멀티캐스트 지연변이 문제에 대한 효율적인 코어 선택 추정 알고리즘

The Algorithm of Efficient Core Selection Estimation for the Multicast Delay Variation Problem and Minimum Delay Variation

안 영 진* 김 문 성** 추 현 승***
Youn-gjin Ahn Moon-seong Kim Hyun-seung Choo

요 약

멀티캐스트 기술이 발전함에 따라 멀티캐스트 라우팅을 지원하는 실시간 그룹 어플리케이션들이 정점 증가하고 있으며 더욱 중요하게 되어가고 있다. 실시간 어플리케이션의 가장 중요한 요소 중의 하나는 지연변이 문제가 있으며 이는 DVBMT(Delay- and delay Variation-Bounded Multicast Tree) 문제로 잘 알려져 있다. DVBMT 문제란 제한된 시간 내에 종단간의 지연과 지연변이를 만족시키는 것을 뜻한다. DVBMT는 NP-Complete 문제로 알려져 있으며, 이를 풀기위한 시도로 DVMA(Delay Variation Multicast Algorithm), DDVCA(Delay and Delay Variation Constraint Algorithm) 등이 제안된 바 있다. 본 논문에서는 이러한 알고리즘보다 더욱 효율적인 알고리즘을 제안한다. 성능평가를 통한 본 알고리즘의 효율성은 DDVCA보다 더욱 높은 것으로 평가되었으며 그 효율 정도는 9%~29%의 성능향상을 보인다. 본 논문의 시간복잡도는 DDVCA와 같은 $O(mn^2)$ 이다.

Abstract

With the development of the multicast technology, the realtime strategy among the group applications using the multicast routing is getting more important. An essential factor of these real-time application is to optimize the Delay- and delay Variation-Bounded Multicast Tree (DVBMT) problem. This problem is to satisfy the minimum delay variation and the end-to-end delay within an upper bound. The DVBMT problem is known as NP-complete problem. The representative algorithms for the problem are DVMA, DDVCA, and so on. In this paper, we show that the proposed algorithm outperforms any other algorithm. The efficiency of our algorithm is verified through the performance evaluation and the enhancement is up to about 9% to 29% in terms of the multicast delay variation. The time complexity of our algorithm is $O(mn^2)$.

☞ Keyword : DVBMT 문제(Delay- and delay Variation-Bounded Multicast Tree Problem), 지연변이(delay variation), DDVCA (Delay and Delay Variation Constraint Algorithm)

1. 서 론

최근 유니캐스팅, 브로드캐스팅, 멀티캐스팅은 다중접속 네트워킹 어플리케이션을 지원하기 위한 방법들이다. 유니캐스트 어플리케이션은 하나의 패킷을 전송자가 각각의 그룹 멤버들에게 전송하는 것이며, 브로드캐스트 어플리케이션은 전송자가 하나의 패킷을 보내면 브로드캐스트 주소를 통해 모두 보내는 것이다. 하지만 멀티캐스트 어플리케이션은 하나의 패킷이 통신을 원하는 멤버들에게 복사되어 가는 것을 말하며, 이는 통신 인프라에서 점점 중요하게 여겨지고 있는 실정이다. 멀티캐스트 통신에서 QoS(Quality of Service)는 사용자의 요구, 실시간 환경, 고속 네트워크

* 준 회 원 : LG전자 정보통신사업본부 단말연구소 연구원 yjahn@ece.skku.ac.kr
** 정 회 원 : 성균관대학교 정보통신공학부 연구교수 moonseong@ece.skku.ac.kr
*** 종신회원 : 성균관대학교 정보통신공학부 부교수 choo@ece.skku.ac.kr (교신저자)
[2006/08/16 투고 - 2006/09/01 심사 - 2006/12/03 완료]

등을 모두 만족시키는 것을 말한다. 따라서 QoS 멀티캐스트는 항상 중요한 문제로 대두되고 있다.

QoS 전송을 위해서 멀티캐스트 트리 생성이 중요한 역할을 한다. 효율적인 멀티캐스트 트리 생성 문제는 NP-complete이며, 이러한 문제를 기반으로 하는 많은 휴리스틱들[1, 2, 3]이 제안되었다. 대역폭 효율성을 따지는 저비용 멀티캐스트 트리를 구성하는 것 뿐만 아니라 실시간 전송을 지원하는 서비스 네트워크에서 전송지로부터 메시지를 전달할 때에 한정된 시간 내에 받을 수 있도록 만드는 것 또한 매우 중요한 요소로 생각할 수 있다. 만약 전송지로부터의 메시지가 시간 내에 원하는 목적지들까지 도달하지 못한다면 이는 의미 없는 정보가 되기 때문이다. 따라서 종단간 지연시간을 만족하기 위해 제한 시간을 보장하는 중요한 방법이 필요하다. 이를 멀티캐스트 종단간 지연 문제라 부른다[4].

멀티캐스트 네트워크에서 실시간 어플리케이션을 고려하는 또 하나의 중요한 요소는 지연변이 문제이다. 멀티캐스트를 이용한 화상회의 중에 발표자의 말은 동기를 맞추어 모든 참여자에게 전달해야 한다. 그렇지 않으면 통신상에서 상호간 화상으로 회의를 한다는 느낌을 받을 수 없을 것이다. 더 나아가 온라인 비디오 게임의 상황을 고려해 볼 때, 게임 속의 캐릭터가 동시에 움직여야 하는 문제를 지니고 있는 상황에서 캐릭터들이 서로 동기를 맞추어야 할 필요가 있다. 이러한 모든 문제들은 지연변이 문제[5]에 해당하는 것으로서 제한된 시간 내에 패킷들이 거의 동시에 전달되어야만 한다.

본 논문에서는 멀티캐스트에서의 종단간 지연 문제 및 지연변이 문제에 대한 효율적인 알고리즘을 제안한다. 지금까지 가장 효율적이라고 알려진 DDVCA(Delay and Delay Variation Constraint Algorithm)[6]와 비교해 볼 때, 제안하는 알고리즘은 보다 지연변이 면에서 우수한 성능을 보이며, 시간복잡도 면에서는 $O(mn^2)$으로 DDVCA와 같다. 본 논문은 다음과 같이 구성한다. 2장에서는 멀티캐스트 라우팅에 대한 네트워크 모델링 등 관련된 연구를 보인다. 3장은 제안하는 알고리즘을 상세히 설명한다. 그런 후, 4장에서 시뮬레이션을 통한 성능측정을 하고, 마지막으로 5장에서 본 논문의 결론을 제시한다.

2. 관련연구

2.1 네트워크 모델

n개의 노드와 l개의 링크를 지닌 그래프 $G=(V,E)$에 의해서 표현된 네트워크를 생각한다. V는 노드들의 집합이며, E는 링크들의 집합이다. 각 링크 $e=(i,j)\in E$는 지연시간 $d(e)\geq 0$을 갖는다. 하나의 링크가 갖는 지연시간은 그 링크의 대기지연, 전송지연, 그리고 전파지연의 합이다. 노드 u에서 v까지 경로를 $P(u,v)=\{(u,i),(i,j),\cdots,(k,v)\}$로 나타내자. 주어진 시작노드 $s\in V$와 목적노드 $d\in V$에서, $(2^{s\rightarrow d},\infty)$는 s에서 d까지 가능한 모든 경로들의 집합이다.

$(2^{s\rightarrow d},\infty)=\{P_k(s,d) \mid s$에서 d까지 가능한 모든 경로들, $^{\forall}s,\ d\in V,\ ^{\forall}k\in\Lambda\}$

여기서, Λ는 첨자들의 집합(Index Set)이다. 즉 시작노드와 목적노드의 모든 경로는 P1(s, d), P2(s, d), ..., Pn(s, d) ($1\leq k\leq n$)로 나타낼 수 있다. 어떠한 경로 P_k의 경로 지연시간은 다음과 같이 주어진다.

$$\Phi_D(P_k)=\sum_{e\in P_k}d(e)\ ,\ ^{\forall}P_k\in(2^{s\rightarrow d},\infty)$$

$(2^{s\rightarrow d},\Delta)$는 s에서 d까지 종단간 지연시간이 Δ 지연 제한을 만족하는 경로들의 집합이다. 그러므로 다음과 같이 $(2^{s\rightarrow d},\Delta)\subseteq(2^{s\rightarrow d},\infty)$이다.

멀티캐스트 통신에서 메시지는 멀티캐스트 그룹(M, $|M|=m$)이라 하는 $M\subseteq V\backslash\{s\}$의

수신자들에게 전달되어진다. 시작노드 s에서 멀티캐스트 수신자 $m_i \in M$으로 메시지가 지나가는 경로는 $P(s, m_i)$로 표현한다. 따라서, 멀티캐스트 라우팅 트리는 $T(s, M) = \bigcup_{m_i \in M} P(s, m_i)$로 정의 하고 메시지는 $T(s, M)$을 사용하여 s에서 목적노드들 M으로 전송한다.

2.2 DVBMT 문제

멀티캐스트 통신에서 중요한 두개의 QoS 측정 기준을 소개한다[5]. 멀티캐스트 종단간 지연시간 제한인 Δ는 시작노드에서 목적노드로의 경로가 가지는 종단간 지연의 허용할 수 있는 상한선을 나타낸다. 이 측정 기준은 멀티캐스트 메시지로 다루어지는 정보가 시작노드에서 보내진 후 Δ 시간 후에는 효력을 상실함을 의미한다.

멀티캐스트 지연변이 δ는 시작노드 s에서 각각의 목적노드들 $m_i \in M$과의 경로 지연시간들 간의 최대 차이이다.

$$\delta = max \{ | \phi_D(P(s,m_i)) - \phi_D(P(s,m_j)) | , \forall m_i, m_j \in M, i \neq j \}$$

는 [5]에 처음으로 정의되었고 토론된 문제인 종단간 지연시간 제한을 만족하면서, 멀티캐스트 지연 변이를 최소화 하는 DVBMT 문제는 다음을 만족하는 트리를 찾는 것이다.

$$min \{ \delta_\alpha \mid \forall m_i \in M, \forall P(s,m_i) \in (2^{s \to m_i}, \Delta), \forall P(s,m_i) \subseteq T_\alpha , \forall \alpha \in \Lambda \}$$

여기서 T_α는 $M \cup \{s\}$를 포함하는 멀티캐스트 트리를 나타내며, DVBMT 문제는 NP-complete로 알려져 있다.

2.3 선행 알고리즘

DVBMT 문제를 해결하기위해 잘 알려진 두 개의 멀티캐스트 트리생성 알고리즘이 있다. 하나는 DVMA(Delay Variation Multicast Algorithm) [5] 이고 또 다른 하나는 DDVCA(Delay Variation Constraint Algorithm) [6]이다. Rouskas와 Baldine가 제안한 DVMA는 각각의 노드가 전체 네트워크 토폴로지를 알고 있다는 가정에서 진행한다. 이 알고리즘은 지연제약조건을 만족하는 노드들로 이루어진 신장트리(spanning tree)를 우선생성하기에 제약조건을 거스르는 일부 목적노드들은 트리에 포함이 안 될 수도 있다. 그 후에, 현 생성된 트리와 트리에 존재하지 않는 노드들 간에 지연시간과 지연변이를 만족하는 경로를 찾아서 트리에 추가시키는 과정을 거친다. 이때 만족스런 경로를 찾기 위해서 k-최단경로알고리즘이 사용된다. Pi-Rong Sheu와 Shan-Tai Chen이 제안한 DDVCA 알고리즘은 코어 기반 트리(CBT: Core based Tree)[7]에 바탕을 두고 있다. DDVCA는 우선 $m_i \in M$, $v \in V$에 대해 모든 (m_i, v) 쌍의 최소 지연시간들을 계산한다. 그리고 DDVCA는 각 노드에 대해서 그 각각의 노드와 모든 목적노드들 사이의 지연 변이 δ_α들을 계산한다. 그런 후에, 최소 지연변이 $\delta = \{\delta_\alpha \mid \alpha \in \Lambda\}$를 갖는 노드를 코어(Core)노드로 선택한다. 최종적으로, 각 목적노드는 최소 지연 경로를 통해 이 코어노드에 연결된다. 시작노드 역시 최소 지연 경로를 통해 이 코어노드에 연결된다. 만약 이 코어노드가 Δ 제한 지연을 만족하지 못한다면 DDVCA는 δ 다음으로 가능한 최소 지연변이 코어노드를 선택하고 위와 동일한 과정이 Δ 요구 조건을 만족할 때까지 반복한다.

생성한 트리의 지연변이의 관점에서, DDVCA가 DVMA보다 미세하게 우수하다는 것은 이미 알려져 있다. 그러나 시간 복잡도는 m을 수신자(목적노드)의 수, n을 네트워크의 총 노드 수로 보았을 때 DDVCA는 $O(mn^2)$이고 DVMA는 $O(klmn^4)$이다. 여기서 k와 l은 $k^{th}(l^{th})$ 최단경로 알고리즘을 사용할 때의 상수 값이다.

3. 제안하는 알고리즘

본 장은 DDVCA보다 우수하며 멀티캐스트 지연변이를 최소화 시킬 수 있는 멀티캐스트 트리를 생성하는 새로운 알고리즘을 제안한다. 제안한 알고리즘의 기본적인 아이디어는 CBT[7]에 바탕을 두고 있다. CBT에서 멀티캐스트 트리를 생성하기 위하여 사용하는 방법은 백본을 구성하는 코어 라우터를 선택하는 것으로 시작한다. 또한 본 장에서는 DDVCA와 유사한 개념을 이용하지만 코어의 선택 면에서 그보다 더욱 효과적이라는 것을 보일 것이다.

3.1 ESC 알고리즘 설명

본 논문에서는 최소의 멀티캐스트 지연변이를 갖는 멀티캐스트 트리를 생성하는 ESC(Estimation of Selecting Core) 알고리즘을 제안한다. ESC 알고리즘은 코어노드의 선정과 멀티캐스트 트리를 생성하는 두 부분으로 이루어진다. 코어노드 선정에 있어서 코어노드의 후보노드가 여러 개일 때, DDVCA는 후보노드 중에서 임의로 코어노드를 선택하지만 ESC 알고리즘은 코어의 선택에 있어서 여러 가지 경우를 제시한다. 여기에서 MODE 함수를 정의할 수 있는데, 이는 코어노드의 위치가 멀티캐스트 지연변이에 영향을 주는 경우를 구분한 함수이다. MODE 함수에 덧붙여서 CMP 함수라고 부르는 각 모드에 대한 측정값을 구하는 함수 또한 소개한다. MODE 함수와 CMP 함수는 다음과 같이 정의한다.

<그림 1>의 의사코드(pseudo code)에서 최소 지연시간 경로는 다익스트라(Dijkstra) 알고리즘을 사용한다. 3-5단계에서 ESC 알고리즘은 소스노드에서 모든 노드로 가는 최소 지연시간을 계산한다. 이때, 네트워크 상에 있는 모든 노드에서 소스로의 최소 지연 경로를 거쳐 가는 노드들을 ds_pred를 통해 알 수 있다. 6-8단계에서는 각 목적 노드에서 네트워크 상에 있는 모든 노드로의 최소 지연 시간을 계산한다. 여기서는 모든 노드에서 각 목적노드로 거쳐 가기 위해 이전 노드로의 dest_pred를 구하여 저장한다. 9-21단계는 적절한 코어노드를 찾기 위한 과정이다. 우선, 9-15단계에서 각 노드에서 목적노드들로 생성되는 최소 지연 경로 중에 가장 큰 지연과 가장 작은 지연의 차이 값을 비교하여 그 값이 가장 작은 dv_{min}을 선정하고 또한 Δ를 만족하는지 확인한다. 만약 그러한 코어가 없다면 알고리즘은 트리를 생성할 수 없으므로 종료한다. 16-18단계는 선정된

$$MODE(c) = \begin{cases} \text{I} & \text{if } c = s \\ \text{II} & \text{if } {}^{\exists} m \in P(s,c), where\ {}^{\forall} m \in M \\ \text{III} & \text{if } {}^{\exists} s \in P(m,c), where\ {}^{\forall} m \in M \\ \text{IV} & \text{if II and III} \\ \text{V} & \text{otherwise} \end{cases}$$

, where s 는 소스노드.

$$CMP(x) = \begin{cases} |(ds_{core} + \max\_delay) - ds_{m_k}| & \text{if } x \text{ is II or III or IV} \\ dv_{core} & \text{otherwise} \end{cases}$$

, where $core = MODE^{-1}(x)$,
$\max\_delay = \max\{\min\{\phi_D(P(core,m))\} | {}^{\forall} m \in M \wedge \{m^* \in M |\ {}^{\exists} m^* \in P(s,core) \vee {}^{\exists} s \in P(m^*,core)\}\}$. ds_{v_i}와 dv_{v_i}는 <그림 1>에 정의되어있다.

가장 작은 dv_{min} 와 같은 값을 지닌 후보 코어를 탐색하고, candidate에 후보 코어들을 저장하게 된다. 19-21단계에서는 후보 코어들 중에서 가장 적합한 코어를 찾게 된다. 여기서 CMP라는 비교 지표가 나타나는데, 이는 위에서 설명한 5가지 MODE의 CMP값을 나타낸다. 각 MODE에 따른 자세한 설명은 다음 장에 기술되어있다. 22-23단계는 선택된 코어노드를 중심으로 하여 멀티캐스트 트리가 형성이 되고, 24단계에서는 코어노드와 소스노드의 경로를 잇는다. 마지막으로 25단계에서 멀티캐스트 트리 $T(s, M)$ 을 생성한다.

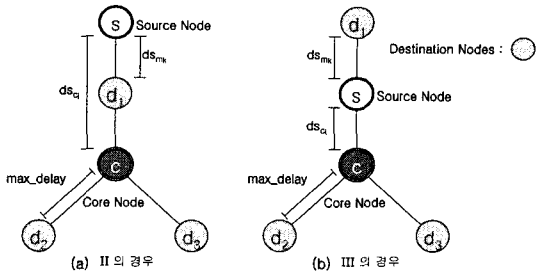
Proposed Algorithm $(G(V.E).\ M,\ s,\ \Delta)$
Input: A directed graph $G(V, E)$, M is the multicast group with $m = |M|$, a source node s, a end-to-end delay bound Δ.
Output: The multicast tree T such that $\phi_D(P(s, m_i)) \le \Delta$, $^{\forall}P(s, m_i) \subseteq T$, $^{\forall}m_i \in M$, and has a small multicast delay variation.

01. **Begin**
02. $dv_{min} = \infty$; $core = \emptyset$; $candidate = \emptyset$; $compare = \infty$; $T = \emptyset$;
    /* *candidate* : candidates of a core node */
    /* *compare* : measure for the 5 modes */
03. **For** $^{\forall}v_i \in V$ **Do**
04. $ds_{v_i} = min\{\varphi_D(P) \mid {}^{\forall}P \in (2^{s \to v_i}, \infty)\}$;
05. $ds\_pred_{v_i} = v$ s.t. $(v, v_i) \in P(s, v_i)$ with ds_{v_i};
06. **For** $^{\forall}m_k \in M$ **Do**
07. $ppd(m_k, v_i) = min\{\phi_D(P) \mid {}^{\forall}P \in (2^{m_k \to v_i}, \infty)\}$, $^{\forall}v_i \in V$;
08. $dest\_pred(m_k, v_i) = v$ s.t. $(v, v_i) \in P(m_k, v_i)$ with $ppd(m_k, v_i)$. $^{\forall}v_i \in V$;
09. **For** $^{\forall}v_i \in V$ **Do**
10. $max_i = max\{ppd(m_k, v_i) \mid {}^{\forall}m_k \in M\}$;
11. $min_i = max\{ppd(m_k, v_i) \mid {}^{\forall}m_k \in M\}$;
12. $dv_{v_i} = max_i - min_i$;
13. **If** $dv_{v_i} < dv_{min}$ and $ds_{v_i} + max_i \le \Delta$
14. **then** $dv_{min} = dv_{v_i}$; $core = v_i$
15. **If** $core = \emptyset$ **then** 끝
16. **For** $^{\forall}v_i \in V$ **Do**
17. **If** $dv_{v_i} = dv_{core}$ and $ds_i + max_i \le \Delta$
18. **then** $candidate = candidate \cup v_i$;
19. **For** $^{\forall}c \in candidate$ **Do**
20. **If** $CMP(MODE(c)) < compare$
21. **then** $compare = CMP(MODE(c))$; $core = c$;
22. **For** $^{\forall}m_k \in M$ **Do**
23. $T = T \cup P(core, m_k)$;
24. $T = T \cup P(s, core)$;
25. Return T
26. **End Algorithm.**

〈그림 1〉 ESC 알고리즘

3.2 5가지 MODE에 대한 코어선택 기준

앞서 제시한 MODE와 CMP 함수와 같이, MODE I은 선택한 코어노드가 소스노드와 일치할 경우이다. 첫 번째 MODE가 아니라면 MODE Ⅱ~Ⅴ의 경우이다.



〈그림 2〉 MODE 함수의 주요 설명

<그림 2>의 (a)는 MODE Ⅱ의 경우[8]로서 소스노드에서 코어노드까지의 경로에 목적노드가 존재하는 경우이다. 여기서 CMP의 값은 코어에서 지나친 목적노드의 지연 값($ds_{c_i} - ds_{m_k}$)과 코어에서 가장 큰 지연 값(max_delay)을 합한 것이다. 다음은 MODE Ⅲ의 경우이다. CMP의 값은 위의 <그림 2>의 (b)와 같이 소스노드에서 코어노드까지의 지연 값과 코어에서 가장 큰 지연을 더한 값($ds_{c_i} + max\_delay$)에 소스노드와 소스노드에 인접한 목적노드 지연값(ds_{m_k})의 차이이다. 네 번째 경우는 MODE Ⅳ의 경우이다. Ⅱ와 Ⅲ을 모두 만족하는 경우, 소스를 중심으로 목적노드들이 있으므로 CMP를 결정하는 요소를 찾아야 한다. 그 비교 대상은 소스에서 목적노드들까지의 최소 지연 경로 값을 비교하는 것이다. 결국 소스로부터 가장 가까운 지연 값이 이후 생성된 트리의 지연 변이에 영향을 주기 때문에 이러한 비교로서 판단을 할 수 있다. 따라서 비교 후에 MODE Ⅱ나 Ⅲ 둘 중에 한 경우를 선택하게 된다. MODE Ⅴ의 경우는 MODE Ⅰ~Ⅳ를 만족하지 않는 경우인데, CMP값을 dv_i 로 한다.

3.3 ESC 알고리즘의 시간복잡도 계산

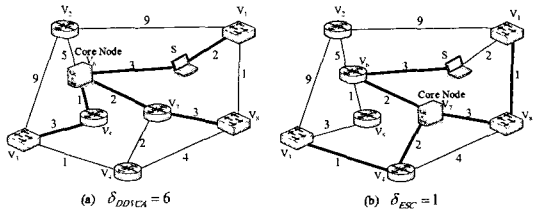
제안한 알고리즘의 시간복잡도는 다음과 같다. 3-5단계는 다익스트라(Dijkstra)의 최단 경로 알고리즘[4]을 사용하여 $O(n^2)$시간을 수행한다. 6-8단계에서 또한 다익스트라 알고리즘을 사용하여

$O(mn^2)$ 시간을 수행한다. 9-14단계는 $O(mn)$의 시간을 수행하며, 15-18단계는 $O(n)$을 수행한다. 19-21단계에서는 최악의 경우 $O(mn^2)$을 수행하는데, 세부적으로 보면 19단계에서 후보 코어노드들이 최악의 경우 $O(n)$개이며, 20-21단계에서 해당하는 목적노드를 찾기 위해 최악의 경우 $O(mn)$의 시간을 수행한다. 22-23단계에서는 $O(m)$의 시간이 걸리므로 전체적으로 19-23단계에서 위에서 말한 바와 같이 $O(mn^2)$의 시간복잡도를 보인다. 24단계는 $O(n)$의 시간이 필요하다. 결과적으로 제안한 알고리즘의 총 시간복잡도는 $O(mn^2)$이며 이는 DDVCA의 시간복잡도와 일치한다.

3.4 사례연구

간단한 예제를 사용하여 제안한 알고리즘을 살펴보자. <그림 3>의 네트워크 토폴로지에서 링크 지연시간은 각 링크 상에 표현하였다. 멀티캐스트 종단간 지연시간 제한 Δ는 10이다. <그림 3>의 (a)는 DDVCA를 통해 얻어진 멀티캐스트 트리이며, <그림 3>의 (b)는 ESC 알고리즘에 의해 생성된 트리이다.

<표 1>로부터 소스노드의 행을 제외하고 목적노드와 모든 노드들 간의 최소 지연 변이를 갖는 노드가 v_2와 v_6 그리고 v_7임을 알 수 있다. 그러나 지연시간 제한 $\Delta = 11$을 만족해야 하므로, 노드 v_2는 코어선정에서 제외된다. 따라서 DDVCA는 임의로 노드 v_6를 코어노드로 선택한다(물론 랜덤이므로 v_7일 수 있으나 [6]의 의사코드는 v_6를 선택하며 여기서는 그러한 선택을 하였다고 가정한다). 그러나 제안한 알고리즘은 노드 v_7을 코어노드로 선택한다. 이는 제안한 알고리즘이 $CMP(MODE(v_6)) = |(3+5)-2| = 6$과 $CMP(MODE(v_7)) = 4-3 = 1$ 중에서 작은 것을 <그림 1>의 19-21단계에서 계산하여, 노드 v_7을 코어노드로 선택하기 때문이다. 결과적으로, DDVCA의 멀티캐스트 지연변이는 6이지만 제안한 알고리즘의 멀티캐스트 지연변이는 1이다.



(a) $\delta_{DDVCA} = 6$ (b) $\delta_{ESC} = 1$

〈그림 3〉 (a) DDVCA와 (b) ESC 알고리즘

〈표 1〉 제안한 알고리즘에 의한 코어노드 선택

		s	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
소스 노드	s	0	2	8	7	7	4	3	5	3
목적 노드	v_1	2	0	9	6	5	6	5	4	1
	v_3	7	6	9	0	1	3	4	3	5
	v_8	3	1	10	5	4	6	5	3	0
max_i		7	6	10	6	5	6	5	4	5
min_i		2	0	9	0	1	3	4	3	0
dv_i		5	6	1	6	4	3	1	1	5

4. 성능평가

성능 평가를 위해 사용한 랜덤 네트워크 토폴로지의 생성에 대해 설명은 다음과 같다. 이 방법 [9]은 n(네트워크의 노드 수)와 P_e(어떠한 두 개의 노드사이에 링크가 있을 확률)을 파라미터로 사용한다. 랜덤 그래프가 랜덤 네트워크의 모델이라면 이러한 그래프는 연결 그래프(Connected Graph)이어야 한다. 따라서 그래프는 최소한 하나의 신장트리를 포함하고 있다. 그러한 맥락에서 우선 랜덤 신장트리를 생성한다. 우리가 알고 있는 바와 같이, $n \geq 3$인 경우가 고려된다. 3개의 노드를 가진 트리는 유일하며, 따라서 우리는 이

것을 초기의 트리로 사용한다. 그리고 n 개의 노드를 가진 신장트리로 확장한다. 확률 P_e 를 조정한 후, 네트워크 토폴로지에 기반한 그래프 생성을 위해 트리에 속하지 않은 링크를 랜덤으로 생성한다. 조정된 확률 P_e^a 를 계산해 보자. $Prob\{event\}$ 는 $event$ 의 확률을 나타낸다. e 가 두 개의 노드 사이에 가능한 링크라고 가정하면, 다음을 얻을 수 있다.

$$P_e = Prob\{e \in spanning\ tree\} + Prob\{e \notin spanning\ tree\} \cdot P_e^a$$

$$P_e = \frac{n-1}{n(n-1)/2} + (1 - \frac{n-1}{n(n-1)/2}) \cdot P_e^a$$

$$\therefore P_e^a = \frac{nP_e - 2}{n-1}$$

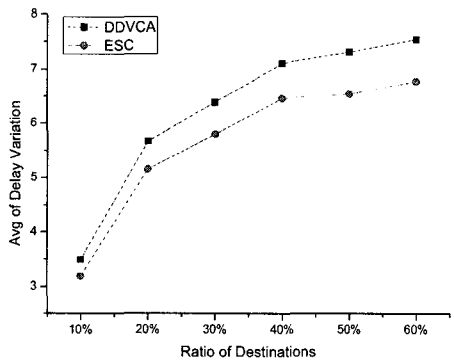
<표 3>에서는 랜덤 네트워크 토폴로지를 생성하는 의사코드를 나타내었다.

〈표 2〉 랜덤 그래프 생성 알고리즘

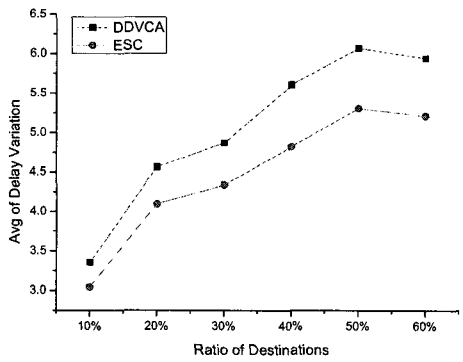
Graph Generation Algorithm

A is an incident matrix r is a simple variable *random*() is a function producing uniformly distributed random values between 0 and 1
01 **Begin** 02 $A_{1,2} = A_{2,1} = A_{2,3} = A_{3,2} = 1$ 03 **For** $i = 4$ to n **Do** 04 $r = (i-1) \times random() + 1$ 05 $A_{r,i} = A_{i,r} = 1$ 06 **For** $i = 1$ to $n-1$ **Do** 07 **For** $j = (i+1)$ to n **Do** 08 **If** $P_e > random()$ **then** $A_{i,j} = A_{j,i} = 1$ 09 **End Algorithm.**

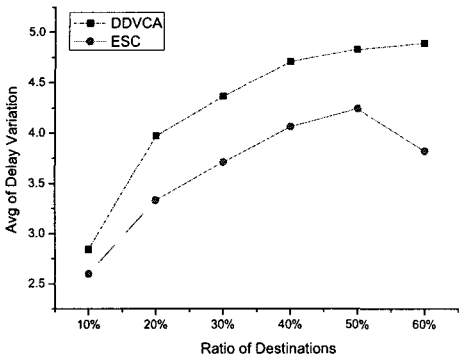
DVBMT 문제에서 최고의 해법으로 알려진 DDVCA와 본 논문에서 제안한 알고리즘을 멀티캐스트 지연변이의 관점에서 비교하였다. 제안한 알고리즘은 Visual C++로 구현하였다. 네트워크 전체 노드 수 50, 100, 200, 400에 대해 각각 10개의 서로 다른 네트워크를 생성하였다. 시작노드는 랜덤으로 선정하였으며, 목적노드들은 네트워크 토폴로지의 전체 노드 중에서 균일분포로 선택하였다. 또한, 멀티캐스트 그룹의 목적노드들은 전체 노드의 10, 20, 30, 40, 50, 그리고 60%를 차지하도록 하였으며, Δ 는 랜덤으로 선정하였다. 각각의 $|V|$ 와 P_e=0.3에 대해서 1000번(10×100=1000)의 시뮬레이션이 이루어졌다. 성능 비교를 위해, 동일한 시뮬레이션 조건상에서 DDVCA를 구현하였다. <그림 4>는 각각의 $|V|$ 에 대한 평균 지연변이($\bar{\delta}$)의 값을 나타낸다. 제안한 ESC 알고리즘의 성능 향상률을 알아보기 위해 efficiency 함수를 사용하였다.



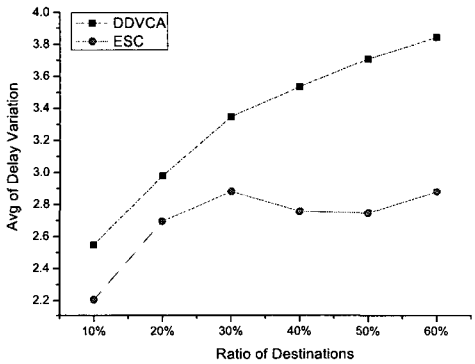
(a) |V|=50



(b) |V|=100



(c) |V|=200



(d) |V|=400

〈그림 4〉 링크간 확률(P_e =0.3)에서 각각의 $|V|$ 마다 평균 지연변이($\bar{\delta}$)의 값

$$efficiency = \frac{\delta_{DDVCA} - \delta_{ESC}}{\delta_{DDVCA}} \times 100\ (\%)$$

그 결과 ESC 알고리즘이 약 9%~29%의 성능향상을 가져왔다. 네트워크의 노드수가 적을 경우에는 코어를 선정하는 폭이 줄어들어 보다 효율적인 지연변이를 위한 멀티캐스트 경로를 찾는 경우의 수가 줄어든다. 하지만 노드수가 많은 경우에는 보다 많은 코어의 후보가 존재하기 때문에 ESC에서 보다 적합한 코어를 찾을 수 있기 때문에 효율성이 더 증가한다.

5. 결 론

본 논문은 종단간 지연시간 제한을 만족하며 멀티캐스트 지연변이 또한 최소화하여 메시지를 전송하는 방법에 대해서 연구하였다. NP-complete로 알려진 DVBMT 문제의 멀티캐스트 트리 생성을 다루는 유명한 알고리즘은 DDVCA[6]이다. DDVCA가 생성한 트리의 멀티캐스트 지연변이의 면에서 DVMA보다 조금 우수하지만 시간복잡도는 $O(mn^2)$ 으로 지금까지 알려진 가장 훌륭한 알고리즘이다. 하지만, ESC 알고리즘의 시간복잡도는 $O(mn^2)$ 으로 DDVCA의 시간복잡도와 동일하며, 더 나아가서 컴퓨터 시뮬레이션 결과는 ESC 알고리즘이 DDVCA보다 더 적은 멀티캐스트 지연변이를 갖는 트리를 생성함을 보인다.

ACKNOWLEDGMENT

본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음. IITA-2006-(C1090-0603-0046).

참 고 문 헌

[1] Y.-C. Bang and H. Choo, "On multicasting with minimum costs for the Internet topology," Springer-Verlag Lecture Notes in Computer Science, vol. 2400, pp.736-744, August 2002.

[2] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for steiner trees," Acta Informatica, vol. 15, pp. 141-145, 1981.

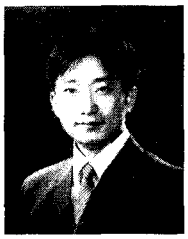
[3] H. Takahashi and A. Matsuyame, "An approximate solution for the steiner problem in graphs," Mathematica Japonica, vol. 24, no. 6, pp. 573-577, 1980.

[4] V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, "Multicast routing for multimedia communication," IEEE/ACM Trans. Networking, vol. 1, no. 3, pp. 286-292, June 1993.

[5] G. N. Rouskas and I. Baldine, "Multicast routing with end-to-end delay and delay

variation constraints," IEEE JSAC, vol. 15, no. 3, pp. 346-356, April 1997.
[6] P.-R. Sheu and S.-T. Chen, "A fast and efficient heuristic algorithm for the delay- and delay variation-bound multicast tree problem," Elsevier Computer Communications, vol. 25, pp. 825-833, 2002.
[7] A. Ballardie, B. Cain, and Z. Zhang, "Core Based Trees (CBT Version 3) Multicast Routing," Internet draft, 1998.
[8] M. Kim, Y.-C. Bang, H.-J. Lim, and H. Choo, "On Efficient Core Selection for Reducing Multicast Delay Variation under Delay Constraints," IEICE Transactions on Communications, vol. E89-B, no. 9, pp. 2385-2393, September 2006.
[9] A. S. Rodionov and H. Choo, "On Generating Random Network Structures: Connected Graphs," Springer-Verlag Lecture Notes in Computer Science, vol. 3090, pp. 483-491, September 2004.

◐ 저 자 소 개 ◑



안 영 진(Youn-gjin Ahn)

2004년 성균관대학교 정보통신공학부 졸업(학사)
2006년 성균관대학교 컴퓨터공학과 졸업(석사)
2005~2007 정보통신부ITRC 지능형HCI융합연구센터 연구원
2007~현재 LG전자 정보통신사업본부 단말연구소 연구원
관심분야 : 네트워크 이동성, 센서 네트워크, 라우팅 프로토콜
E-mail : yjahn@ece.skku.ac.kr

김 문 성(Moon-seong Kim)

1999년 건양대학교 수학과 졸업(학사)
2001년 성균관대학교 수학과 졸업(석사)
2006년 성균관대학교 전기전자 및 컴퓨터공학과 졸업(박사)
2005년 한국전자통신연구원(ETRI) 위촉연구원
2005~2006 정보통신부ITRC 지능형HCI융합연구센터 연구원
2006~현재 성균관대학교 정보통신공학부 연구교수
관심분야 : 라우팅 프로토콜, 모바일컴퓨팅, 정보보호, 성능평가, 수치해석
E-mail : moonseong@ece.skku.ac.kr

추 현 승(Hyun-seung Choo)

1988년 성균관대학교 수학과 졸업(학사)
1990년 University of Texas 컴퓨터공학과 졸업(석사)
1996년 University of Texas 컴퓨터공학과 졸업(박사)
1997년 특허청 심사4국 컴퓨터심사담당관실(사무관)
1998년~현재 성균관대학교 정보통신공학부 부교수
2001년~현재 한국인터넷정보학회/한국시뮬레이션학회 이사
2004년~2006 대통령직속 교육혁신위원회 전문위원
2004년~현재 한국인터넷정보학회 논문지편집위원장
2005년~현재 건강보험심사평가원 전문위원
2005년~현재 한국정보과학회 논문지편집위원
2005년~현재 정보통신부ITRC 지능형HCI융합연구센터장
관심분야 : 유/무선/광네트워킹, 모바일컴퓨팅, 임베디드SW, 그리드컴퓨팅
E-mail : choo@ece.skku.ac.kr