韓國 컴퓨터情報學會 論文誌
第13卷 第1號, 2008. 1.

2008-13-1-3-6

단방향 지연 변이와 일주 지연을 이용한 양단간의 단방향 지연 추정

김 동 근*, 이 재 용**

One-Way Delay Estimation Using One-Way Delay Variation and Round-Trip Time

Dongkeun Kim*, Jaiyong Lee**

요 약

네트워크에서 QoS(Quality of Service) 제공 기술은 양단간 네트워크 경로의 안정성과 성능의 정도를 나타내는 QoS 척도에 대한 실제 측정에 기반을 두고 있다. QoS 척도 중에서 특히 단방향 지연의 측정은 양단간 두 측정 지점 간의 클럭(clock) 동기가 선행되어야 한다. 하지만, 네트워크에서 모든 단말 또는 호스트(host) 사이에는 절대적 또는 상대적인 시간 차이가 존재한다. 본 논문에서는, 단방향 지연, 단방향 지연 변이와 일주 지연(round-trip time: RTT) 간의 관계식을 새롭게 유도하여, 추정 오류가 일주 지연의 사분의 일 이하가 됨을 수학적으로 보여주며, 이를 이용한 단방향 지연과 클럭 오프셋(offset)의 추정 기법을 제안하고 실험을 통하여 본 제안의 유용성을 보여준다.

Abstract

QoS-support technology in networks is based on measuring QoS metrics which reflect a magnitude of stability and performance. The one-way delay measurement of the QoS metrics especially requires a guarantee of clock synchronization between end-to-end hosts. However, the hosts in networks have a relative or absolute difference in clock time by reason of clock offsets, clock skews and clock adjustments. In this paper, we present a theorem, methods and simulation results of one-way delay and clock offset estimations between end-to-end hosts. The proposed theorem is a relationship between one-way delay, one-way delay variation and round-trip time. And we show that the estimation error is mathematically smaller than a quarter of round-trip time.

▸ Keyword : one-way delay, delay variation, jitter, round-trip time, clock synchronization

• 제1저자 : 김동근 교신저자 : 이재용
• 접수일 : 2008. 1. 1, 심사일 : 2008. 1.2, 심사완료일 : 2008. 1.25.
* 연세대학교 전기전자공학과 박사과정 ** 연세대학교 전기전자공학과 교수
※본 연구는 정보통신부 및 정보통신연구진흥원의 대학IT연구센터 지원사업의 연구결과로 수행되었음
(IITA-2007-C1090-0701-0038)

Ⅰ. 서 론

전네트워크에서 QoS(Quality of Service) 제공 기술은 양단간(end-to-end) 네트워크 경로의 안정성(stability)과 성능(performance)의 정도를 나타내는 QoS 척도(metrics)에 대한 실제 측정(measurement)에 기반을 두고 있다[1]. 지연(delay), 지연 변이(delay variation), 가용 대역폭(available bandwidth) 그리고 패킷 오류율(packet error rate) 등과 같은 QoS 척도 중에서 특히 단방향(one-way) 지연의 측정은 양단간 두 측정 지점간의 클럭 동기(clock synchronization)가 선행되어야 한다. 하지만, 네트워크에서 모든 단말(terminal) 또는 호스트(host) 사이에는 절대적(absolute) 또는 상대적(relative)인 시간 차이가 존재 한다. 이러한 차이의 주요 원인으로는 클럭 오프셋(offset), 클럭 스큐(skew) 그리고 클럭 조정(adjustment)이 있다. 이 밖에 시간과 클럭에 관련된 많은 이슈(issue)가 있지만[2, 3, 4, 5, 6], 본 논문은 주로 클럭 오프셋과 클럭 스큐에 집중하기로 한다. 편의상 표기법(notation) 외에 용어개념(terminology)은 논문 [2, 3, 4, 5, 6, 10, 11]을 따르기로 하고, 먼저 주요 용어를 정의하면 다음과 같다 :

- 진동수(Frequency): 클럭이 진행하는 속도.
- 오프셋(Offset): 클럭에 의해 보고되는 시간과 국가 표준시와의 차이. 클럭간의 보고되는 시간 차이는 상대적(relative) 오프셋이라고 한다.
- 스큐(Skew): 클럭과 국가 표준시의 순간 진동수 차이. 그리고 클럭간의 순간 진동수 차이는 상대적 스큐라 한다. 즉, 스큐는 특정 시점에서 표준시에 대한 오프셋의 일차 도함수(first derivative)가 되고 상대적 스큐는 특정 시점에서 두 클럭간 상대적 오프셋의 일차 도함수가 된다.

현재 호스트의 시간을 표준시에 동기화 시키는 가장 정확한 방법은 그림 1과 같이 GPS(Global Positioning System) 위성으로부터 각각의 단말들이 수신기를 통하여 시간 정보를 수신하는 방식이다. 하지만 이 방법은 안테나와 같은 별도의 장치를 요구하기 때문에 특히 유비쿼터스(ubiquitous) 환경과 같이 모든 단말이 자신의 클럭을 갖고, 네트워크 프로토콜을 이용하여 분산 클럭 동기를 실현하는 All-IP 시대에는 적합하지 않다. 이러한 이유로, 클럭 동기 문제를 해결하기 위한 다양한 추정(estimation) 방법이 제안되고 있다.

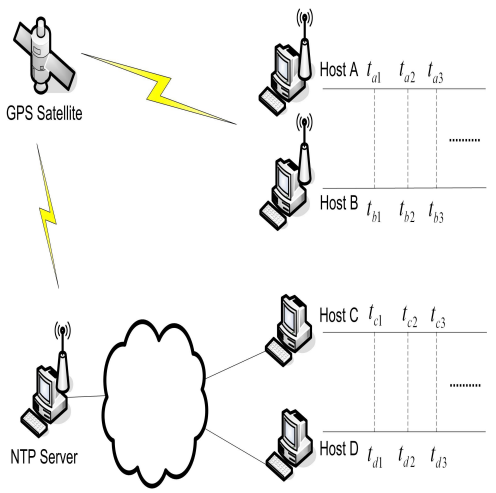

그림 1 GPS 와 NTP를 사용한 시간 동기화 방법
Fig. 1 Clock synchronization methods using GPS and NTP

먼저, 클럭 오프셋 추정은 두 가지 기본적인 가정에 따른 접근법(approach)이 있다. 첫째로는, 양단간의 경로(path)가 양방향 대칭적(symmetric)이라고 가정하여, 일주 지연(round-trip time: RTT)의 1/2을 단방향 지연이라고 추정하는 단순하지만 오차가 큰 추정 기법이 있다. 그림 1에서 보여주듯이 NTP(Network Time Protocol)[3] 서버를 통한 동기화 방법에서 사용되는 클럭 오프셋 추정 기법이 여기에 해당한다. 둘째로, 본 논문에서 제안하는 방식을 포함하여 양단간의 경로가 비대칭적(asymmetric)이라는 특성[5, 6, 12, 13]에 따른 추정하는 방법이 있다. 논문 [5, 6]에서는 최소 RTT를 두 호스트간의 클럭 오프셋으로 사용하였다. 그리고, 논문 [12]에서는 순반향(forward) 지연과 역방향(reverse) 지연의 초기 값, 그리고 전송 측에서 측정한 RTT와 수신 측에서 측정한 RTT를 이용하여 n 번째 순방향과 역방향의 지연을 분석적(analytic)으로 유도하고, 이 식에서 사용된 순방향과 역방향 지연의 초기 값을 휴리스틱(heuristic) 하게 구하는 방법을 제안하고 있다. 결국 이 방법은 그 초기 값을 얼마나 정확하게 추정할 수 있는가의 문제로 귀결된다. 그리고 논문 [13]은 양방향 비대칭적인 대역폭의 추정을 통하여 이에 따른 비대칭한 단방향 지연 추정 방안을 제안하였다.

또한, 클럭 스큐 추정 방법에 대해서는 다양한 방법이 제안되고 있지만[5, 6, 7, 8, 9], 단방향 지연 변이 또는 단방향 지터(jitter)를 이용하는 추정의 경우에는 두 연속한

(successive) RTT의 차이에만 의존하므로 클럭 스큐의 영향은 자연스럽게 상쇄(canceling each other)할 수 있다[11, 12]. 이러한 원리는 본 논문에서 제안하는 방법에도 활용하였다.

더불어, 본 논문의 제안은 단방향 지연과 RTT 그리고 단방향 지터의 관계식을 수학적으로 규명하고, 특정 조건에서 단방향 지연 추정 오류의 상한이 RTT/4보다 작음을 보여준다.

이외에 두 호스트간의 동기뿐만 아니라 그림 1과 같이 네트워크로 연결된 수많은 호스트간의 네트워크 동기화(networked synchronization) 문제에 관해서는 NTP 프로토콜이 좋은 해법으로 사용되고 있지만, 본 논문에서 제안하는 방식 또한 시간 서버(server)가 시간 정보를 요청하는 클라이언트(client)에 대하여 비상태유지(stateless) 특성을 갖도록 고안되어 네트워크 동기화 상황에서 강건한(robust) 특성을 보일 수 있다.

이어서, 2장에서는 동적(active) 측정 방식의 일종인 본 논문에서 제안하는 핑-퐁(ping-pong) 절차에 대하여 설명하고 관련 기본 정의를 연구한다. 3장에서는 동기화된 두 호스트에서 특정 조건하에서는 측정한 단방향 지터의 비율과 미지의(unknown) 단방향 지연의 비율이 같음을 보여주는 새로운 정리(theorem)를 유도하고, 이 정리를 이용하여 비동기된(unsynchronized) 두 호스트에 대해서 단방향 지연과 클럭 오프셋을 추정할 수 있는 일반화된 사례 연구(case study)와 알고리즘을 제안 한다. 4장에서는 제안 알고리즘의 시뮬레이션 결과와 분석을 수행하고 5장에서 끝을 맺는다.

II. 관련 연구

본 장에서는 동적 측정 방법의 일종인 핑-퐁 절차를 소개하고 관련된 기본적인 수학적 정의와 표기법을 정의한다.

본 제안에서 사용된 핑-퐁 절차는 그림 2와 같이 호스트 A가 k번째 시간 스탬프(timestamp) 요청을 ta(k) 시간에 호스트 B에게 요청하는 것으로 시작한다. 호스트 B는 tb(k) 시점에 k번째 패킷을 수신 하는 즉시 tb(k) 시점에 바로 호스트 A에게 응답을 보낸다. 또한, 호스트 A는 ta(k+1) 시점에 그 응답을 받는 즉시, k+1번째 요청을 호스트 B에게 전송한다. 이러한 요청과 응답은 임의의 주어진 기간 동안 반복된다.

두 호스트 모두가 표준시일 필요는 없지만, 그들의 상대적 오프셋은 중요하다. 편의상 그림 2와 같이 호스트 B만 표준시를 갖는다고 가정한다. 각각의 호스트 A와 B에 대해서 Ra(k) 와 Rb(k)로 표시되는 k번째 RTT는 식(1)과 같이 정의되고 측정할 수 있다.

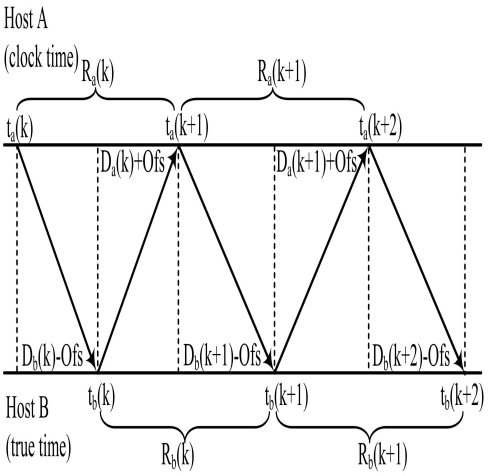

그림 2 비동기된 두 호스트 A 와 B의 사례
Fig. 2 Unsynchronized case between hosts A and B

$$R_b(k) \equiv D_a(k) + D_b(k+1) = t_b(k+1) - t_b(k),$$
$$R_a(k) \equiv D_a(k) + D_b(k) \approx t_a(k+1) - t_a(k). \quad \cdots\cdots (1)$$

호스트 B에 대해서, Db(k)로 표기되는 k번째 실제 단방향 지연과 Jb(k)로 표기되는 k번째 단방향 지터는 다음과 같이 표현 될 수 있다(단, $0 \le k \le n$, for integers n, $k \ge 0$). 지터의 측정은 단지 두 번의 RTT 구간이기 때문에, 클럭 스큐의 영향은 무시할 수 있을 정도로 작고, Ofsa(k)와 Ofsa(k+1)로 표기되는 호스트 A의 k번째와 k+1번째의 실제 오프셋은 서로 상쇄된다. 즉, Ofsa = Ofsa(k) = Ofsa(k+1)이 된다[11, 12]. 따라서, Db(k)와 Jb(k)는 식(2)와 같이 정의되고 측정되며, 호스트 A에 대해서도 같은 방식으로 식(3)과 같이 정리할 수 있다.

$$D_b(k) = [t_b(k) - t_a(k)] + Ofs_a(k) \text{(, for integer } k \ge 0),$$
$$J_b(k) \equiv D_b(k+1) - D_b(k) = R_b(k) - R_a(k). \quad \cdots (2)$$

$$D_a(k) = [t_a(k+1) - t_b(k)] - Ofs_a(k) \text{(, for integer } k \ge 0),$$
$$J_a(k) \equiv D_a(k+1) - D_b(k) = R_a(k+1) - R_b(k). \quad \cdots (3)$$

결국, 단방향 지터는 클럭의 동기화 없이 자신과 상대(peer)의 RTT만으로 구할 수 있음을 알 수 있다[11, 12].

III. 제안하는 내용

3.1 동기화된 두 호스트의 이상적인 사례 연구

그림 3과 같이 두 호스트가 동기화된 상태에서 Ja(k)≠0 이고 Jb(k)≠0 이라고 가정하자. 각각의 호스트에서 측정한 단방향 지터의 비 Ja(k)/Jb(k)는 식(4)와 같이 정의하고 변형할 수 있다.

$$\frac{J_a(k)}{J_b(k)} \equiv \frac{D_a(k+1)-D_a(k)}{D_b(k+1)-D_b(k)} = \frac{R_a(k+1)-R_b(k)}{R_b(k)-R_a(k)}$$

$$= \frac{D_a(k+1)}{D_b(k+1)} \times \left(\frac{1-\frac{D_a(k)}{D_a(k+1)}}{1-\frac{D_b(k)}{D_b(k+1)}} \right)$$

$$= \frac{D_a(k)}{D_b(k)} \times \left(\frac{\frac{D_a(k+1)}{D_a(k)}-1}{\frac{D_b(k+1)}{D_b(k)}-1} \right). \quad \cdots\cdots (4)$$

식(4)는 단방향 지터의 비가 "미지의 실제(unknown and true)" 단방향 지연으로 정의 될 뿐만 아니라, 측정한 RTT로 표현될 수 있음을 보여준다.

또한, 식(4)에서 괄호(.) 안의 두 항이 1이 되면, 단방향 지연의 비는 클럭 동기화가 필요 없이 측정 가능한 RTT로 표현되는 지터의 비와 같음을 식(5)에서 알 수 있다.

$$\frac{J_a(k)}{J_b(k)} = \frac{D_a(k)}{D_b(k)} = \frac{D_a(k+1)}{D_b(k+1)} = \frac{R_a(k+1)-R_b(k)}{R_b(k)-R_a(k)}$$

$$, when \left(\frac{1-\frac{D_a(k)}{D_a(k+1)}}{1-\frac{D_b(k)}{D_b(k+1)}} \right) = \left(\frac{\frac{D_a(k+1)}{D_a(k)}-1}{\frac{D_b(k+1)}{D_b(k)}-1} \right) = 1. \quad \cdots\cdots (5)$$

그리고, 이 경우(호스트 A와 B가 동기화 되고, 식(5)가 성립하는 경우)에 측정한 RTT를 이용하여 Db(k), Db(k+1), Da(k) 와 Da(k+1)의 해(solution)를 식(6)과 같이 구할 수 있고 이 해의 유일함(uniqueness)은 부록에서 증명 하였다.

$$D_a(k+1) = \frac{R_a(k+1)-R_b(k)}{R_a(k+1)-R_a(k)} R_a(k+1),$$

$$D_a(k) = \frac{R_a(k+1)-R_b(k)}{R_a(k+1)-R_a(k)} R_a(k),$$

$$D_b(k+1) = \frac{R_b(k)-R_a(k)}{R_a(k+1)-R_a(k)} R_a(k+1),$$

$$D_b(k) = \frac{R_b(k)-R_a(k)}{R_a(k+1)-R_a(k)} R_a(k). \quad \cdots\cdots (6)$$

이어서 다음 절부터는 본 절에서 소개한 동기된 두 호스트 간의 특수한 사례 연구를 기반으로, 비동기된 두 호스트간의 일반화된 사례 연구를 통하여 식(5)가 성립되는 경우를 실질적(practical)으로 탐지(detection)하고 측정할 수 있는 방법과 함께, 단방향 지연과 클럭 오프셋의 추정 방법을 제안한다.

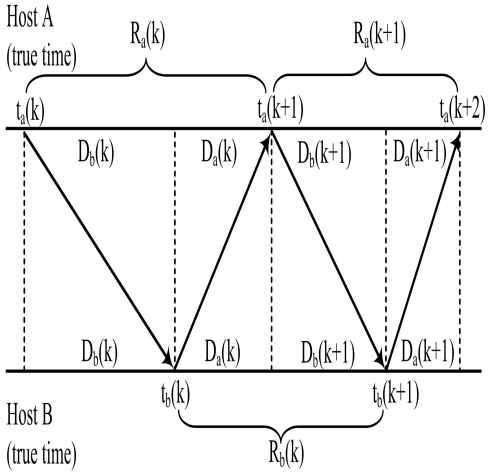

그림 3 동기화된 두 호스트 A 와 B의 사례
Fig. 3 Synchronized case between hosts A and B

3.2 비동기된 두 호스트의 현실적인 사례 연구

그림 4에서 호스트 A와 B는 서로 비동기된 상태이다. 두 호스트가 동기된 경우의 식(6)과 같이 구한 해는 비동기된 상태에서는 식(7)과 같이 Db(k), Db(k+1), Da(k) 와 Da(k+1)가 아닌 그림 4의 B(k), A(k), B(k+1)와 A(k+1)가 된다. 또한, 만약에 각각의 B(k), A(k), B(k+1)와 A(k+1)가 각각의 tb(k) - ta(k), ta(k+1) - tb(k), tb(k+1) - ta(k+1) 그리고 ta(k+2) - tb(k+1)와 같은 경우에는,

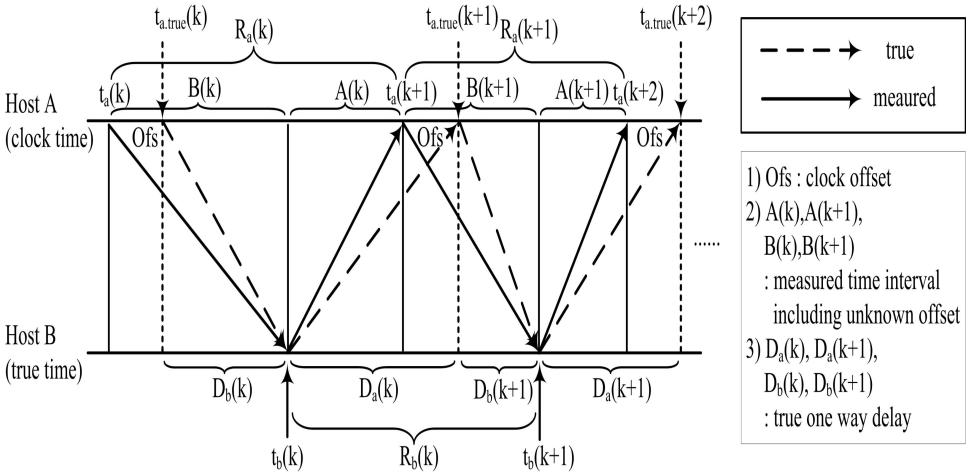

그림 4 두 호스트 A 와 B의 비동기 사례 연구(단, Ofsa 〉 0)
Fig. 4 Unsynchronized case study between hosts A and B(, when Ofsa 〉 0)

실제 오프셋 Ofsa(k)가 식(8)과 같이 표현될 수 있다.

$$B(k)=\frac{R_b(k)-R_a(k)}{R_a(k+1)-R_a(k)}R_a(k)$$
$$=t_b(k)-t_a(k)>0,$$

$$A(k)=\frac{R_a(k+1)-R_b(k)}{R_a(k+1)-R_a(k)}R_a(k)$$
$$=t_a(k+1)-t_b(k)>0,$$

$$B(k+1)=\frac{R_b(k)-R_a(k)}{R_a(k+1)-R_a(k)}R_a(k+1)$$
$$=t_b(k+1)-t_a(k+1)>0,$$

$$A(k+1)=\frac{R_a(k+1)-R_b(k)}{R_a(k+1)-R_a(k)}R_a(k+1)$$
$$=t_a(k+2)-t_b(k+1)>0. \quad \cdots\cdots (7)$$

$$Ofs_a(k)=B(k)-D_b(k)=D_a(k)-A(k)$$
$$=B(k+1)-D_b(k+1)=D_a(k+1)-A(k). \quad \cdots\cdots (8)$$

그리고, 이와 같이 호스트 A와 B가 비동기된 경우에 식(7)을 만족하는 경우를 탐지했다는 것은 식(5)가 특정한 오류 범위 내에서 성립하는 경우가 탐지된 것을 의미하고, 그 오류 범위는 아래에서 계속하여 구해보기로 한다.

식(8)에 따라서, Ofsa(k) 〉 0 인 경우에, 각각 ta.true(k) 와 ta.true(k+1)로 표기되는 실제 측정 시간(단, tb(k)는 자체로 실제 측정 시간이 된다.)은 각각 B(k)와 B(k+1)의 범위에 위치하여야 한다. 따라서, Ofsa(k)는 B(k)와 B(k+1) 중에서 더 작은 범위로 제한된다. 즉, 0 ≤ Ofsa(k) ≤ min〔B(k), B(k+1)〕 이 된다. 그리고, 같은 논리로 Ofsa(k) 〈 0 인 경우에는 |Ofsa(k)|가 A(k)와 A(k+1) 중에서 더 작은 범위로 제한된다. 즉, 0≤ |Ofsa(k)| ≤ min〔A(k), A(k+1)〕이 된다. 게다가, Ofsa.estim(k)로 표기되는 추정 오프셋의 오류를 최소화하기 위하여, Ofsa.estim(k)는 식(9)와 같이 그 제한 범위의 이분의 일로 선택한다.

$$Ofs_{a.estim}(k)=\begin{cases}\dfrac{\min[B(k),B(k+1)]}{2}, & for\ Ofs_a(k)>0,\\ \dfrac{-\min[A(k),A(k+1)]}{2}, & for\ Ofs_a(k)<0.\end{cases} \quad \cdots\cdots (9)$$

EOfs(k) = |Ofsa(k) - Ofsa.estim(k)|로 표현할 수 있는 Ofsa.estim(k)의 오류 범위는 식(10)과 같이 정리된다.

$$0\le E_{Ofs}(k)\le\begin{cases}\dfrac{\min[B(k),B(k+1)]}{2}, & for\ Ofs_a(k)>0,\\ \dfrac{\min[A(k),A(k+1)]}{2}, & for\ Ofs_a(k)<0.\end{cases}$$
$$\le\frac{\min[R(k),R(k+1)]}{4}. \quad \cdots\cdots (10)$$

비록 Ofsa.estim(k)를 식(9)와 같이 결정할 수 있지만, Ofsa(k)가 Ofsa(k) 〉 0 또는 Ofsa(k) 〈 0 인지 알 수 없

다. 하지만, Ofsa(k)가 Ofsa(k) 〉 0 또는 Ofsa(k) 〈 0 이 되도록 의도적으로 Ofsa(i)(단, i는 초기 오프셋의 인덱스)를 결정하여 초기화 할 수 있다. 게다가, 단방향 지연의 양방향 비대칭성이 시간의 흐름에 따라 역전되지 않는 경우에는, Ofsa.estim(k) 와 EOfs(k)는 min〔R(k), R(k+1)〕/4의 범위로 제한된다. 왜냐하면, 의도적으로 min{min〔A(i), A(i+1)〕, min〔B(i), B(i+1)〕}로 초기화시키면, EOfs(k)는 min〔B(k), A(k), B(k+1), A(k+1)〕/2 의 범위로 제한될 수 있기 때문이다. 즉, 이러한 유도 결과는 본 제안의 추정 오류가 단순히 RTT/2를 사용하는 방식보다 수학적으로 두 배의 정밀도 향상을 갖는다는 것을 보여준다.

$$
\begin{aligned}
D_{b.estim}(k) &= -Ofs_{a.estim} + (t_b(k) - t_a(k)) \\
D_{a.estim}(k) &= Ofs_{a.estim} + (t_a(k+1) - t_b(k)) \\
D_{b.estim}(k+1) &= -Ofs_{a.estim} + (t_b(k+1) - t_a(k+1)) \\
D_{a.estim}(k+1) &= -Ofs_{a.estim} + (t_a(k+2) - t_b(k+1)) \quad \cdots\cdots `(11)
\end{aligned}
$$

그리고, Da.estim(k), Da.estim(k+1), Db.estim(k) 와 Db.estim(k+1)로 표기 되는 추정 단방향 지연은 식(8)에서 Ofsa(k)를 Ofsa.estim(k)로 대치하여 식(11)과 같이 구할 수 있다.

3.3 조건의 탐지

식(7)의 경우는 양변이 실수이기 때문에, 식(7)의 경우를 탐지하고 측정할 수 있는 경우(즉, 식(5)가 식(10)의 오류 범위 내에서 성립하는 경우)가 현실적으로 자주 발생하기 힘들기 때문에, 우리는 몇 가지 기준(criterion)과 보완을 필요로 한다.

먼저, 이러한 기준을 정의하기 전에 식(7)을 분석하여 보면 다음과 같이 세 가지 조건을 알 수 있다 :

1) 측정의 시작, 패킷 타임아웃(timeout) 그리고 비정렬(out-of-order) 패킷 수신을 고려하여 두 개의 연속된 RTT가 측정 가능하여야 한다.
2) Ra(k) 와 Rb(k) 는 단조(monotonic) 증가 또는 단조 감소 해야 한다. 즉, {Ra(k) 〈 Rb(k) 〈 Ra(k+1)} 또는 {Ra(k) 〉 Rb(k) 〉 Ra(k+1)}.
3) 측정 시간 ta(k) 와 tb(k)는 단조 증가해야 한다. 즉, {ta(k) 〈 tb(k) 〈 ta(k+1)} 그리고 {ta(k+1) 〈 tb(k+1) 〈 ta(k+2)}.

상기 세 가지 조건에 따라, "측정가능성(measurability)" α로 명명한 패러미터(parameter)를 정의하기로 한다. 이 패러미터는 식(12) 와 같이 총 핑-퐁 회수에 대해서 상기 세 가지 조건을 만족하는 회수의 비율로 정의 한다.

$$
\alpha \equiv \frac{measured\ counts}{total\ ping\ counts} \quad \cdots\cdots (12)
$$

비록 측정가능성 α가 식(7)의 경우가 탐지되는(즉, 식(5)가 식(10)의 오류 범위 내에서 성립하는) 빈도를 어느 정도 반영할 수 있는 현실적인 기준이기는 하지만, 얼마나 정확하게 식(7)을 만족하는지 알 수 없다. 따라서, 식(13)과 같이 "정확도(accuracy factor)" 벡터(vector) Δ(k) = 〔δ0(k), δ1(k),δ2(k), δ3(k)〕를 정의하기로 한다. 이 벡터는 측정한 값이 식(7)에 근접하는 정도를 나타내고 있다.

$$
\begin{aligned}
&\delta_0(k) \equiv B(k) - (t_b(k) - t_a(k)), \\
&\delta_2(k) \equiv B(k+1) - (t_b(k+1) - t_a(k+1)), \\
&\delta_1(k) \equiv A(k) - (t_a(k+1) - t_b(k)), \\
&\delta_3(k) \equiv A(k+1) - (t_a(k+2) - t_b(k+1)), \\
&|\Delta(k)| = \sqrt{\delta_0(k)^2 + \delta_1(k)^2 + \delta_2(k)^2 + \delta_3(k)^2}. \quad \cdots\cdots (13)
\end{aligned}
$$

즉, δ0(k), δ1(k), δ2(k) 와 δ3(k)가 0에 가까울수록, 식(7)에 근접한 결과를 얻을 수 있다. 따라서 측정된 모든 |Δ(k)| 중에서 최소의 |Δ(m)|인 경우에, 식(9)를 이용하여 최선으로 추정된 Ofsa.estim(m)를 결정할 수 있다(단, 최적의 추정 방법은 추가 연구가 필요하다).

3.4 의도적인 오프셋 초기화

앞에서 Ofsa(k)가 Ofsa(k) 〉 0 인지 또는 Ofsa(k) 〈 0 인지 알 수 있는 방법은 없지만 Ofsa(k) 〉 0 또는 Ofsa(k) 〈 0 이 되도록 의도적으로 초기 오프셋을 선택할 수 있다고 언급하였다. 이러한 초기 오프셋의 선택은 측정가능성과 정확도에 직-간접적으로 영향을 미칠 수 있는 요소이지만, 실제로 추정하려는 오프셋처럼 비교 대상이나 추정 대상은 아니다. 왜냐하면, 우리가 최종적으로 추정하고자 하는 오프셋은 정확도 벡터 Δ(k)와 같이 식(7)에 가장 부합하는 정도를 나타낼 수 있어야 한다. 이어서, 그림 5의 전체 측정 과정의 일부로써, 식(9)에서 보다 작은 범위로 추정 오프셋을 제한할 수 있도록 하기 위하여 의도적으로 초기 오프셋의 부호와 크기를 결정하는 알고리즘을 제안 한다 :

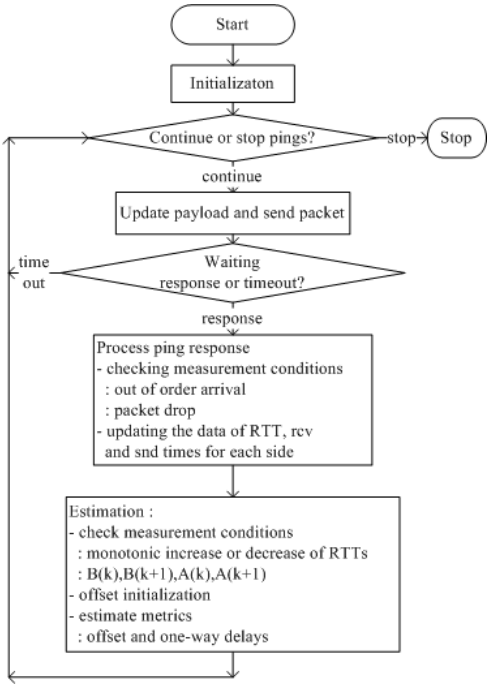

그림 5 전체 측정 흐름도 개요
Fig. 5. Measurement flow chart

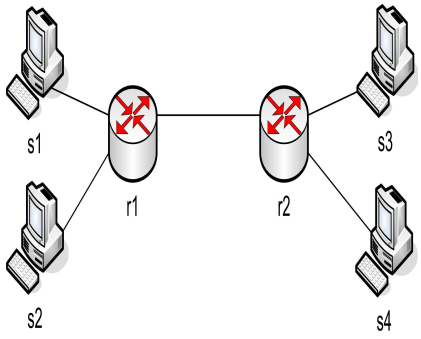

그림 6 시뮬레이션 구성도
Fig. 6 Simulation topology

1) 측정을 시작하여 상기 측정가능성의 세 가지 조건에 부합하는 연속된 Ra(k)와 Rb(k)를 기반으로 B(k), A(k), B(k+1)와 A(k+1)을 구한다.
2) 미리 정한 회수 내에서 B(k)와 A(k) 중에서 작은 경우가 더 많이 발생하는 것을 택하고, 이렇게 선택된 B(k)나 A(k) 중에서 최소 값을 초기 오프셋의 타겟(target)으로 선정한다.
3) 2)에서 정한 타겟을 식(9)에 적용하여 초기 오프셋의 크기와 부호를 결정한다.

IV. 시뮬레이션 결과 및 분석

본 절에서는, OMNet++ 시뮬레이터[14]를 이용하여 본 논문에서 제안한 추정 방안을 평가하였다. 먼저 그림 5의 전체 측정 흐름도는 RTT 측정으로 네트워크의 경로의 상태를 체크하는 용도로 많이 사용되는 어플리케이션인 핑(ping)과 상당히 유사하다. 차이점으로는 본 논문에서 제안하는 핑-퐁 절차를 구현하기 위하여, 핑 패킷을 주기적으로 전송하지 않고, 핑에 대한 응답을 받아서 측정 및 추정 관련 계산 처리 후에 바로 다음 핑 패킷을 전송하는 방식으로 구현하였다.

시뮬레이션 구성도(topology)는 그림 6과 같이 s1에서 s4까지의 네 개의 호스트와 r1과 r2의 두 개의 라우터(router)로 구성 되었다. 호스트 s3는 표준시 서버이고 호스트 s4는 FTP 어플리케이션의 싱크(sink)로 설정하였다. 그리고, 호스트 s1 은 표준시 서버에게 시간스탬프를 요청하는 클라이언트이고, 호스트 s2는 FTP 어플리케이션의 소스(source)로 설정하였다. 호스트와 라우터는 1.5 Mbps의 물리 링크(physical link) 상에서의 PPP(Point-to-Point Protocol) 데이터링크(data link) 프로토콜로 연결되었다.

먼저, 측정가능성 α 의 실험 결과는 표 1처럼 다양한 네트워크 트래픽(traffic) 환경 아래서 수행되었다. 표 1에서 최대 크기 10에서 50 개의 패킷을 갖는 PPP 수신 버퍼의 혼잡제어(congestion control) 알고리즘으로 Drop-tail 과 RED[7] 알고리즘이 사용되었다. 패킷 폐기(drop) 수가 적어질수록 측정가능성 α의 값은 커지는 것을 알 수 있다. 하지만, 비록 패킷 폐기가 급격하게 증가하더라도 측정가능성 α의 값은 그만큼 급격하게 저하되지는 않는다는 것 또한 알 수 있다. 측정가능성 α가 27% 이상의 값을 갖는 다는 의미는, 측정가능성의 세 가지 조건하에서 총 핑-퐁 패킷 중에서 단방향 지연과 클럭 오프셋을 더욱 정확하게 추정할 수 있는 기회를 약 27% 이상 갖게 된다는 것을 뜻한다. 측정가능성 α 의 세 가지 조건 중에서 두 번째 조건은 네트워크 혼잡에 따른 RTT 요동(fluctuation)이 발생하는 경우에 흔히 발생할 수 있는 상황이다[15].

그림 7은 표 1의 마지막 행의 조건으로, 본 논문에서 제안하는 방식과 실제 단방향 지연 그리고 RTT/2의 결과를 윈도우-평균한(window-averaged) 결과(단, 윈도우 크기 = 20)를 보여주고 있다. 다른 방식은 추정 오류의 수학적(이론적) 상한이 실시간(real-time) 추정 시에 RTT/2 보다 작다고 볼 수 없기 때문에 따로 비교하지 않는다.

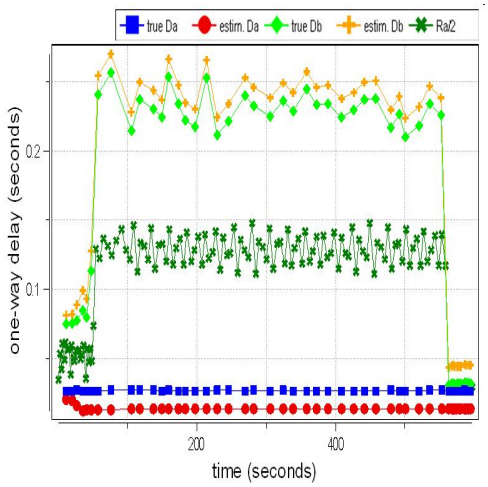

그림 7 추정지연과 실제 지연 그리고 RTT/2의 비교
Fig. 7 Estimated delay vs true delay vs. RTT/2

그림 7에서 볼 수 있듯이, 본 논문에서 제안하는 방식으로 추정한 단방향 지연 "estim. Da 과 estim. Db"은 RTT/2인 "Ra/2"와 명확히 구분되면서, 실제 단방향 지연 "true Da 와 true Db"를 매우 밀접하게 추적하고(closely tracking) 있음을 알 수 있다. 그리고 estim. Da, estim. Db, true Da 그리고 true Db 의 값을 표기한 점의 수가 Ra/2를 표기한 점의 수와 다른 것은 측정가능성의 세 가지 조건을 만족하는 경우만 고려되기 때문이다.

마지막으로, 본 논문의 제안은 다음과 같은 장점을 가지고 있다:

- 양단의 호스트가 동기화되어 있음을 가정하지 않는다.
- 단방향 지연의 변화를 실시간으로 추적할 수 있다.
- 오히려 네트워크 혼잡 상황에서 더 잘 동작한다.
- 단지 연속하는 두 개의 RTT 단위로 추정하기 때문에 측정을 지속하는 경우에 발생할 수 있는 누적적인 오류(cumulative error)가 발생하지 않는다.

V. 결론

본 논문에서는 특정 조건에서, 비동기 상태에서 측정 가능한 단방향 지연변이의 비율과 미지의 단방향 지연의 비율이 같음을 보여주는 정리를 유도하고 제안하였으며, 이를 이용한 단방향 지연과 클럭 오프셋의 추정 방법을 제안하고 시뮬레이션을 통하여 그 실용성을 보였다. 또한 제안하는 정리를 통하여 특정 조건하에서 추정 오류의 상한이 수학적으로 RTT/4 보다 작음을 보여 주어 기존의 RTT/2을 사용하는 방식보다 두 배의 정밀도 향상이 이론적으로 가능함을 보였다.

그리고 제안하는 방식은 시간 서버가 시간 정보를 요청하는 클라이언트에 대하여 비상태유지 특성을 갖도록 고안되어 네트워크 동기화 상황에서 강건한 특성을 보일 수 있으며, 앞에서 언급하였듯이 양단의 호스트가 동기화되어 있음을 가정하지 않고, 단방향 지연의 변화를 실시간으로 추적할 수 있으며, 오히려 네트워크 혼잡 상황에서 더 잘 동작하고, 마지막으로 연속하는 두 개의 RTT 단위로 추정하기 때문에 측정을 지속하는 경우에 발생할 수 있는 누적적인 오류가 발생하지 않는 장점을 가지고 있다.

하지만, 측정된 값들로부터 최적의 추정 값을 결정하는 방법과 핑-퐁 측정에 따른 측정 트래픽의 오버 헤드(overhead)의 영향과 감소 방안에 대해서는 추가 연구가 필요하다.

표 1 측정가능성(Measurability) (총 핑-퐁 회수=3000)
Table 1. Measurability (total ping-pong counts=3000)

PPP Queue Scheduling	Simulation Time (s)	Max. Queue (packets)	Drop Counts	Measured Counts	Measurability α (%)
Drop-tail	826.530	10	191	828	27.6
	803.985	30	55	896	29.9
	802.843	50	1	959	32.0
RED	846.278	10	162	809	27.0
	804.842	30	50	879	29.3
	803.385	50	8	960	32.0

Ⅵ. 부록

• 식 (6)의 유일성 증명

Da(k) : Da(k+1) = Db(k) : Db(k+1)인 상황에서 Rb(k)의 길이를 유지하면서 tb(k)와 tb(k+1)을 동시에 좌-우로 이동시킬 경우 Ra(k), Ra(k+1)와 Rb(k)의 길이는 일정하지만, Da(k) : Da(k+1) ≠ Db(k) : Db(k+1)하게 된다. 따라서, Da(k) : Da(k+1) = Db(k) : Db(k+1)이고 정해진 Ra(k), Ra(k+1)와 Rb(k)로 구할 수 있는 Da(k), Da(k+1), Db(k) 그리고 Db(k+1)의 값은 유일하다.

참고문헌

〔1〕 V. Paxson, G. Almes, J. Mahdavi, and M. Mathis, "Framework for IP performance metrics," RFC 2330, IETF, May 1998.

〔2〕 D.Mills, "Improved algorithms for synchronizing computer network clocks," IEEE/ACM Transactions on Networking, vol. 3, no. 3, pp. 245-254, Jun 1995

〔3〕 D. Mills, "Network Time Protocol (Version 3): Specification, Implementation and Analysis," RFC 1305, IETF, March 1992.

〔4〕 D. Mills, "Modelling, Analysis of Computer Network Clocks," Technical Report 92-5-2, Electrical Engineering Department, University of Delaware, May 1992.

〔5〕 V. Paxson, Measurements and Analysis of End-to-End Internet Dynamics, Ph.D. thesis, University of California Berkeley, April 1997.

〔6〕 V. Paxson, "On Calibrating Measurements of Packet Transit Times," In Proceedings of ACM SIGMETRICS, June 1998.

〔7〕 S. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurements," In Proceedings of INFOCOM 1999, March 1999.

〔8〕 Sue B. Moon, Measurement and Analysis of End-to-End Delay and Loss in The Internet, Ph.D Thesis, University of Massachusetts Amherst , Feb. 2000.

〔9〕 Li Zhang, Zhen Liu, and Cathy Honghui Xia, "Clock Synchronization Algorithms for Network Measurements," In Proceedings of INFOCOM 2002.

〔10〕 G. Almes, S. Kalidindi, and M. Zekauskas, "A One-way Delay Metric for IPPM," RFC 2679, IETF, Sep. 1999.

〔11〕 C. Demichelis and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)," RFC 3393, IETF, Nov. 2000.

〔12〕 Jin-Hee Choi, Chuck Yoo, "One-Way Delay Estimation and Its Application," Elsevier Computer Communications, Vol. 28, Issue 7, May 2005.

〔13〕 Masato TSURU, Tetsuya TAKINE, Yuji OIE, "Estimation of clock offset from one-way delay measurement on asymmetric paths," In Proceedings of the 2002 Symposium on Applications and the Internet, Feb. 2002.

〔14〕 A. Varga, OMNeT++: Discrete Event Simulation Environment. Available at http://www. omnetpp.org

〔15〕 S. Floyd, and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Trans. on Networking, vol.1, no. 4, p. 397-413, August 1993.

저 자 소 개

김 동 근

1997년 8월 : 연세대학교 전자공학과 졸업

1999년 8월 : 연세대학교 전기컴퓨터 공학과 석사

1999년 9월 ~ 현재 : 연세대학교 전기전자공학과 박사과정

2005년 5월 ~ 현재 : 삼성전자 책임 연구원

이 재 용

1977년 2월 : 연세대학교 전자공학과 졸업

1984년 2월 : Iowa 주립대학교 컴퓨터 공학과 석사

1987년 2월 : Iowa 주립대학교 컴퓨터 공학과 박사

1987년 7월 ~ 1994년 8월 : 포항공대 전산과 부교수

1994년 9월 ~ 현재 : 연세대학교 전기전자공학부 교수