1999년도 대한전기학회 하계학술대회 논문집 1999. 7.19-21

유전 알고리즘을 이용한 고장포용 라우팅 알고리즘 설계

문 대 근, 김 학 배
연세대학교 전기 · 컴퓨터공학과

A Fault-Tolerant Routing Algorithm Using a Genetic Algorithm

Daekeun Moon and Hagbae Kim
Dep. of Electrical & Computer Engineering, Yonsei University

Abstract - 고신뢰도의 요구를 보장하는 병렬 구조의 분산시스템의 사용이 증가함에 따라 네트워크 상에서 메시지전달을 방해하는 요소고장의 영향을 최소화시킬 수 있는 고장포용 라우팅에 대한 중요성이 부각되고 있다. 그러나, 네트워크의 복잡한 환경 때문에 요소고장을 극복하기 위한 고장포용 라우팅 알고리즘의 설계는 쉬운 일이 아니다. 본 논문에서는 2차원 메쉬 네트워크에 적용되는 최적의 고장포용 라우팅 알고리즘을 설계하기 위하여 관련 응용분야에서 그 유용성이 검증된 유전 알고리즘을 이용한다. 제안된 알고리즘은 wormhole 라우팅 방식을 사용하며, 교착상태를 없애기 위하여 하나의 물리적 채널을 공유하는 4개의 가상채널을 사용한다. 마지막으로, 시뮬레이션을 통하여 제안된 알고리즘이 기존의 다른 고장포용 라우팅 알고리즘보다 우수함을 증명한다.

1. 서 론

근래들어 널리 활용되고 있는 병렬구조의 분산메모리 멀티컴퓨터는 프로세서, 지역메모리, 라우터 등을 요소로 갖는 많은 노드들로 구성되어 있다. 각 노드의 컴퓨터들은 각각의 할당된 프로그램을 수행하는 동시에 네트워크 내에 분산되어 있는 노드와 노드컴퓨터 사이의 데이터를 수시로 상호 교환한다. 따라서 라우팅 전략은 네트워크의 성능 및 신뢰도를 좌우하는 가장 중요한 요소 중의 하나이다.

일반적으로 고장포용 라우팅이란 메시지들이 요소고장에 때문에 그들의 목적지에 도달할 수 없을 경우에 요소고장들을 피해서 이들을 전달시키는 라우팅기법으로 정의된다. 그러므로, 신뢰도가 중요시되는 응용분야에 활용되는 멀티컴퓨터에서는 고신뢰도 확보를 위해 고장포용 라우팅이 중요한 의미를 가진다. 최근까지 이러한 관점에서의 다양한 기법들을 사용한 고장포용 라우팅 알고리즘들이 제안되었다[1,2,3]. 본 논문에서는 기존의 방법들과는 다르게 고장포용 라우팅 알고리즘을 유도하기 위해서 최적화 알고리즘 중에 하나인 유전 알고리즘을 이용함으로써 보다 큰 확장성을 가질 수 있도록 한다.

본 논문에서는 자신 주변의 노드상태(정상 또는 고장)만을 감지할 수 있는 노드들로 구성된 wormhole 라우팅 기법[4]이 적용된 2차원 메쉬 네트워크를 기본 토폴로지로 설정한다. 또한, 교착상태(deadlock)를 방지하기 위하여 한 개의 물리적 채널을 공유하는 4개의 가상채널[5]을 사용한다. 이러한 조건하에서 본 논문은 유전 알고리즘을 이용한 최적의 고장포용 라우팅 알고리즘을 제안한다. 이를 위해 모든 패킷들은 자신이 바로 이전에 머물렀던 노드로는 다시 전달되지 않도록 하는 기본적인 염색체 구조를 설정하고 유전 알고리즘이 고장포용 라우팅 알고리즘을 구성하는 라우팅표를 진화시키는 동시에 성능도 향상시키도록 한다.

2. 유전 알고리즘

자연계에서 생물의 진화과정을 살펴보면 어떤 세대를 형성하는 개체들의 집합 즉, 개체군 중에서 환경에 대한 적합도가 높은 개체가 살아남아 재생산할 수 있는 확률이 높으며, 이 때 선택, 교배, 돌연변이로서 다음세대의 개체군을 형성하게 된다. 유전 알고리즘은 이와 같이 생물의 진화과정을 인공적으로 모델링한 알고리즘이다. 인공 유전 시스템인 유전 알고리즘은 자연세계의 진화 현상에 기초한 계산 모델로서 John Holland[6]에 의해 1975년에 개발된 전역적인 최적화 알고리즘이다. 이는 모든 생물은 주어진 다양한 환경 속에 적응함으로써 살아남는다는 Darwin의 적자생존의 이론을 기본개념으로 한다. 유전 알고리즘은 풀고자 하는 문제에 대한 가능한 해들을 정해진 형태의 자료 구조로 표현한 다음 이들을 점차적으로 변형함으로써 점점 더 좋은 해들을 만들어 낸다. 자연계의 모든 생물들은 유전인자인 염색체에 의해서 생식과 유전 등 진화과정을 반복하여 발전해 간다. 유전 알고리즘에서는 생물학적 유전인자인 염색체에 해당하는 문자열을 가지고 생물과 같은 선택, 교배, 돌연변이를 거쳐서 다음 세대의 새로운 자손을 인공적으로 만들어 낸다. 자연의 생물유전을 모방한 연산자들을 반복적으로 적용하여 적합한 해를 탐색한다.

3. 고장포용 라우팅 알고리즘

Wormhole 라우팅 방식은 교착상태에 민감하기 때문에 교착상태를 없애기 위한 부가적인 기법을 요구한다. 교착상태를 해결하기 위해서 본 논문에서는 4개의 가상채널(vc_1, vc_2, vc_3, vc_4)을 사용한다. 발생된 메시지가 사용하게 될 가상채널은 그 메시지의 출발노드와 목적노드의 위치관계에 따라 결정되며, 한 번 결정된 가상채널은 그 메시지가 목적지에 도달할 때까지 변하지 않는다. 표 1은 가상채널의 사용을 보여준다. 이와 같은 가상채널의 설정은 메시지끼리의 충돌을 최소화할 수 있으므로 교착상태를 효과적으로 막을 수 있다.

표 1. 가상채널의 사용

출발노드 x_1, y_1에서 목적노드로 전달되는 메시지	
두 노드사이의 위치관계	사용되는 가상채널
$x_2 \geq x_1$ and $y_2 > y_1$	가상채널 1(vc_1)
$x_2 \geq x_1$ and $y_2 \leq y_1$	가상채널 2(vc_2)
$x_2 < x_1$ and $y_2 > y_1$	가상채널 3(vc_3)
$x_2 < x_1$ and $y_2 \leq y_1$	가상채널 4(vc_4)

메쉬 구조에서 발생된 임의적인 요소고장의 배열형태는 라우팅전략에 심각한 영향을 주며, 경우에 따라서는 메시지 전달의 실패를 유도할 수도 있다. 이러한 단점을 최소화하기 위해서 본 논문에서는 요소고장 중에 노드고장만을 고려하며, [2]에서 제안된 것과 유사한 비활성화 규칙을 사용한다. 즉, 임의의 모양으로 배열된 고장지역을 블록형태로 바꾼다. 그러면, 모든 노드는 자신 주변에 최대 하나의 고장노드만을 갖게 된다.

라우팅 알고리즘을 구성하는 라우팅표는 노드에 도달한 메시지의 출력방향을 결정하는 역할을 한다. 본 논문에서는 2차원 메쉬를 구성하는 모든 노드들이 같은 라우팅표를 사용한다고 가정한다. 메시지가 전달된 노드는 그 메시지가 이전에 머물렀던 노드와 최종적으로 전달되어야 할 노드에 대한 정보를 가지고 있고 이 정보를 바탕으로 라우팅표에서 메시지의 출력방향을 결정한다. 임의의 노드에 도달한 메시지가 이전에 머물렀던 노드의 방향은 4방향(Up, Down, Right, Left)으로 나누어지고, 그 메시지의 목적노드 방향은 8방향(UL, UR, RU, RD, DR, DL, LD, LU)으로 나누어진다. 예를 들어, 그림 1에서 목적노드 1과 목적노드 2는 메시지가 현재 머물고 있는 노드를 기준으로 각각 UR과 RD에 위치해 있다. 그리고 메시지가 이전에 머물렀던 노드의 방향은 Left이다. 따라서, 목적노드 1로 전달될 메시지는 라우팅표의 (UR, Left)부분의 값을 읽어들여 그 방향으로 메시지를 전달하고, 목적노드 2로 전달될 메시지는 라우팅표의 (RD, Left)부분의 값을 읽어들여 그 방향으로 메시지를 전달한다.

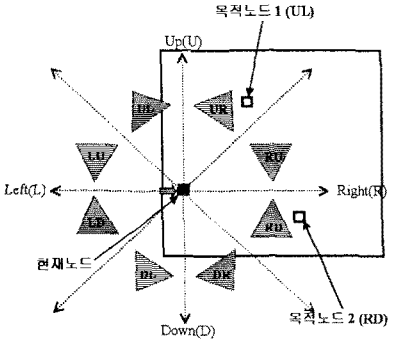

그림 1. Left에서 전달된 메시지의 목적노드 방향

메쉬 구조는 규칙적인 구조가 아니기 때문에 한 노드에 연결된 물리적인 채널의 수가 노드의 위치에 따라 다르다. 본 논문에서는 이러한 사항을 고려하여 모든 노드에 같은 라우팅표를 적용함으로써 발생할 수 있는 메시지 전달의 실패를 막기 위한 경계노드에서의 예외처리를 실행한다. 경계노드에서 라우팅표로부터 읽어온 값, 즉 메시지의 출력방향이 메쉬를 벗어나면 그 방향과 반대방향이 출력방향으로 설정되고 설정된 방향으로 메시지가 전달된다. 그러나 설정된 방향의 노드가 고장노드이면 메시지가 이전에 머물렀던 노드의 방향과 반대방향으로 출력방향이 다시 설정되고 그 방향으로 메시지가 전달된다. 그림 2는 이러한 예외처리를 보여준다.

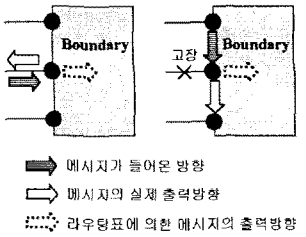

그림 2. 경계노드에서의 예외처리

유전 알고리즘의 사용을 위해서는 먼저 라우팅 알고리즘을 구성하는 라우팅표의 파라미터들을 이진 스트링 형태의 염색체로 표현해야 한다. 본 논문에서 사용되는 라우팅표는 노드주위의 고장배열에 따라서 두개의 부분표로 나누어진다. 첫 번째 부분표는 노드주위에 고장노드가 없을 때 사용되는 것으로 5×8=40(세포)의 크기를 가진다. 이 계산에서 앞의 숫자는 이전에 메시지가 머물렀던 노드의 방향, 뒤의 숫자는 목적노드의 방향과 관련된 숫자이다. 두 번째 부분표는 노드주위에 한 개의 고장노드가 존재할 때 사용되며 이 부분표의 크기는 4×8=32(세포)이다. 임의의 노드에 도달한 메시지의 출력방향을 표현하는 세포는 네 가지 상태(up, down, right, left)중에 하나를 나타내기 때문에 한 세포는 2비트로 표현된다. 따라서 본 논문에서 사용되는 염색체의 크기는 (40×2)+(32×2)=144(비트)이다. 최종적으로, 염색체로 표현된 라우팅표는 그림 3에 나타나 있다.

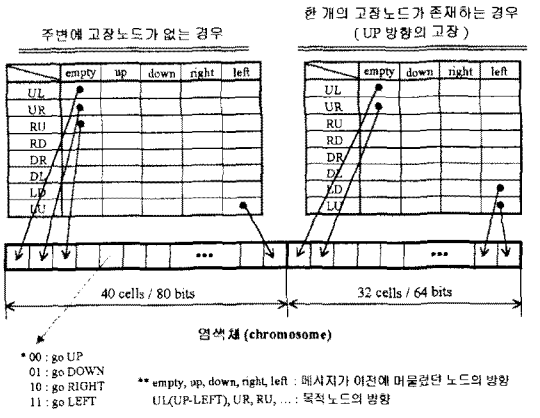

그림 3. 라우팅표의 염색체 표현

라우팅표는 노드의 주변 고장환경에 대한 모든 정보를 포함하지는 않는다. 단지 주변 고장노드의 수에 따른 특별한 경우만을 포함한다. 실제로 라우팅표를 적용할 때 주변 고장노드의 수에 관련된 부분표를 그림 4와 같이 변환하여 사용하면 모든 경우를 해결할 수 있기 때문에 이와 같은 라우팅표의 설정은 합리적이다.

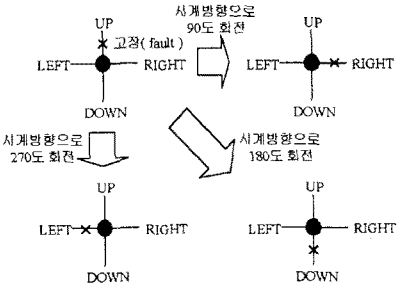

그림 4. 회전에 의한 라우팅표의 변환

유전 알고리즘의 주 연산자인 교배는 다음과 같은 세 단계로 이루어진다. 먼저 부모세대로부터 두 개의 염색체(라우팅표)가 선택된다. 두 번째로 두 염색체의 교배가 일어나는 부분이 염색체의 MSB로부터 임의적으로 선택된다. 마지막으로 교배점 이후의 모든 비트들을 서로 교환한다. 유전 알고리즘의 다른 연산자인 돌연변이는 스트링의 임의의 부분의 값을 바꾸는 것으로, 최적에 가까운 결과를 찾는 유전 알고리즘의 능력을 향상시킨다.

선택 메커니즘의 평가 과정에서는 모든 부모들의 성능이 계산되고 이 값이 적합도로 저장된다. 만약 한 부모가 다른 부모들 보다 더 좋은 적합도를 갖는다면 이 부모가 자손을 발생시킬 확률이 높다. 본 논문에서는 라우팅 알고리즘을 구성하는 라우팅표의 성능을 나타내는 요소로 지연시간을 사용하였으며, 유전 알고리즘의 반복시행때마다 메시지의 출발노드와 목적노드, 메시지의 출발

시간, 고장노드의 수와 위치를 임의적으로 발생시켰다. 따라서 부모세대 라우팅표의 성능은 메시지들의 지연시간의 합으로 계산되어지고, 이 값이 라우팅표의 적합도로 부모에 저장된다. 지금까지 설명한 두 연산자와 선택 메커니즘을 적용한 유전 알고리즘의 학습단계는 다음과 같다.

1) P개의 개체를 임의적으로 초기화한다.
2) 선택 메커니즘에 의해서 P개의 개체에 대한 적합도를 구하고 이 적합도를 기준으로 모든 개체들을 순차적으로 정렬한다.
3) 적합도가 가장 우수한 상위 25% 즉, P/4개의 개체를 선택하여 다음 세대로 수정 없이 전달하고 나머지 개체들은 제거한다. 선택된 개체들은 다음세대의 부모로 선정되며 이들의 교배와 돌연변이 연산을 통하여 나머지 3P/4개의 개체를 생산한다.
4) (3)단계에 의해서 새로운 세대가 만들어지면 (2)단계를 재수행한다. 이러한 과정은 만족할만한 해를 얻을 때까지 반복된다.

4. 시뮬레이션에 의한 성능평가

본 절에서는 유전 알고리즘을 이용한 고장포용 라우팅 알고리즘(f-GA)과 [1]에서 제안된 f-cube4와의 성능을 비교한다. 두 알고리즘의 성능비교는 10×10 메쉬 네트워크에 3%의 노드가 고장인 경우, 5%의 노드가 고장인 경우에 대한 시뮬레이션을 통하여 이루어진다. 여기에서 네트워크상의 각 노드는 프로세서, 라우터, 가상채널에 연결된 버퍼로 구성되고, 각 메시지는 20개의 플릿으로 나누어진다. 또한, 각 가상채널에 연결되어 있는 queue는 한 개의 플릿만을 저장할 수 있다. 네트워크에 적용되는 부하(load)와 노드고장의 수에 따른 각 시뮬레이션은 20,000클럭동안 실행된다. 각 라우팅 알고리즘의 성능은 정규화된(normalized) 네트워크 처리량과 메시지 지연시간을 통하여 측정되었다. 정규화된 네트워크 처리량은 부하가 최대일 때 전달될 수 있는 메시지 수에 대한 실제 전달된 메시지 수의 비율로 정의되고, 메시지 지연시간은 한 메시지가 자신의 출발노드로부터 목적노드까지 도달하는데 소요된 시간으로 정의된다.

그림 5는 정규화된 부하가 적용되었을 때 각각의 고장환경에 대한 f-cube4와 f-GA의 정규화된 처리량을 보여준다. 두 그래프를 살펴보면, 3% 고장의 경우에 f-cube4의 처리량은 0.36에 수렴하는 반면에 f-GA의 처리량은 0.39에 수렴하고, 5% 고장의 경우에 f-cube4와 f-GA의 처리량은 각각 0.31, 0.34에 수렴한다. 즉, 요소고장이 존재하는 동일한 네트워크 환경에서는 f-GA가 f-cube4보다 높은 네트워크 처리량을 갖는다. 그림 6은 f-cube4와 f-GA의 메시지 지연시간을 보여준다. 두 그래프에서 네트워크에 노드고장이 존재하는 경우를 살펴보면, 같은 부하 조건하에서 f-GA가 f-cube4보다 낮은 지연시간을 갖는다. 이러한 사실들은 f-GA가 f-cube4보다 요소고장을 가진 네트워크에서 요소고장의 영향을 상대적으로 적게 받는다는 의미로 해석된다. 따라서, 안정성 및 고신뢰도를 요구하는 2차원 메쉬 네트워크에서 열악한 환경으로 인한 네트워크 자원의 고장발생시 안정된 메시지 전달, 즉, 네트워크의 효율적 운영을 위해서는 f-cube4보다 본 논문에서 제안된 f-GA을 사용하는 것이 효과적이라고 결론지을 수 있다.

4. 결 론

열악한 환경에서 동작하는 노드나 채널 등의 네트워크 요소들은 언제나 고장이 발생할 확률을 가지고 있다. 그러므로, 안정성 및 고신뢰도를 요구하는 시스템은 요소고장이 발생한 상황에서도 정상적으로 시스템이 동작할 수 있도록 설계되어야 한다. 이를 위한 하나의 방법으로

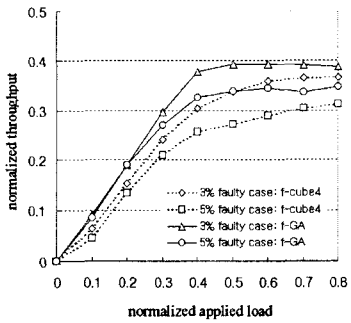

그림 5. 네트워크에 적용된 부하에 따른 f-GA와 f-cube4의 정규화된 네트워크 처리량 변화

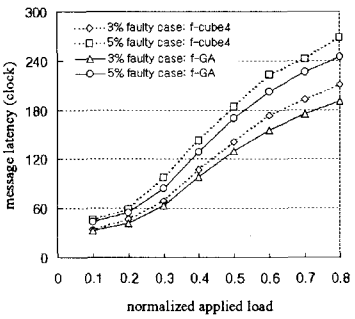

그림 6. 네트워크에 적용된 부하에 따른 f-GA와 f-cube4의 메시지 지연시간 변화

본 논문에서는 유전 알고리즘을 이용한 고장포용 라우팅을 제안하였으며, 시뮬레이션을 통하여 제안된 알고리즘의 성능이 우수함을 증명하였다.

본 논문에서 제안한 고장포용 라우팅 알고리즘은 유전 알고리즘을 사용하였기 때문에 여러 방면으로의 확장이 용이하다. 예를 들어, 혼잡제어 등과 같은 복잡한 주변 정보를 염색체로 표현되는 라우팅표에 추가시킴으로써 보다 효과적인 라우팅표를 얻어낼 수 있다. 그러나 라우팅표를 나타내는 염색체의 길이가 너무 길면 유전 알고리즘의 학습횟수가 지수적으로 증가한다. 따라서, 이러한 확장에 있어서는 합리적인 염색체 구성이 중요하다.

[참 고 문 헌]

[1] R. V. Boppana and S. Chalasani, "Fault-Tolerant Wormhole Routing Algorithms for Mesh Networks," *IEEE Trans. on Computers*, vol. 44, no. 7, pp.848-864, July, 1995

[2] Y. M. Boura and C. R. Das, "Fault-Tolerant Routing in Mesh Networks," *International Conference on Parallel Processing*, 1995

[3] C. C. Su and K. G. Shin, "Adaptive Fault-Tolerant Deadlock-Free Routing in Meshes and Hypercubes", *IEEE Trans. Computers*, Vol. 45, No. 6, pp.666-683, June, 1996

[4] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. on Computers*, vol. C-36, no. 5, pp.547-553, May, 1987

[5] W. J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. on Parallel and Distributed Systems*, vol 3, No. 2, pp.194-205, March, 1992

[6] J. H. Holland, "Adaptation in Natural and Artificial Systems," Ann Arbor, MI: Univ. of Michigan Press, 1975