한국컴퓨터정보학회 동계학술대회 논문집 제19권 제1호 (2011. 1)

연결 정보를 이용한 P2P 스트리밍 네트워크 구조

이상훈°, 한치근*
°*경희대학교 컴퓨터공학과

A P2P Streaming Network Topology Algorithm Using Link Information

Sang-Hoon Lee°, Chi-Geun Han*
°*Dept. of Computer Engineering, Kyung Hee University

● 요 약 ●

IPTV의 스트리밍 서비스를 위해 P2P를 이용하는 방법이 활발하게 연구되어지고 있다. 이 논문에서는 topology를 최적화하는 방안으로 P2P에서 각 peer 간에 연결 및 전송 정보를 이용하는 방법을 제안한다. 제안하는 방법은 mesh-network에서 각 peer에 연결된 link의 수를 이용하여 업로드 대역폭을 추정하는 알고리즘을 기반으로 한다. 이 방법은 자원의 관리를 위해 업로드 대역폭을 판단하기 위한 메시지 과부하를 효과적으로 줄여주지만 스트리밍에서 주어진 연산만을 수행할 경우 업로드 대역폭과 무관한 형태로 network topology가 잘못 구성될 가능성을 가지고 있다. 본 논문에서는 기존 연구에서 부족했던 부분들을 정리하고 극복할 수 있는 각각의 알고리즘들과 적용했을 시에 예상되는 결과를 제시한다.

키워드: IPTV(IPTV), P2P 스트리밍(P2P Streaming), 토폴로지 최적화(Topology Optimization)

I. 서론

인터넷의 발전과 다양한 형태의 단말이 나타남으로 인해 라이브 미디어 스트리밍을 이용한 IPTV 서비스에 대한 연구가 활발하게 이뤄지고 있다.

스트리밍을 위한 대안 중 전통적인 클라이언트-서버 방식이나 IP 멀티캐스트 방식은 각각 확장성이 낮거나 새로운 장비를 도입하는 추가비용이 크게 발생하여 네크워크 상에 도입하기가 어렵다. 이에 반해 P2P 방식은 사용자들의 자원을 공유하여 확장성이 좋고, 자원 활용률이 좋다는 장점을 가진다. 하지만 P2P 방식을 활용하는 데에도 몇 가지 단점이 존재한다.

P2P 방식에서는 각 peer의 자원이 이질적이어서 각각에 대해 모두 서비스 품질을 만족시키기가 어려우며, peer의 편입 및 이탈이 잦아서 topology의 변경이 자주 일어난다. 또한 스트리밍 환경일 경우 지연시간에 대해서 일반적인 P2P 파일 전송 조건보다 더 엄격한 제한을 가지게 된다[1-3].

이 논문에서는 topology를 최적화하는 방안으로 P2P 방식에서 각 peer 간에 연결 및 전송 정보를 이용하는 방법을 제안한다. 제안하는 방법은 mesh-network에서 각 peer에 연결된 link의 수를 이용하여 업로드 대역폭을 추정하는 알고리즘을 기반으로 한다.

II. 관련 연구

P2P 스트리밍을 위한 topology 구성 알고리즘 관련 연구로서 다양한 의견들이 제시되고 있다. 그중 특히 mesh-network의 확장성 및 견고성과 tree-network의 효율성이라는 장점을 모두 얻기 위해 hybrid 형태를 취하는 경우가 있다. [그림 1]과 같이 일정 수준 이상 연결을 유지하여 network 신뢰성이 보장된 peer들은 source에 가까운 위치에 tree형태로 연결시키고 연결이 오래 되지 않은 peer들은 tree의 leaf부분에 mesh형태로 연결을 시키는 hybrid network를 구성하는 방안[4]이 제시된 바 있다.

이를 기초로 각 peer들을 유사한 지역별로 묶이도록 서버를 설정하여 메시지 오버 헤드를 줄이고, 업로드 대역폭이 넓은 peer들을 분산되도록 제한하여 자원 편중 현상에 대해서 조절할 수 있도록 하는 방법이 추가로 연구되기도 하였다[5].

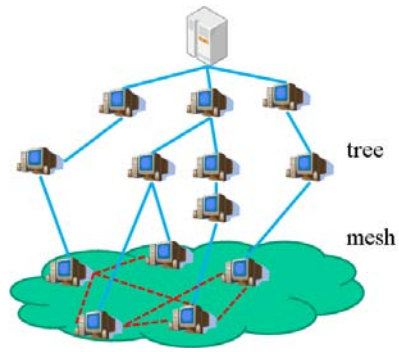

그림 1. 하이브리드 network의 예
Fig. 1. an example of hybrid network

hybrid가 아닌 mesh-network를 기본으로 한 연구도 있다.

Adaptive Overlay 알고리즘[6]은 [그림 2]에서와 같이 각 peer에서 연결된 link의 수와 사용여부만을 이용하여 업로드 대역폭을 추정하는 방안을 제시했다.

논문의 결과를 보면 실제 업로드 대역폭을 기준으로 만들어지는 topology와 거의 근접하면서 메시지 오버헤드를 크게 감소시키고 단순한 형태의 알고리즘으로 구현되어질 수 있다는 점에서 주목할 가치가 있다.

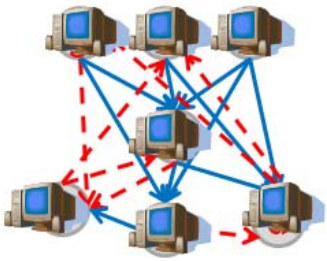

그림 2. used와 unused로 link를 포함하는 mesh network 예
Fig. 2. an example of mesh network including used and unused links

III. 본론

1. 문제 정의

Adaptive Overlay 알고리즘의 동작은 Algorithm 1에 기술되어 있다. 이 알고리즘은 미리 정해진 αL과 αH라는 두 비율을 이용하여 peer p의 unused link 비율 card($\overline{U}$(p))을 used link card(U(p))와 두 비율의 곱 사이의 값으로 유지한다. card($\overline{U}$(p))이 αLcard(U(p)) 보다 낮아지면 peer p가 업로드 대역폭이 많이 남았다고 추정한다. 따라서 자신의 child peer 집합 N0(p)와 연결된 peer들의 집합 C(p)에서 현재 link수가 가장 많은 peer n+개 선택하여 추가로 link를 연결한다.

```
Algorithm 1: Adaptive Overlay Algorithm
for every δc received chunks do
/* scan and adjust p's neighbourhood */
  if (card(Ū(p)) < αL card(U(p)) then
/* grow the neighbourhood */
    Select n+ neighbors from (C(p) − N0(p) − {p})
    Add the n+ chosen neighbours to N0(p))
  end
  else if (card(Ū(p)) > αH card(U(p)) then
/* shrink the neighbourhood */
    Select n- neighbors from U(p) for culling
    Cull the n- neighbours from N0(p)
  end
  else
/* p's neighbourhood does not change */
  end
end
```

반대로 card($\overline{U}$(p)이 αHcard(U(p))보다 낮아지면 요구량이 p의 업로드 대역폭을 넘었다고 추정하여 N0(p)중 현재 link수가 가장 적은 peer n-개를 선택하여 연결을 제거한다.

이 알고리즘은 스트리밍에서 주어진 연산만을 수행할 경우 업로드 대역폭과 무관한 형태로 network topology가 잘못 구성될 가능성을 가지고 있다. [그림 3]처럼 source와 인접한 peer와 인접하지 않은 peer 양쪽에게 전송을 받는 child peer가 있다고 가정한다. 그 child peer는 이미 더 빠른 peer에게 전송을 받고 있기 때문에 인접하지 않은 peer의 업로드 대역폭 여부와는 상관없이 그 peer와 child peer간의 연결은 언제나 사용되지 않게 된다.

이러한 사용되지 않는 연결은 그 peer가 업로드 대역폭에 있어서 포화상태에 도달했다고 잘못 추정하여 network topology를 망가뜨릴 수 있다.

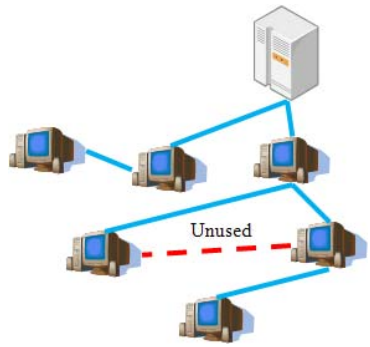

그림 3. 스트리밍에서의 unused link 문제
Fig. 3. unused links problem in streaming

제안하는 방법에서는 연결 정보로부터 업로드 대역폭을 추정하여 오버헤드를 줄이는 방법을 이용한다. 다만 기존의 방법에서처럼 매 순간의 link수만을 이용하지 않는다. 대신 각각의 peer마다 queue를 두어서 일정한 시간동안 사용되었던 link 수를 저장하도록 한다.

이 queue에 저장된 값(이하 기여도)들의 합을 이용하여 peer의 연결을 조절함으로써 일정 기여도 이상인 peer들을 source에 가깝도록 tree 형태로 배치하고 기여도가 낮은 peer들은 tree 하단의 mesh topology에 묶이도록 하는 것이 가능해진다.

2. 제안하는 P2P topology 구성 방법

2.1 변수 정의

이 논문은 미디어 콘텐츠를 제공하는 source peer와 각 peer들이 network상에서 P2P로 연결되어 있는 환경을 가정한다. 각각의 peer들은 source로부터 스트리밍을 통하여 데이터를 실시간으로 얻게 된다.

제안하는 방법의 알고리즘 기술에 관한 변수들의 정의는 아래와 같다.

L(p) : peer p에 연결된 link의 총 갯수
Cq : 각 peer별로 기여도를 저장할 순환 큐. 모두 0으로 초기화.
I : 각 peer별로 turn이 돌아왔을 때 큐에 저장할 위치.
sum : 순환 큐에 저장된 기여도의 총합
$\overline{U}$peer(p) : peer p와 unused-link로 연결된 peer들의 집합
$\overline{U}$link(p) : peer p와 연결된 unused-link의 집합
Ulink(p) : peer p와 used-link로 연결된 peer들의 집합
Upeer(p) : peer p와 연결된 used-link의 집합
sumL : 해당 peer가 확장이 가능하다고 판단하는 최소 기여도 합
sumt : 동적, 정적 topology를 결정하는 최소 기여도합
αL, αH: 확장과 축소를 결정하는 used link의 비율 변수
dL, dH: 동적 형태 일 때 αL, αH에 대입되는 상수.
β : 안정화 형태 일 때 αL, αH에 대입되는 상수. 0에 가까운 최소값.

2.2 기여도를 적용한 topology 구성 알고리즘

```
Algorithm 2: algorithm based on contribution principle
for(every δc received chunks) do
  /* 이번 turn의 used link를 Queue에 갱신 */
  p.sum += card(Ulink(p)) – p.Cq[p.I]
  p.Cq[p.I] = card(Ulink(p))
  p.I = ++p.I % MAX_Queue
  end
  /* scan and adjust p's neighborhood */
  if( p.sum < sumt ) then
    αL= β;αH= β; else αL = dL; αH = dH;
  if( card( Ulink(p) ) > αHcard(Ulink(p) ) then
    /* shrink the neighborhood */
    Algorithm 3 : Shrink algorithm
    end
  else if( card( Ulink(p) ) < αLcard(Ulink(p) ) then
    /* grow the neighborhood */
    if(repeat shrink and growth) then
      continue;
    else
      Algorithms 4: Growth algorithm
    end
```

전체 알고리즘을 pseudo code로 나타내면 algorithm 2와 같다. 제안하는 방법에서는 추가로 hybrid-network를 구성하여 전술한 장점들을 얻고, Adaptive Overlay 알고리즘에서 발생한 문제점을 해결할 수 있도록 기여도 연산을 수행한다.

기여도를 기준으로 하는 topology 배치는 queue의 크기를 조절함으로써 신뢰도와 업로드 대역폭 추정량 중 어느 쪽에 더 높은 비중을 두고 topology를 구성할 것인지 쉽게 조절이 가능하다.

2.3 Shrink 알고리즘

```
Algorithm 3: Shrink algorithm pseudo code
/* 기여도 순으로 n- 개의 peer를 선택하여 연결 제거 후 이웃 peer에 편입 시킴*/
selection_queue = sort_by_ascending_use_sum( Upeer(p))[0..n-]
  remain_neighborhood = {N0(p) - selection_queue - {p} }
  i = 0, j = 0
  for(; j < sizeof(remain_neighborhood); j++) then
    if(remain_neighborhood[j].sum > sumL)
      potential_parent_queue .add(remain_neighborhood[j])
    end
  end
  /* 선택된 이웃 peer의 수가 지나치게 적다면 전체 이웃 peer에 고르게 편입. */
  if (sizeof(potential_parent_queue ) < threshhold_num ) then
   remain_neighborhood =
   sort_by_descending_use_sum(remain_neighborhood)
   while( i < n- ) do
     add selection_queue [i] to N0(remain_neighborhood[i])
   end
 else
  /* 기여도 합을 이용하여 기여도가 높은 peer 우선으로 편입되도록 함. */
 sort_by_descending_use_sum( potential_parent_queue )
 while( i < n- ) do
  add selection_queue [i] to
  N0(potential_parent_queue[i++%size(potential_parent_queue)])
  end
 end
end
```

제안하는 방법에서 shrink 연산을 수행할 때는 기존의 방법과 달리 peer에서 선택된 link들을 제거하지 않는다. 그래서 child peer에 인계하여 연결이 소실되는 부분에 대해서 따로 조건적으로 보장할 필요가 없어졌다. 세부적인 알고리즘은 algorithm 3에서 자세히 다루고 있다.

본 논문에서는 기존의 알고리즘에 기여도라는 개념을 추가한다. 이로써 mesh 형태만이 아닌 hybrid 형태를 가지는 것이 가능하도록 유도하였고, network 구성시 알고리즘에서 의도치 않았던 오류가 발생할 가능성을 사전에 방지할 수 있었다.

2.4 Growth 알고리즘

```
Algorithm 4: Growth algorithm pseudo code
/* 기여도 순으로 n+ 개의 peer를 선택 */
selection_Queue =
sort_by_descending_use_sum(C(p)- N0(p) -{p})[0..n+]
/* 만약 선택된 peer들 중 가장 기여도의 합이 큰 peer가 현재 peer의 기여도 합보다 크다면 위치 교환 */
  if( selection_Queue [0].sum > p.sum ) then
    add selection_Queue[0] to p.parent
    add p to N0(selection_Queue[0])
  end
  add selection_Queue to N0(p)
end
```

제안하는 방법에서 Growth 연산을 수행할 때 차이점은 algorithm 4에 표현된 것처럼 새로 편입되는 child들 중 parent peer보다 더 큰 기여도를 가지는 peer가 있다면 더 source에 근접하도록 능동적으로 연결을 바꾼다는 점이다.

기존의 방법과 비교했을 때 이러한 능동적인 위치 변경은 업로드 대역폭 추정치가 큰 peer들이 source에 근접하는 시간을 훨씬 단축시키는 결과를 보일 것으로 예상된다.

IV. 결론

본 논문에서는 link 정보를 이용하여 업로드 대역폭을 추정함으로써 메시지 과부하를 줄이고, 기여도를 저장하여 신뢰도를 판단함으로써 hybrid-network의 이점을 얻을 수 있도록 제안하였다. 또한 child peer가 기여도가 클 경우 서로의 연결을 교환함으로써 업로드 대역폭 추정치가 큰 peer들이 보다 빨리 source에 근접하도록 유도하여 스트리밍 상황에서 발생하는 부적절한 network 형태 가능성을 차단하였다.

추후에는 실제적인 성능평가를 거쳐 알고리즘을 검증하고, 특수한 상황을 보장하기 위한 chunk scheduling strategy를 연구할 것이다.

참고문헌

[1] F. Thouin, and M. Coates, “Video-on-Demand Networks: Design Approaches and Future Challenges,” IEEE Networks, pp.42-48, March/April 2007.

[2] W.-P. Ken Yiu, X. Jin, and S.-H. Gary Chan, “Challenges and Approaches in Large-Scale P2PMedia Streaming,” IEEE Multimedia, pp.50-59, April-June 2007.

[3] D.-E. Meddour, M. Mushtag, and T. Ahmed, “Open Issues in P2P Multimedia Streaming,” Proceedings of MultiComm'06, pp.43-48, June, 2006.

[4] Wang, Y. Xiong, and J. Liu, “mTreebone : A Hybrid Tree/Mesh Overlay for application-layer live video multicast,” Proceeding of IEEE ICDCS'07, May, 2007.

[5] Haesun Byun, Meejeong Lee “A Hybrid P2P Overlay Architecture for Live Media Streaming”, Journal of Korea Information Science Society, Vol.36, No.6, pp. 481~491, Dec. 2009.

[6] R. Lobb, A. P. Couto da Silva, E. Leonardi, M. Mellia, and M. Meo, “Adaptive overlay topology for mesh-based p2p-tv systems,” in ACM NOSSDAV, Williamsburg, Virginia, USA, June 2009.