k-ary n-cube네트웍에서 결함허용실시간통신을 위한 우회경로 설정

이 경 희, 최 문 옥, 이 충 세
충북대학교 컴퓨터과학과

Establishing detours for Fault-Tolerance Real-Time Communication in k-ary n-cube Networks

Kyung Hee Lee, Moon Ok Choi, and Chung-Sei Rhee
Dept. of Computer Science, Chungbuk Nat'l Univ.

요 약

실시간 어플리케이션이 확장되고 복잡해질수록 시스템이나 네트웍에 존재하는 결함에 대응해야 할 필요성은 더 높아진다 이런 작업의 활용도가 높지는 않더라도 하나의 결함이 시스템 전체에 영향을 미칠 가능성은 항상 있기 때문에 신뢰도 제공면에서 요구되는 작업이다. 전통적인 결함허용 방법은 여분의 하드웨어나 소프트웨어를 중복사용함으로서 결함에 대처하고자 하였다. 본 논문에서는 k-ary n-cube네트워크에 대하여 네트웍을 구성하는 요소를 중복하는 것이 아니라 네트웍의 결함발생시 통신경로를 우회함으로써 결함을 허용하는 방법을 제안한다.

1. 서 론

많은 어플리케이션들이 시간제약을 보장함과 동시에 효율적인 비용으로 결함을 극복할 수 있는 통신 서비스를 요구한다[4] 예를 들면, 오디오나 비디오같은 연속적인 디지털 멀티미디어 데이터 통신이 요구되는 화상회의나 분산실시간 제어시스템 등이 그 예이다 전통적인 데이터그램 서비스와는 달리 Quality of Service(QoS)를 보장하는 실시간 통신서비스의 필요성은 더욱 높아지고 있다. 많은 어플리케이션에서 실시간 서비스를 제공하기 위하여 채택하고 있는 방법은 실시간 채널을 이용한 접근법이다. 임의의 노드가 패킷을 전송하려고 할 때 네트웍에 패킷 데드라인내에 전송이 가능한 경로를 설정하여 각 링크마다 최소의 데드라인을 할당한 후, 그 경로를 이용하여 모든 패킷을 전송하는 방법이 실시간 채널을 이용한 방법이다. 이러한 경우 설정된 경로에 결함이 발생되더라도 요구한 지연시간 내에 패킷이 전달가능하도록 하는 방법에 대한 연구가 요구된다.

본 논문에서는 실시간 통신에서의 결함허용에 대한 기존의 연구를 살펴보고 하이퍼큐브 네트웍의 주채널에 결함이 발생했을 경우 우회할 보조채널설정 알고리즘을 제안한다. 2장에서는 실시간 채널에 대한 관련 연구를 기술하고, 3장은 관련연구로써 메쉬네트웍에서 결함허용 방법을 설명한다. 4장은 하이퍼큐브네트웍에 기반한 우회경로설정 알고리즘을 제안하고, 5장에서 결론을 맺는다.

2. 실시간채널

실시간 채널이란 송신노드(source node)에서부터 수신노드(destination node)까지 패킷이 사용자가 지정한 지연시간(delay bound) 내에 전송되는 것을 보장하는 가상채널을 말한다. 일단 채널이 한번 설정되면 실시간 채널의 모든 패킷들은 같은 경로로 전송된다[12] 채널 설정을 요구하는 각 노드는 자신의 패킷발생유형(traffic pattern)을 채널에 포함된 링크들에 알려주고, 설정된 채널이 모든 패킷을 데드라인내에 전송가능한지를 체크하여야 한다. 패킷발생유형은 (σ,ρ)-model[7], (x_{min}, x_{ave}, I, s)-model[8]로 표현가능하다. 네트워크를 구성하는 링크의 전송량에는 한계가 있으며, 패킷마다 시간제약이 있으므로 데드라인 내에 전달된다는 보증을 할 수 없기 때문에 이러한 체크가 요구된다. 이러한 패킷발생유형을 표현하기 위하여 링크에 패킷이 도달되는 가장 짧은 주기(T), 링크가 패킷을 완전히 전송하는데 걸리는 가장 긴 시간(C), 채널의 지연시간(d)을 파라메터로 이용한다.

어떤 노드가 새로운 실시간 채널 설정요구를 받게 되면, 다음 두 가지 사항을 고려한다[2].

1) 스케쥴링 문제

n개의 채널 집합 $\tau_i = (T_i, C_i, d_i)$, $i=1,2,\ldots,n$,이 하나의 링크를 통과한다고 할 때 집합에 속한 모든 채널이 스케쥴링이 가능한지 판단하는 문제

2) 최소 데드라인 할당문제

하나의 링크상에서 n-1개의 채널이 스케쥴링이 가능하다고 가정하고 새로운 채널 τ_n이 최소 패킷도달 시간 T_n과 최대전달시간 C_n의 조건에서 설정가능하려면 링크의 최소지연시간 d_n을 계산하여 할당하는 문제

위 두 가지 문제를 해결하는 방법으로 다음의 정리를 이용한다[1].

[정리 1] n개의 채널 집합 $\tau_i = (T_i, C_i, d_i)$, $i=1,2,\ldots,n$이 하나의 링크를 통과한다고 할 때 그 집합에 속한 모든 채널이 스케쥴링이 가능한가의 문제는 다음의 2가지 조건을 만족하면 된다.

1) $\sum_{i=1}^{n} C_i/T_i \le 1$

2) $\forall t \in S,\ \sum_{i=1}^{n} \lceil (t-d_i)/T_i \rceil^{+} C_i \le t$, *where* $S = \bigcup_{i=1}^{n} S_i$,

$S_i = \{d_i + nT_i; n = 0,1,\ldots, \lfloor (t_{\max} - d_i)/T_i \rfloor\}$,

$t_{\max} = \max\left\{d_1, \ldots, d_n, \left[\sum_{i=1}^{n}(1-d_i/T_i)C_i\right]/(1-\sum_{i=1}^{n} C_i/T_i)\right\}$,

and $\lceil x \rceil^{+} = n$ if $n-1 \le x \le n$, $n = 1,2,\ldots$,

and $\lceil x \rceil^{+} = 0$ for $x < 0$

[정리 2] 패킷전송에 걸리는 시간을 표현하는 함수 $f(t,d_n) = \sum_{i=1}^{n} \lceil (t-d_i)/T_i \rceil^{+} C_i$와 S가 정리1을 만족할 때, $\forall t \in S, f(t,C_n) \le t$ 이면 최소 지연시간(d_n)은 Cn 이다. 그렇지 않으면 최소지연시간은 $d_n = \max\{d^t \cdot t \in G\}$이다.

단, $G = S \cap \{t : f(t,C_n) > t$ 그리고 $d^t = C_n + k_f^t T_n + \varepsilon_f^t + \varepsilon_f^t$로 계산된다. 여기서, $k_f^t = \lceil f(t,C_n) - t/C_n \rceil - 1$, $\varepsilon_f^t = f(t,C_n) - t - k_f^t T_n$, $k_f^t = \lfloor (t-C_n)/T_n \rfloor$ 이다.

실시간서비스를 제공하는 네트웍의 임의의 노드에서 채널설정요구가 있을 때, 위의 두 가지 정리를 이용하여 알고리즘 1의 절차로 채널이 설정된다[2].

알고리즘 1 : 실시간 채널설정 방법

1단계. 정리2를 사용하여 링크 $l_j, j = 1,\ldots,k$ 상의 최소패킷 데드라인 $d^t_{\min,j}$을 계산하여 각 링크에 할당한다.

2단계 $\sum_{j=0}^{k} d^t_{\min,j} \le D_i$이면, 요구한 채널을 설정한다. 링크 l_j의 데드라인을 $d_j^t = d^t_{\min,j} + \delta_i/k$로 할당한다. 단, $\delta_i = D_i - \sum_{j=1}^{k} d^t_{\min,i}$이다 만약 위 조건을 만족하지 않을 경우엔 채널을 설정하지 않는다.

실시간 채널의 모든 패킷은 같은 경로를 통하여 전송되기 때문에 그 경로 상에 있는 하나의 컴포넌트라도 결함이 발생되면 전체적인 채널이 사용할 수 없도록 된다[1] 그러므로 실시간 채널의 신뢰도를 높이기 위한 대표적인 방법 몇 가지를 살펴보면, 첫째 여분의 링크와 노드가 충분한 채널로 확장을 하여, 패킷들을 고장난 노드나 링크를 우회하여 전송하는 방법[3]이다. 둘째, 같은 패킷을 동시에 여러 개의 서로 다른 경로를 통하여 전송하고 전송된 결과를 비교하여 결함이 있는 패킷을 제거하는 방법이 있다.

본 논문에서는 첫 번째 결함허용방법을 이용한다. 실시간 채널을 설정할 때 주채널과 보조 채널을 동시에 설정하여 주채널에 결함이 발생할 경우 보조 채널로 패킷을 우회하여 전송하는 방법이다. 이 방법은 보조채널이 주채널과 같이 최단경로로 설정됨과 동시에 주채널과는 서로 다른(disjoint) 경로로 구성되어야만 결함을 극복하는 데 유리하다. 그러나 이 방법의 단점은 보조 채널을 설정할 때에도 주채널과 같은 양의 자원을 예약하게 됨으로써 자원의 낭비를 초래하게 되며, 사용되지 않는 보조 채널로 인하여 새로운 주채널의 설정까지 방해하는 경우가 발생될 수 있다. 이를 개선한 방법으로는 보조 채널을 설정할 때에는 주채널보다 더 적은 비율로 자원을 예약함으로써 채널을 multiplexing하는 방법[4]과 더 이상 불필요한 보조채널은 제거함으로써 다른 채널의 설정에 방해가 되는 것을 막는 방법이 있다[2].

3. 메쉬네트웍에서 우회경로 설정방법

메쉬네트웍에서는 경로설정도 비교적 용이하고 채널상에 결함이 발생할 경우 설정이 가능한 우회경로도 계산이 간단하다. 예를 들어 m×n 메쉬네트웍에서 0번노드에서 마지막 (mn-1)번까지 가능한 경로의 수는 (m×(n-1))이다. 설정가능한 채널의 수가 적기 때문에 허용할 수 있는 결함의 수도 작다. 따라서 하나의 결함을 허용할 수 있는(single failure immune) 채널을 설정하는데 필요한 여분의 링크와 노드를 탐색하는 휴리스틱한 방법이 제시되었다[2]. 그림1을 보면 송신노드 1에서 수신노드 1까지의 주채널이 설정되어 있는데, 이 주채널을 구성하는 노드나 링크에 결함이 발생될 경우 우회할 수 있는 링크와 노드를 표현한 예이다

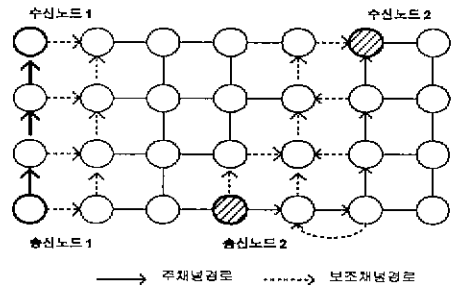


그림 1 메쉬네트웍에서 우회경로설정의 예

4. k-ary n-cube네트웍에서 우회경로 설정방법

메쉬네트웍에서는 그림1에서도 볼 수 있듯이 설정가능한 보조채널에 제한이 많으며 1개 이상의 결함을 극복할 수 없는 경우가 많다. 그러나 하이퍼큐브 구조로 네트웍을 설계하고, 최단경로 설정이 불가능할 경우엔 비최단경로로(hamming distance 이상의 길이를 갖는) 라우팅 하도록 하여 n차원 하이퍼큐브에서 n-1개의 노드 혹은 링크의 결함을 허용하는 알고리즘이 제시된 바 있다[4]. 다만 이 방법은 실시간을 고려한 것은 아니다

k-ary n-cube네트웍에서도 실시간 채널을 설정하기 위해서 앞에서 살펴본 정리를 이용하여 알고리즘 1과 같은 방법으로 채널을 설정한다. 예를 들면, 그림 2(a)의 노드000이 송신노드로 노드 222에 패킷을 전송할 실시간 채널설정을 요구하면, 채널이 설정될 링크마다 지연시간을 계산하여 할당하고, 각 링크의 지연시간을 모두 합한 것이 패킷의 데드라인보다 작으면 데드라인내에 전송이 가능하므로 요청한 채널을 설정한다. 결함 발생했을 때 k-ary n-cube네트웍에서는 메쉬네트웍에서 보다 많은 우회경로를 가질 수 있다. 그림 3에서 알 수 있듯이 송신노드가 000이라고 할 때 연결되어있는 링크가 3개이므로 최대 2개까지 결함은 허용가능하다. 일반적인 경우, 노드 000처럼 가장자리노드가 아닌 경우엔 6개의 링크에 연결되어 있으므로 패킷의 데드라인에 여유만 있다면 5개의 결함은 허용될 수 있다 물론 하이퍼큐브 네트웍에서도 각 링크마다 대역폭에 한계가 있으므로 채널을 설정할 때 적절한 대책이 별도로 요구된다. 예를 들어 앞서 제시한 것처럼 채널을 multiplexing하는 방법[4]이나 사용되지 않는 채널을 일정시간마다 제거하는 방법[2], 또는 본 논문에서 제시하는 바와 같이 중복되는 채널의 설정을 줄이기 위하여 주채널과 완전히 다른 경로를 이용하는 채널 3개만

보조 채널로 설정하는 방법이 가능하다 보조채널의 개수를 3개로 한정한 것은 3차원 큐브의 경우에 주채널과 겹치지않는 3개의 경로가 설정가능하며, 실시간을 제공하지 않는 경우에 하이퍼큐브구조에서 2개의 보조채널만 사용하여도 채널선점기능을 도입하면 n-1개의 결함이 허용가능 하다[6].

우회경로 알고리즘을 살펴보기에 앞서 다음과 같은 가정을 한다

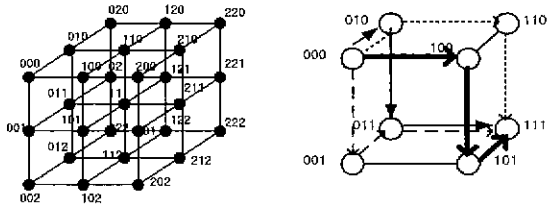
1) 노드의 번호가 1bit만 다른 노드와 노드 사이에 링크가 존재한다.

그림 2(b)의 노드 000을 보면 링크에 연결된 노드가 001, 010, 100으로 모두 한 비트만 다른 노드들이다. 이들 노드를 노드000의 이웃노드(neighbor)라 한다.

2) 보조채널을 설정할 때 주채널과 완전히 다른 경로를 사용하는 채널만 설정한다. 이때 비최단경로도 허용한다.

그림2(b)는 그림2(a)의 서브네트웍을 표현한 것으로 주채널C_p가 지나는 경로와 공통인 링크를 갖는 채널들은 설정하지 않고 완전히 다른 경로를 갖는 채널만 설정한다. C_p = (000, 100, 101, 111), C_{b1} = (000, 010, 011, 111), C_{b2} = (000, 010, 110, 111), C_{b3} = (000, 001, 011, 111)이 된다.

단, C_p는 주채널의 경로, C_b는 보조채널의 경로를 의미하며, 채널의 경로는 거치는 노드 번호를 이용하여 나타낸다. 즉, C=(N_s, N_1, N_2,...,N_d)로 표현한다. 여기서 N_s는 송신노드, N_d는 수신노드, N_n은 채널상의 노드들을 의미한다.



(a) (b)

그림 2. 3-ary 3-cube 네트워크

3) 다음에 제시된 보조채널설정 알고리즘은 k-ary n-cube를 기반으로 한다. k-ary n-cube는 그림2(a)에서 알 수 있듯이 각 노드번호를 k진법을 이용하여 표현하며, k-ary n-cube는 전체 네트웍을 부분네트웍으로 분할하거나 더 큰 네트웍으로 확장하는데 편리한 구조이다.

알고리즘2는 실시간 채널을 설정할 때 주채널과 함께 설정되는 보조채널 설정 방법을 기술한 것이다.

알고리즘 2 : 보조채널설정 알고리즘

```
1. Set up primary channel Cp;
2. if s>d then flag = 0 else flag =1;
2. While( number of backup channel = 3){
3. i = 1;
4' u = neighbor node of source node;
5' switch(flag){
6:    case 0:{
7'          for (j = 0; number of Nd, j+k )
8'            if vj is adjacent to u && vj is not in Cp
               && j > number of node u
9:              then add to Cbi;
10:   }
11:   case 1:{
12:         for (j = 0; number of Nd; j-k )
13:           if vj is adjacent to u && vj is not in Cp
               && j < number of node u
14:           then add to Cbi;
15:    }}}
16 end;
```

k : 큐브구조를 나타내는 array변수

v_i : i번 노드

위와 같은 방법으로 채널을 설정하면 주채널과 링크나 노드가 전혀 겹치지 않는 채널이 설정된다. 그러므로 어떤 보조채널을 새로운 주채널로 선택하더라도 결함있는 노드나 링크를 우회하게 되므로 결함을 극복할 수 있다.

5. 결 론

실시간 서비스를 제공하는 네트웍에서 결함을 허용하는 문제는 시간제약하에서 네트웍의 신뢰도를 보증하여야 하므로 많은 어려움이 따른다. 본 논문에서 제안한 알고리즘은 우회경로설정 방법이다. 기존에도 우회경로설정 알고리즘은 제안된 바 있으나 결함이 있는 노드나 링크를 우회하는 보조채널을 설정하는 방법이었다. 그러나 이 방법은 보조채널이 설정되는 경우의 수가 증가되어 사용되지 않는 채널까지 설정되도록 허용함으로써 효율면에서 좋지않은 결과를 보였다. 앞으로 하이퍼큐브의 특성을 이용하여 보조채널설정에 소요되는 시간복잡도를 개선하고 결함이 발생된 주채널을 새로운 채널로 교환하는 지연시간(channel switching delay)도 줄이는 방법이 연구되어야 한다.

참 고 문 헌

[1] Q Zheng and K G Shin, "On the Ability of Establishing Real-Time Channels in Point-To-Point Connected Packet Switching Networks," *IEEE Trans. Comm.*, vol. 42, nos. 2/3/4, pp. 1096-1105, Feb./Apr 1992.

[2] Q. Zheng and K.G. Shin, "Fault-Tolerant Real-Time Communication in Distributed Computing Systems," *IEEE Trans. Parrallel and Distributed sys.* vol 9. no. 5 pp.470-480, May 1998.

[3] B. Chen, S. Kamat, and W. Zaho, "Fault-Tolerant Real-Time Communication in FDDI-Based Networks," *Proc. Real-Time Systems Symp.*, 1995

[4] Seungjea Han, K. G. Shin, "A Primary-Backup Channel Approach to Dependable Real-Time Communication in Multi-hop Networks," *IEEE Trans. on Computers vol. 47, no 1*, pp 46-61, 1998

[5] M. S Chen and K.G Shin, "Adaptive Fault-tolerant Routing in Hypercube Multicomputers," *IEEE. Trans. Computers*, vol. 39, no. 12, pp.1406-1416, Dec 1990

[6] 심재철, 김동승, "하이퍼큐브 컴퓨터에서의 결함허용 경로배정기법," *한국정보과학회 논문지*, 제23권, 5호, pp. 509-518, 1996년 5월.

[7] R.L Cruz,"A Calculus for Network Delay, Part I' Netwrok Elements in isolation," *IEEE Trans. on Information Theory*, vol. 37, no. 1, pp.132-141, Jan. 1991.

[8] S.J. Golestan, "A Framing Strategy for Congestion Management," *IEEE Jornal on Selected Areas In Communications*, vol 9, no 7, pp 1064-1077, Sep. 1991