스트리밍 서버의 고장탐지 기법에 대한 성능 분석

전성규° 차호정
연세대학교 컴퓨터과학과
{skjeon,hjcha}@cs.yonsei.ac.kr

Performance Evaluation of a Failure Detection mechanism for Streaming Server

Seongkwu Jeon° Hojung Cha
Dept. of Computer Science, Yonsei University

요 약

본 논문은 스트리밍 환경에서 서버의 고장을 빠르게 탐지하기 위해 동적임계점을 사용하고 이에 대한 성능을 분석한다. 제안된 기법은 스트리밍의 특성을 이용하여 질의 전송 시간을 결정하게 되는데 서버의 패킷도착 지연으로 인해 발생되는 질의 전송 시간의 증가를 최소화시키기 위해 패킷 지연도착 시간을 반영하지 않는 알고리즘을 적용하였다. 고장탐지에 대한 성능분석을 위해 스트리밍의 종류에 따라 질의 전송 시간이 다양하게 적용될 수 있기 때문에 다양한 스트리밍 자료를 활용하여 실험하였으며 제안된 기법의 성능을 검증하였다.

1. 서론

스트리밍 환경에서 예고되지 않은 서비스 중단은 소비자 서비스 수준의 저하를 초래하게 된다. 이를 개선하기 위해서는 서버의 고장을 지속적으로 감시해야 하는데 스트리밍 환경에서는 보다 빠른 고장탐지가 요구된다. 서버의 고장탐지는 질의를 주기적으로 보내거나 'heartbeat' 신호를 통해 이루어진다[1]. 이 경우 고정된 시간이내에 서버의 고장을 탐지하기 어렵다. 또한 서버와 사용자 사이에 정의된 고장탐지 프로토콜이 요구되어 기존 구조의 변경이 불가피하다[2]. 질의 전송은 지속적으로 수행하게 되며 네트워크 자원의 불필요한 낭비를 초래하게 된다. 따라서 제안된 것이 동적임계점을 활용한 서버의 고장탐지 방법이다[3]. 동적임계점은 서버의 응답지연시간을 기준으로 질의 전송시간을 결정하는 임계점을 단순히 최대 값으로 고정하지 않고 네트워크 상태 변화에 따라 적응적으로 이동하는 것으로 스트리밍 미디어에 의존적으로 결정된다. 질의는 스트리밍 패킷을 이용하여 전송되고 기존 프로토콜의 변경 없이 고장탐지를 수행할 수 있으며 스트리밍 서버가 서비스를 지원하지 않고 있다고 판단될 때에만 전송된다. 동적임계점은 스트리밍환경에 적합하도록 설계되었지만 스트리밍의 특성상 패킷의 지연도착이 발생할 경우 동적임계점이 증가하게 되어 스트리밍 서버의 고장탐지 시간이 늦어진다는 단점을 가지고 있다. 또한 질의 전송은 최대 패킷 도착 간격을 이용한 고장탐지보다 훨씬 증가하였다.

본 논문에서는 동적임계점의 단점을 개선하기 위해 스트리밍 서버의 패킷 지연도착을 동적임계점에 적용하지 않고 질의만 전송함으로써 동적임계점의 증가를 방지하여 고장탐지를 보다 빠르게 수행토록 개선된 동적 임계점 (E-DT: Enhanced - Dynamic Threshold)을 이용하였다. 또한 동적임계점은 스트리밍 미디어의 특성에 따라 다양하게 변화 될 수 있기 때문에 성능 검증을 위해 여러 미디어를 가지고 실험을 수행하였다. 본 논문의 구성은 2장에서 개선된 동적임계점과 이를 이용한 스트리밍 서버의 고장탐지에 대해 설명하고 3장에서는 여러 스트리밍 미디어에 대한 실험 및 분석결과를 제시하며 4장에서 결론을 맺는다.

본 연구는 한국과학재단에서 지원하는 특정기초연구사업으로 수행하였음 (과제번호 : R01-2002-000-00141-0).

2. 동적 임계점을 이용한 서버의 고장탐지

다음은 동적 임계점의 증가를 유발하는 패킷 지연도착 시간을 반영하지 않도록 설계되어진 개선된 동적 임계점(E-DT)에 대해 설명하고 이를 이용한 스트리밍 서버의 고장탐지에 대해 기술한다.

2.1 개선된 동적 임계점 (E-DT)

기존 동적 임계점은 갑작스런 패킷 지연도착 시간을 그대로 반영하기 때문에 동적 임계점의 증가를 초래하게 된다. 하지만 이를 동적임계점에 반영하지 않게 되면 감소율을 증가시킬 수 있다.

$$if(Cur_i < MaxRDT)$$
$$Dt_{i+1} = \begin{cases} Dt_i - f(t) & if(Dt_i > Cur_i), \\ Cur_i + f(t) & if(Dt_i \leq Cur_i) \end{cases}$$
$$elseif(Cur_i \geq MaxRDT)$$
$$Dt_{i+1} = \begin{cases} Dt_i - f(t) & if(Dt_i > Cur_i), \\ Dt_i + f(t) & if(Dt_i \leq Cur_i) \end{cases}$$

그림 1. E-DT 알고리즘

그림 1은 개선된 동적 임계점(E-DT)에 대한 알고리즘이다. 서버가 응답하는 메시지 P_{i-1}와 P_i의 도착 시간차이를 Cur_i라고 할 때 최대 패킷 도착 간격 MaxRDT는 $MAX(Cur_{i-1}, Cur_i)$가 된다. i번째 동적 임계점 DT_i는 Cur_i와 MaxRDT를 비교하여 결정된다. 이때 Cur_i가 MaxRDT를 초과하게 될 경우 서버의 패킷 지연도착 시간으로 간주하여 DT_i를 감소율 함수 $f(DT_i - Cur_i)$ 만큼 증감시킨다. $Cur_i \geq MaxRDT$ 일 경우, 기존에 Cur_i를 사용하여 증가했을 때보다 동적 임계점의 감소율이 증가하였다. 일시적으로 증가되는 패킷 지연도착 시간을 적용하지 않았기 때

문에 동적 임계점의 감소를 유도하여 다음 패킷 도착 시간을 보다 정확히 예측할 수 있었다. 따라서 임계점의 감소는 빠른 고장탐지를 가능케 한다. 감소율 함수 $f(t)$는 패킷 도착 간격이 포와송 분포를 따른다고 가정할 때 Markov chain에서 상태 전환 확률[4]을 적용하면 $f(t) = \frac{t}{e^{\frac{1}{\ln(t)}}}$과 같다. t 는 동적 임계점과 패킷 도착 간격의 차이를 말하며 도착간격의 차이가 작을수록 감소율은 감소된다.

2.2 스트리밍 서버의 고장탐지

스트리밍 서버의 고장탐지는 서버와 클라이언트 및 스트리밍을 제어하고 관리하며 서버의 고장탐지를 수행하는 프록시로 구성된 시스템 구조에서 수행된다[3]. 스트리밍 환경에서 별도의 heartbeat 신호나 질의를 사용하지 않고 기존 스트리밍 패킷과 프로토콜을 이용하였다.

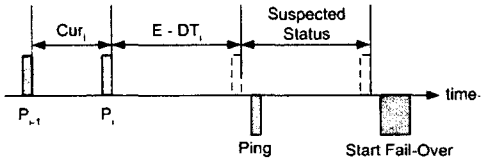

그림 2. E-DT를 통한 Fail-Over의 시작시점

그림 2는 동적 임계점을 이용하여 스트리밍 서버의 고장탐지를 수행한 뒤 Fail-Over의 시작시점까지를 나타내고 있다. 현재 패킷 도착간격을 통해 계산된 E-DT 시간만큼 다음 스트리밍 패킷을 기다리고 있다가 도착되지 않으면 질의 메시지를 전송한다. 질의 메시지는 RTSP의 GET-PARAMETER를 전송한다. 이때 전송속도를 향상시키기 위해 클라이언트에서 전송되었던 메시지를 이용한다. E-DT 시간을 초과하여 스트리밍 패킷이 도착되지 않으면 서버의 고장의심 단계로 전환된다. 스트리밍 서버의 응답메시지를 대기하는 시간은 사용자 정의에 의해 결정되지만 2*E-DT 만큼 적용하면 된다[2]. 만약 고장의심 단계에 스트리밍 패킷이 도착하게 되어도 고장의심 단계는 해제되지 않고 스트리밍 서버의 응답 메시지의 도착에 의해서 고장의심 단계가 해제되어야 한다. 고장의심 단계의 해제절차를 제공함으로써 여러 프록시에서 하나의 서버에 질의 메시지를 동시에 전송하는 것을 방지할 수 있다. 만약 서버에서 응답메시지가 도착함에도 불구하고 스트리밍 패킷이 지속적으로 도착되고 있지 않으면 Fail-Over를 수행해야 한다. 이는 소비자 서비스 수준을 만족하기 위해 필요하다. Fail-Over의 시작시점은 E-DT + Suspected Status time으로 계산되며 마지막 스트리밍 패킷의 정보를 저장하고 있어야 한다. 그림 2의 경우 P_i의 sequence number와 timestamp 및 synchronization source identifier 등의 정보를 저장하고 있어야 한다[5]. 스트리밍 서버가 서비스를 수행하지 못하고 있다고 판단되면 새로운 스트리밍 서버를 선택해야 한다. 프록시에서 각 서버에 대한 세션정보를 활용하여 부하가 적은 서버를 선택하여 새롭게 세션을 맺는다. Fail-Over의 절차는 프록시에서 저장되어 있는 세션정보를 이용하고 스트리밍 패킷에 대한 정보는 마지막에 저장된 RTP 패킷 정보로 수정하여 전송하면 된다[3].

3. 성능 분석

스트리밍 서버의 Fail-Over에 대한 실험은 빠른 고장탐지를 위한 질의 전송 시간(E-DT)과 이를 통해 서버의 서비스 중단을 확인하고 Fail-Over를 시작하는 시간에 대해서만 실시하였다. 실제 Fail-Over에 소요된 시간은 본 논문의 범위를 넘는 것이어서 제외시켰다. 실험환경은 Darwin Server 4.1[6]와 QuickTime player version 6을 사용하였으며 프록시는 리눅스 환경에서 kdeveloper-2.1로 개발되었다. 실험을 간단히 하기 위해 하나의 세션만 사용하였다. 실험환경은 두 실험실에 각각 서버와 프록시를 두고 그 사이에 게이트웨이를 둠으로써 클러스터 스트리밍 서버와 유사하게 구축하였다[그림 3].

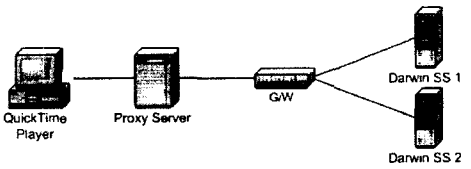

그림 3. 스트리밍 서버의 Fail-Over 실험환경

실험대상인 스트리밍 미디어는 3가지 종류로 표 1과 같다. 각 미디어에는 70k, 69k, 1.2M bits/sec의 평균 비트전송률을 가지고 상영 시간은 30초 100초, 150초의 3종류로 되어 있다.

Media Type	Duration(sec)	Average bitrate (bits/sec)
tomb1.mov	34	68,440
tomb2.mov	"	671,936
tomb3.mov	"	1,161,120
T3_1.mov	96.11	70,040
T3_2.mov	"	699,392
T3_3.mov	"	1,217,952
X2_1.mov	149.02	69,744
X2_2.mov	"	690,960
X2_3.mov	"	1,297,528

표 1. 실험용 스트리밍 미디어

동적 임계점은 초기값을 0, 10000, 50000, 150000 및 990000 (ns) 로 하여 실험을 실시하였다. 동적 임계점의 인자로는 빠른 고장탐지를 의미하는 속도 (Speed)와 질의 전송 횟수를 의미하는 정확도 (Accuracy)가 있다. 속도는 평균 임계점을 평균 패킷 도착 시간으로 나눈 값으로 0에 가까울수록 빠른 고장탐지를 수행하고 있다는 것을 의미한다. 정확도 역시 낮을수록 질의 전송 횟수가 작아 고장탐지의 정확성이 높은 것이다.

그림 4는 최대 패킷 도착 간격(Max)의 속도인자에서 동적 임계점(DT)의 속도인자 차이를 비교하여 나타낸 것으로 기존보다 성능이 향상되었음을 나타내고 있다. 그림 4의 (a)는 전송률이 높고 재생시간이 길수록 동적임계점이 우수한 성능을 나타내고 있다. 그림 4의 (b)는 동적임계점의 초기값이 높을수록 성능이 감소되고 있으며 재생시간이 짧아 동적임계점의 감소가 제대로 이루어지지 않은 경우 최대 패킷 도착 간격을 이용한 임계점 보다 성능이 더 떨어지는 것으로 나타났다.

그림 5는 전송률과 동적임계점의 초기값에 따른 정확도를 최대 패킷 도착 간격과 비교하여 나타내고 있다. 전송률이 높

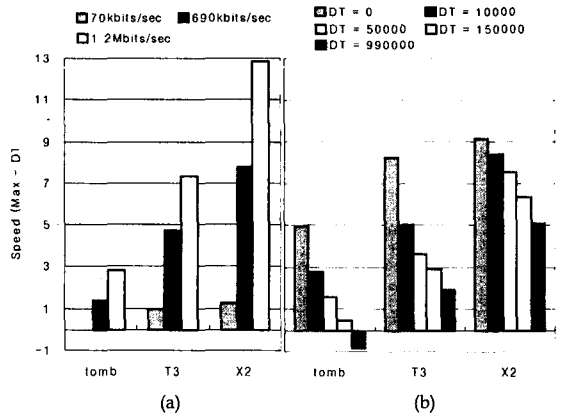

그림 4. 전송률 및 동적 임계점 초기값에 따른 속도 비교

고 동적 임계점의 초기값이 작을수록 질의 전송 횟수가 많음을 알 수 있다. 이것은 초기에 스트리밍 패킷이 전송되면서 임계점이 점차적으로 증가되어 질의 횟수가 많아지는 현상이지만 그림 5의 (b)에서처럼 동적 임계점의 초기값이 평균 패킷 도착 간격보다 큰 150000과 990000 (ns) 일 경우, 이러한 초기 질의 횟수 증가를 방지하였기 때문에 정확도가 좋아진 것을 확인하였다. 전체적으로 최대 패킷 도착 간격보다는 정확도가 낮아 질의 전송 횟수가 많지만 재생시간이 짧으며 동적 임계점의 초기값이 클 경우 상대적으로 정확도가 높음을 알 수 있다. 동적 임계점은 초기값이 평균 패킷 도착 간격보다 크고 미디어의 상영시간이 길며 전송률이 높을 경우 우수한 성능을 나타내고 있다. 따라서 동적임계점 알고리즘이 잘 동작되고 있음을 알 수 있다.

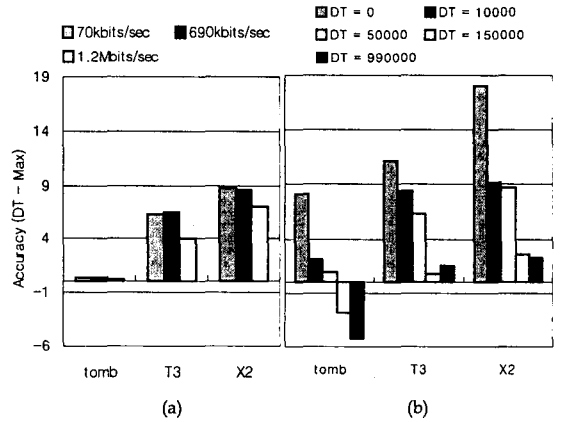

그림 5. 전송률 및 동적 임계점 초기값에 따른 정확도 비교

고장탐지 시간은 현재 패킷 도착지연시간으로부터 임계점을 초과하여 질의를 전송한 뒤 스트리밍 서버의 응답시간이 임계점의 두 배를 초과하는 시점인 Fail-Over 시작 시간까지로 정의 한다. 따라서 고장탐지 시간은 서버와의 시간 동기화 문제로 인해 스트리밍 서버의 정확한 고장탐지 시간을 의미하지 않는다. 실험은 스트리밍 서버가 상영이 시작된 후 약 15초가 경과된 다음 정지시켜 고장탐지 시간을 측정하였다. 표 2는 스트리밍 서버의 고장탐지에 소요된 시간을 최대 패킷 응답시간과 비교하여 나타내고 있다. 미디어의 상영시간이 짧을수록 고장탐지 시간이 짧아지며 동적임계점의 초기값이 0초일 때 가장 빠른 고장탐지를 수행하지만 패킷의 평균 도착시간을 넘는 0.01초보다 클 경우 유사한 고장탐지 시간을 보여주고 있다. 또한 평균 비트 전송률에서는 유사한 고장탐지 시간을 나타내고 있다. 동적임계점을 사용하게 되면 최대 패킷도착간격을 사용했을 때보다 평균적으로 13%이상의 성능향상을 보여준다.

종 류		고장탐지 시간 (sec)		탐지 향상률
		Max	DT	(DT/Max:%)
미디어	tomb	0.3165	0.2498	16
	T3	0.2933	0.2524	11
	X2	0.3326	0.2907	10
DT 초기값 (sec)	0	0.3653	0.2436	26
	0.01	0.3710	0.3248	12
	0.05	0.2553	0.2367	7
	0.15	0.3044	0.2704	10
	0.99	0.2748	0.2461	9
평균 비트전송률 (kbits/sec)	70	0.3151	0.2499	13
	690	0.2876	0.2553	11
	1200	0.3398	0.2877	14
Total		0.3142	0.2643	13

표 2. 스트리밍 서버에 대한 고장탐지 소요 시간

4. 결론

본 논문에서 제안한 개선된 동적 임계점을 이용하면 서버와 클라이언트의 수정 없이 기존 시스템을 그대로 활용하면서 스트리밍 서버의 고장을 빠르게 탐지할 수 있다는 것을 실험을 통해 확인하였다. 또한 스트리밍 서버의 Fail-Over를 직접 수행시켜 동적 임계점에 대한 성능을 재확인 하였다. 향후 과제는 Fail-Over이후 발생되는 스트리밍 미디어에 대한 동기화와 여러 세션에서 발생되는 프록시의 부하 문제가 있다.

참고 문헌

[1] A.Robertson, "Linux-HA Heartbeat System Design," *Proceedings of the 4th Annual Linux Showcase & Conference*, Atlanta, USA, Oct 2000.

[2] Christof Fetzer, " Perfect Failure Detection In Timed Asynchronous Systems," *IEEE Transactions of Computers 2003*, vol.52 No.2 pp.99-112, Feb. 2003.

[3] 전성규, 차호정, "동적임계점을 이용한 스트리밍 서버의 Fail-Over 구조", 한국정보과학회 2003년 춘계학술발표대회 논문집, 2003년 4월.

[4] N.H.Vaidya, "On Checkpoint Latency," *Proc. of the 1995 Pacific Rim International Symposium on Fault-Tolerant Systems*, Newport Beach, CA, USA, pp.60-65, Dec. 1995.

[5] H.Schulzrinne, S.Casner, R.Frederick and V.jacobson, "RTP : A Transport Protocol for Real-Time Applications," RFC 1889, January 1996.

[6] 'QTSS : Quick Time Streaming Server,' URL http://www.apple.com/quicktime/products/qtss/.