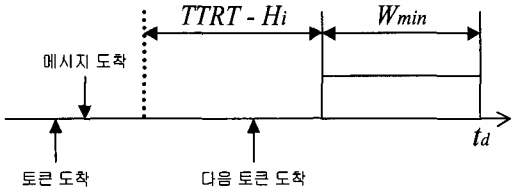
늦게 발생했거나 ② 동기적 메시지는 마감시간을 보장할 수 있도록 발생했지만 동기적 메시지가 발생하기 전에 토큰을 받았기 때문에 다음 노드로 토큰을 그냥 전달하였고 다음 번에 토큰을 받았을 때에는 이미 동기적 메시지의 마감시간을 보장할 수 없게 되기 때문이다. TTP는 위의 모든 이유에 의해 I 프레임의 마감시간을 놓치게 된다. TTP/C는 다음의 이유에 의해 동기적 메시지의 마감시간을 놓친다. ①의 경우 동기적 메시지의 마감시간을 t_d 라고 할 때 메시지가 $t_d - W_{\min}$ 시간 이후에 발생한 경우이다. ②의 경우는 그림 5의 (a)와 같이 구간 $(t_d - W_{\min} - TTRT + H_i,\ t_d - W_{\min})$사이에서 동기적 메시지가 토큰의 도착시간보다 늦게 발생할 경우이다. 예를 들어 $0 < \varepsilon < \varepsilon' < TTRT$ 를 만족하는 ε와 ε'가 있을 때 토큰이 시간 $t_d - W_{\min} - TTRT + H_i + \varepsilon$에 도착하고 동기적 메시지가 $t_d - W_{\min} - TTRT + H_i + \varepsilon'$에 발생한다고 한다. 토큰이 도착했을 때 전송할 동기적 메시지가 없기 때문에 TTP/C는 규칙에 의해 긴급한 토큰을 전달한다. 노드가 다음 토큰을 받는 최악시간은 $t_d - W_{\min} + \varepsilon'$이므로 이런 동기적 메시지의 마감시간은 TTP/C 프로토콜에 의해 보장되지 못한다. 그러나 그림 5의 (b)와 같이 시간 $t_d - W_{\min} - TTRT + H_i$ 또는 이전에 동기적 메시지가 준비되어 있으면 TTP/C에 의해 노드는 구간 $(t_d - W_{\min} - TTRT + H_i,\ t_d - W_{\min})$사이에서 다음 토큰을 받을 수 있으므로 이런 동기적 메시지의 마감시간은 TTP/C에 의해 보장된다.

그림 6은 노드의 개수가 각각 10, 21, 23개 일 때 발생 지연시간이 길어짐에 따라 I 프레임이 마감시간을 놓치는 비율의 변화를 보인다. 위에서 설명한 바와 같이 발생 지연시간이 길어질수록 I 프레임의 마감시간 보장성은 떨어진다. 그러나 같은 동기적 메시지의 발생에 대해서

TTP/C는 TTP보다 높은 마감시간 보장성을 갖는다.



(a)



(b)

그림 5 TTP/C에 의해 마감시간 보장되기 위한 동기적 메시지의 발생

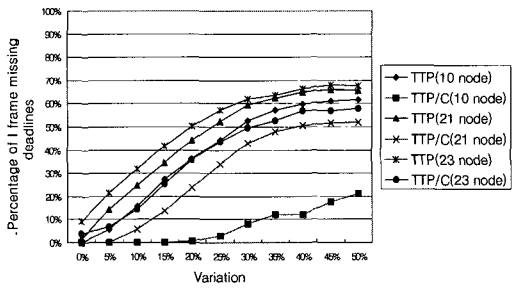


그림 6 노드의 개수가 10, 21, 23일 때 메시지의 발생 지연시간의 변화에 따른 I 프레임이 마감시간을 놓치는 비율

실험을 통하여 TTP/C는 TTP보다 높은 동기적 메시지의 마감시간 보장성을 가짐을 알 수 있었지만 반면에 매우 적은 비동기적 메시지의 처리율을 가짐을 알 수 있었다. 이는 노드가 긴급한 토큰을 받으면 비동기적 메시지를 전송하지 않기 때문이다. 노드가 비동기적 메시지를 전송하지 못하는 경우는 이전 노드 중 긴급한 동기적 메시지를 가지고 있는 노드가 존재하거나 자신이 긴급한 동기적 메시지를 가지고 있는 경우이다. 이 때 노드가 비동기적 메시지를 전송하게 되면 긴급한 메시지는 마감시간을 놓치게 될 수 있으므로 동기적 메시지의 마감시간을 보장하기 위해 비동기적 메시지를 전송하지 않고 양보해야 한다. 긴급한 동기적 메시지를 위해 이전의 노드들이 충분하게 전송 대역폭을 양보해 주었다 할지라도 긴급한 토큰을 받은 노드는 이러한 사실을 알지 못하기 때문에 토큰을 받은 노드는 자신의 전송 대역폭을 양보해 주어야 한다. TTP/C는 동기적 메시지의 마감시간 보장을 위해서 비동기적 메시지의 전송 대역폭을 사용하기 때문에 낮은 비동기적 메시지 처리율은 높은 동기적 메시지의 마감시간 보장이라는 측면에서 충분히 보상될 수 있다.

Hamdaoui[9]는 비동기적 메시지의 평균 응답시간을 줄이기 위해 동기적 메시지의 마감시간을 어기지 않는 범위에서 비동기적 메시지를 동기적 메시지보다 먼저 전송하는 개념을 사용하였다. 그러나 실험을 통해 네트워크의 링을 따라 전달되는 토큰의 urgent_counter 값이 대부분의 시간 동안 1이상임을 알 수 있었다. 그러므로 TTP/C에서 비동기적 메시지의 처리율을 높이기 위해 Hamdaoui의 방법을 응용하는 것은 비동기적 메시지 전송에 대해 큰 성능 향상을 가져오지 못한다.

5. 결 론

본 논문에서는 동기적 메시지의 발생 시간이 동적으로 변하는 분산 실시간 시스템에서 보다 높은 동기적 메시지의 마감시간 보장성을 제공하는 TTP/C를 제안하였다. TTP/C는 비동기적 메시지의 전송 대역폭이나 긴급하지 않은 동기적 메시지의 전송 대역폭을 양보 받아 사용함으로써 긴급한 동기적 메시지의 마감시간을 보장한다. TTP/C는 TTP와 달리 발생된 동기적 메시지의 상태에 따라 적응성 있게 메시지를 전송하므로 보다 효율적으로 네트워크의 전송 대역폭을 활용한다.

TTP에서 노드가 토큰을 받을 때 단순히 할당된 동기적 용량 만큼 동기적 메시지를 전송하고 가능하면 비동기적 메시지를 전송하는 프로토콜 오버헤드를 갖는다. 그러나 TTP/C는 TTP가 갖는 프로토콜 오버헤드 외에 전송할 동기적 메시지의 상태를 판별하고 토큰을 처리하는 프로토콜 오버헤드를 추가적으로 갖는다. 그러나 전송하기 위해 준비된 동기적 메시지의 마감시간은 고정되어 있으며 동기적 메시지의 $W_{min}(t)$와 $W_{max}(t)$는 전송 유무에 따라 상수적으로 감소하거나 변하지 않기 때문에 그 오버헤드는 매우 작다.

TTP/C는 동적인 동기적 메시지에 대한 높은 마감시간 보장성을 갖지만 낮은 비동기적 메시지 처리율을 갖는다. 그러므로 TTP/C와 같은 동기적 메시지의 마감시간 보장

성을 유지하면서 높은 비동기적 메시지 처리율을 갖는 프로토콜의 연구가 필요하다.

참 고 문 헌

[1] G. Agrawal, B. Chen, W. Zhao, and S. Davari, Guaranteeing synchronous message deadlines with the timed token protocol, In *Proceedings of IEEE International Conference on Distributed Computing Systems*, IEEE, June 1992

[2] Sijing Zhang and Alan Burns, An Optimal Synchronous Bandwidth Allocation Scheme for Guaranteeing Synchronous Message Deadlines with the Timed-token MAC Protocol, *IEEE/ACM Transactions on Networking*, vol. 3, no. 6, pp. 729 741, December 1995

[3] S.M. Yang, et al., A Protocol for Real-Time Message Scheduling in LAN/MAN, *Hawaii Int'l Conf. On System Sciences*, pp. 613-621, 1993

[4] D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time communication in multi-hop networks," *IEEE Transaction on Parallel and Distributed Systems*, 5(10):1044-1056, October 1994

[5] Kang G. Shin and Qin Zheng, FDDI-M: A Scheme to Double FDDI's Ability of Supporting Synchronous Traffic, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 6, No. 11, November 1995

[6] G. Agrawal, B. Chen, and W. Zhao, Local Synchronous Capacity Allocation Schemes for Guaranteeing Message Deadlines with the Timed Token Protocol, *IEEE INFOCOM*, March 1993

[7] Q. Zheng and K. G. Shin, Synchronous bandwidth allocation in FDDI networks, In *Proceedings ACM Multimedia'93: First International Conference on Multimedia*, August 1993

[8] Moncef Hamdaoui and Parameswaran Ramanathan, Selection of Timed Token Protocol Parameters to Guarantee Message Deadlines, *IEEE/ACM Transactions on Networking*, vol. 3, no.3, pp. 340 351, June 1995

[9] Moncef Hamdaoui and Parameswaran Ramanathan, Deferring Real-Time Traffic for Improved Non-Real-Time Communication in FDDI Networks, *IEEE Transactions on Computers*, Vol. 44, No. 12, December 1995

오 성 훈

1996년 2월 인천대학교 정보통신공학(학사). 1998년 8월 숭실대학교 전자계산학(석사). 1998년 9월 ~ 현재 숭실대학교 컴퓨터학과 박사과정. 관심분야는 실시간 스케줄링, 운영체제, 실시간 통신 프로토콜 등임.

최 중 섭

1993년 2월 인천대학교 전자계산학 (학사). 1995년 8월 숭실대학교 전자계산학(석사). 1996년 3월 ~ 현재 숭실대학교 컴퓨터학과 박사과정. 관심분야는 실시간 운영체제, 스케줄링, 분산 실시간 시스템

양 승 민

1978년 2월 서울대학교 전자공학(학사). 1983년 4월 (미) Univ. of South Florida 전산학(석사). 1986년 12월 (미) Univ. of South Florida 전산학(박사). 1988년 ~ 1993년 (미) Univ. of Texas at Arlington 조교수. 1993년 ~ 현재 숭실대학교 컴퓨터학과 부교수. 1996년 ~ 1998년 국회도서관 정보처리국장. 관심분야는 실시간 시스템, 운영체제, 결함허용 시스템 등임.