채널배선 문제에 대한 분산 평균장 유전자 알고리즘

홍철의*

Distributed Mean Field Genetic Algorithm for Channel Routing

Chuleui Hong*

이 논문은 2008년도 상명대학교 교내 연구비를 지원받았음

요 약

본 논문에서는 MPI(Message Passing Interface) 환경 하에서 채널배선 문제에 대한 분산 평균장 유전자 알고리즘(MGA, Mean field Genetic Algorithm)이라는 새로운 최적화 알고리즘을 제안한다. 분산 MGA는 평균장 어닐링(MFA, Mean Field Annealing)과 시뮬레이티드 어닐링 형태의 유전자 알고리즘(SGA, Simulated annealing-like Genetic Algorithm)을 결합한 경험적 알고리즘이다. 평균장 어닐링의 빠른 평형상태 도달과 유전자 알고리즘의 다양하고 강력한 연산자를 합성하여 최적화 문제를 효율적으로 해결하였다. 제안된 분산 MGA를 VLSI 설계에서 중요한 주제인 채널 배선문제에 적용하여 실험한 결과 기존의 GA를 단독으로 사용하였을 때보다 최적해에 빠르게 도달하였다. 또한 분산 알고리즘은 순차 알고리즘에서의 최적해 수렴 특성을 해치지 않으면서 문제의 크기에 대하여 선형적인 수행시간 단축을 나타냈다.

ABSTRACT

In this paper, we introduce a novel approach to optimization algorithm which is a distributed Mean field Genetic algorithm (MGA) implemented in MPI(Message Passing Interface) environments. Distributed MGA is a hybrid algorithm of Mean Field Annealing(MFA) and Simulated annealing-like Genetic Algorithm(SGA). The proposed distributed MGA combines the benefit of rapid convergence property of MFA and the effective genetic operations of SGA. The proposed distributed MGA is applied to the channel routing problem, which is an important issue in the automatic layout design of VLSI circuits. Our experimental results show that the composition of heuristic methods improves the performance over GA alone in terms of mean execution time. It is also proved that the proposed distributed algorithm maintains the convergence properties of sequential algorithm while it achieves almost linear speedup as the problem size increases.

키워드

병렬/분산처리, 평균장 어닐링, 시뮬레이티드 어닐링, 유전자 알고리즘, 채널배선

Key word

parallel/distributed processing, mean field annealing, simulated annealing, genetic algorithm, channel routing

* 상명대학교 컴퓨터과학부

접수일자 : 2009. 12. 09
심사완료일자 : 2009. 12. 24

Ⅰ. 서 론

본 논문에서는 채널배선 문제에 적용할 수 있는 합성 알고리즘의 분산처리 기법을 제안한다. 제안된 분산 평균장 유전자 알고리즘(MGA, Mean field Genetic Algorithm)은 평균장 어닐링(MFA, Mean Field Annealing)[1,2,3,4]과 유전자 알고리즘(GA, Genetic Algorithm)[5,6]의 합성 알고리즘이다. 시뮬레이티드 어닐링(SA, Simulated Annealing)[4]이 평형 상태에 도달하는데 오랜 시간이 걸리는데 비하여 평균장 어닐링은 평형상태에 빠르게 도달하는 특성을 가지고 있다. 이는 시뮬레이티드 어닐링이 순차적인 상태변환으로 평형 상태에 도달하는데 평균장 어닐링은 시스템의 상태를 평균장 근사법(mean-field approximation)으로 간략하게 해결하기 때문이다. MFA는 SA 기법에서 임의로 상태를 변화시키는 것과 달리 평균장 근사법을 사용하여 산출되는 평균값으로 대치시키는 방법으로 평형상태에 빠르게 도달한다.

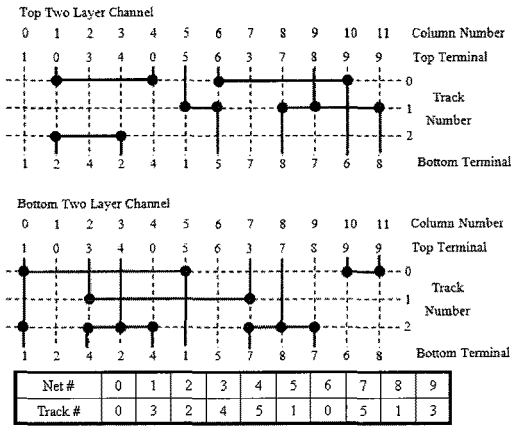
유전자 알고리즘을 평균장 어닐링과 결합시키기 위하여 전형적인 유전자 알고리즘을 변경하였다. 유전자 알고리즘에서 유전자 선택 시 시뮬레이티드 어닐링에서 사용하는 Metropolis Criteria에 의해서 새로이 생성된 상태를 선택하게 하였다[7]. 이렇게 함으로서 평균장 어닐링에서 도달한 평형상태가 유전자 알고리즘에서도 유지된다. 이렇게 변경된 유전자 알고리즘을 시뮬레이티드 어닐링 형태의 유전자 알고리즘(SGA, Simulated annealing-like Genetic Algorithm)라 부르기로 한다.

일반적으로 최적화 알고리즘은 전역 최적 상태에 도달하는 오랜 시간이 걸린다. 따라서 본 논문에서는 제안된 최적 알고리즘을 분산처리에 의해서 빠르게 해를 구하고자 한다. 분산화는 2단계로 나누어 적용된다. 먼저 평균장 어닐링에 적용되고 평균장 어닐링의 결과를 받아 최적화 작업을 수행하는 SGA에 적용한다.

제안된 분산 합성 알고리즘은 최적화 문제에 일반적으로 적용 가능하나 알고리즘의 성능을 측정하기 위하여 최적화 문제의 하나인 채널배선(channel routing) 문제에 적용하였다.

Ⅱ. 채널배선 문제

채널배선 문제는 VLSI 회로 및 기판의 설계 자동화에서 중요한 분야이다. 채널배선 문제에서 채널은 터미널(terminal)이라 불리는 점들로 이루어진 2개의 수평 행으로 구성되며 터미널은 기판의 열에 맞추어서 일정한 간격으로 배치된다. 트랙(track)은 미리 기판에 주어져 있는 연결 가능한 행으로 배선 경로를 나타낸다. 네트(net)는 같은 번호의 터미널이 배선 경로, 즉 트랙을 따라 연결되어 있는 선을 말하며 다른 네트와는 겹쳐져 있으면 안 된다. 층(layer)은 터미널을 연결한 네트의 배선 영역을 말한다. 따라서 본 논문에서 정의하는 채널배선 문제란 주어진 모든 터미널을 연결하는 네트를 구현하는데 층의 개수를 최소화하는 문제이다. 그림 1은 2개의 층으로 구현된 채널배선 문제의 예를 보여준다.



Net #	0	1	2	3	4	5	6	7	8	9
Track #	0	3	2	4	5	1	0	5	1	3

그림 1. 10-네트, 3-트랙, 2-층의 채널배선 문제의 예
Fig. 1 Channel Routing for 10-Net, 3-Track, 2-Layer Problem

채널배선 문제는 NP-완전 문제로 아직까지 전역 최적해를 찾는 다항식 알고리즘은 개발되어 있지 않다[8, 9]. 또한 해를 구하는데 긴 시간이 걸리므로 knock-knee 모델, 시뮬레이티드 어닐링, 신경망을 이용하여 짧은 시간에 해를 구하고자 병렬화를 연구하였다[10,11].

Ⅲ. 분산 평균장 유전자 알고리즘(MGA)

본 논문에서 제안된 분산 평균장 유전자 알고리즘(MGA)은 평균장 어닐링과 시뮬레이티드 어닐링의 특성을 지닌 유전자 알고리즘을 결합한 합성 알고리즘을 분산처리가 가능하도록 개선한 최적화 알고리즘이다.

평균장 어닐링(MFA)은 시뮬레이티드 어닐링(SA)에 비하여 빠르게 열 평형상태에 도달하여 최적해를 빠르게 얻을 수 있으나 최종 결과 좋지 않은 단점을 가지고 있다. 반면에, 유전자 알고리즘(GA)은 선택, 교차, 돌연변이와 같은 유용하며 풍부한 연산자를 보유하고 있어 최적해 접근 특성이 우월한 반면 어닐링 알고리즘에서 사용하는 온도 개념을 가지고 있지 않아 두 알고리즘을 바로 결합 할 수는 없다.

제안된 분산 MGA에서는 유전자 알고리즘에 시뮬레이티드 어닐링에서 사용하는 Metropolis criteria를 교차 및 돌연변이 연산자에 적용하여 온도 변화에 따른 열 평형상태를 유지하도록 하였다. 따라서 제안된 분산 MGA에서는 MFA의 빠른 최적해 접근 특성과 GA의 풍부한 연산자를 이용한 우수한 최적해를 도출하는 장점을 결합한 합성 알고리즘이다.

분산 MGA에서는 해의 상태를 비용함수, $C(s)$,로 표현하며, 비용함수의 값이 가장 작은 상태가 본 실험에서 얻고자 하는 최적해를 의미한다. 채널배선 문제에서 주어진 네트가 트랙에 배정될 때 수직 또는 수평 방향으로 다른 네트와 겹침이 발생할 수 있다. 따라서 비용 함수는 겹침이 없는 네트의 배정을 찾고자 하며 모든 네트가 트랙에 배정되었을 때의 겹침의 개수로 표현한다.

$$C(s)=\sum_{i=0}^{N-1}\sum_{j\neq i}\sum_{p=0}^{M-1} s_{ip}\,s_{jp}\,o_{ij} \qquad (1)$$

N: 총 네트 수
M: 총 트랙 수
s_{ip}: 네트 i가 트랙 p에 배정될 확률
s_{jp}: 네트 j가 트랙 p에 배정될 확률
o_{ij}: 네트 i와 j가 겹치면 1, 아니면 0

3.1. 분산 평균장 어닐링(MFA)

평균장 어닐링(Mean Field Annealing, MFA)은 물리학에서 평균장 근사법(mean-field approximation)에 기초한 시뮬레이티드 어닐링(SA)으로부터 파생되었다. 시뮬레이티드 어닐링이 상태변환을 임으로(random) 수행하는 반면에 평균장 어닐링은 평균장 근사법을 사용하여 계산되는 평균값의 상태에 접근하게 하여 열(온도) 평형상태에 빠르게 도달한다.

MFA에서는 스핀 행렬(spin matrix)로 해의 상태를 표현한다. 본 논문에서는 채널배선 문제에서의 스핀 행렬(spin matrix)을 네트를 나타내는 N개의 행과 배정된 트랙을 나타내는 M개의 열로 구성하여 각 네트가 트랙에 배정된 상태를 확률로 표현한다. 스핀 행렬의 요소인 스핀 (i,p)의 값 s_{ip} 는 네트 i를 트랙 p에 배당할 확률을 표시한다. 여기에서 s_{ip}는 $0\leq s_{ip}\leq 1$범위에서 연속 변수이다.

MFA에서 초기 상태의 스핀 값은 임의의 네트가 임의의 트랙에 배정될 확률은 같게 정한다. 따라서 스핀 행렬의 초기 스핀 값은 전체 트랙 수분의1로 정한다. 최적화 단계가 진행 되어 최적해에 도달하면 스핀 값은 0 또는 1로 수렴한다. 만일 s_{ip}가 1로 수렴되면 네트 i는 트랙 p에 최종 배정되는 것을 의미한다.

표 1는 그림 1의 예제에 대한 초기와 최종 상태의 스핀 행렬을 보여준다.

표 1. 그림 1에 대한 10×6스핀 행렬의 예
(a) 초기상태 (b) 최종해 상태
Table. 1 10×6 Spin Matrix of Fig. 1
(a) Initial State (b) Final State

	0	1	2	3	4	5
0	1/6	1/6	1/6	1/6	1/6	1/6
1	1/6	1/6	1/6	1/6	1/6	1/6
2	1/6	1/6	1/6	1/6	1/6	1/6
3	1/6	1/6	1/6	1/6	1/6	1/6
4	1/6	1/6	1/6	1/6	1/6	1/6
5	1/6	1/6	1/6	1/6	1/6	1/6
6	1/6	1/6	1/6	1/6	1/6	1/6
7	1/6	1/6	1/6	1/6	1/6	1/6
8	1/6	1/6	1/6	1/6	1/6	1/6
9	1/6	1/6	1/6	1/6	1/6	1/6

(a)

	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	0
3	0	0	0	0	1	0
4	0	0	0	0	0	1
5	0	1	0	0	0	0
6	1	0	0	0	0	0
7	0	0	0	0	0	1
8	0	1	0	0	0	0
9	0	0	0	1	0	0

(b)

평균장(mean field) Φ_{ip}는 다음과 같이 정의된다.

$$\Phi_{ip} = -\frac{\partial C(s)}{\partial s_{ip}} = -\sum_{j \neq i}^{N-1} s_{jp} o_{ij} \quad (2)$$

$-\Phi_{ip}$는 네트 i가 트랙 p에 배당될 때 얻어지는 비용함수의 감소를 나타낸다.

각각의 스핀 값 sip 는 $e^{\Phi_{ip}/T}$에 비례하므로 sip를 정규화 하면 다음과 같다. 여기서 T는 온도를 나타낸다.

$$s_{ip} = \frac{e^{\Phi_{ip}/T}}{\sum_{q=0}^{M-1} e^{\Phi_{iq}/T}} \quad (3)$$

식 (1)에 의해서 $C(s)$는 s_{ip}에 선형 비례하므로, 목적함수 변화 ΔC는 스핀(i, p)의 변화 값 Δs_{ip} 에 비례한다.

$$\Delta C = \sum_{p=0}^{M-1} \Delta C_{ip} = \sum_{p=0}^{M-1} \Phi_{ip} \Delta s_{ip} \quad (4)$$

여기서 $\Delta s_{ip} = s_{ip}^{new} - s_{ip}^{old}$로 정의된다.

평균장 어닐링을 분산처리 하기 위해서 $N \times M$스핀 행렬은 열 우선으로 나누어 스핀 행렬의 한열 또는 이웃한 여러 열을 분산처리에 사용되는 각 노드에 할당한다. 평균장 어닐링 수행 과정 중 네트 i가 임의로 선택되면 다른 열을 할당 받은 모든 노드가 스핀 값 s_{ip}를 계산하기 위한 연산에 참여한다. 여기서 노드 사이의 통신이 발생한다. 다음은 채널배선 문제에 대한 분산MFA 알고리즘의 의사 코드로 기술한 것이다.

while (비용변화(ΔC)가 지정된 작은 값 ε 보다 크다) **begin**

각 노드에서는 같은 seed를 사용하여 같은 네트 i 선택지역 평균장을 계산한다.

$$\Phi_{ip} = -\sum_{j \neq i}^{N-1} s_{jp} o_{ij} \quad \text{for } 0 \leq p \leq M-1$$

global-sum 연산을 사용하여 새로운 스핀 값을 계산 한다.

$$s_{ip} = e^{\Phi_{ip}/T} / \sum_{q=0}^{M-1} e^{\Phi_{iq}/T}, \text{ T: 온도}$$

global-sum 연산을 사용하여 스핀 값 갱신에 따른 비용변화를 계산 한다.

$$\Delta C = \sum_{p=0}^{M-1} \Phi_{ip} (s_{ip}^{new} - s_{ip})$$

i-행의 스핀 값을 갱신한다.

$$s_{ip} = s_{ip}^{new} \quad \text{for } 0 \leq p \leq M-1$$

global-collect 연산을 사용하여 i-행의 스핀 값 저장

end

MFA를 수행하는데 냉각 스케줄(cooling schedule)은 최종해에 많은 영향을 미친다. 그러므로 냉각 스케줄은 문제의 성격 및 목적함수에 따라 신중하게 선택하여야 한다. 일정 온도에서의 Markov 체인의 길이는 그 온도에서 평형상태에 도달하는데 필요한 상태변환의 수를 나타낸다. 본 문제에서는 Markov 체인의 길이를 네트의 개수, N, 번의 연속적인 상태변환 동안 비용함수의 값 즉, 비용 변화가 ε=0.5 보다 적을 때 평형상태에 도달한 것으로 간주한다.

3.2. 분산 시뮬레이티드 어닐링 형태의 유전자 알고리즘(SGA)

유전자 알고리즘은 생물학적 진화 모델에 근거한 강력한 경험적 탐색 방법이다. 유전자 알고리즘은VLSI 배선을 포함한 광범위한 NP-완전 최적화 문제에서 전역 최적해를 구하는 효과적인 방법으로 많은 연구가 되어 왔다[12].

먼저 유전자 알고리즘에 사용되는 유전자 표기법에 대하여 정의한다. n개의 네트, m개의 트랙, l개의 층을 가진 채널배선 문제에 대하여 신경망을 이용한 방법[13]은 $n \times m \times l$ 프로세싱 요소를 사용하여 신경망 구성하였다. 그러나 본 논문에서는 각 네트에 배정되는 트랙의 식별자로 염색체(chromosome)또는 개체(individual)를 구성한다. 즉, 염색체의 크기는 네트의 크기이며 염색체를 구성하는 유전자는 네트 순서대로 배정되는 트랙의 식별자, 즉 번호이다. 트랙의 식별자, p_i는 0에서부터 $m \times l$-1의 정수로 표현한다. 예를 들어 i번째 네트가 a번째 층 b번째 트랙에 배정되면 p_i= b+(a-1)×m로 표현한다. 신경망을 이용한 알고리즘에서는 m와 l이 증가함에 따라

프로세싱 요소가 급격히 증가하나 유전자 알고리즘에서의 염색체의 크기는 네트의 개수로 트랙과 층의 개수에는 무관하다. 예를 들어 그림 1에서와 같이 10-Net, 3-Track, 2-Layer 문제의 경우 신경망에서는 10×3×2=60의 프로세싱 요소가 필요하고 10-Net, 10-Track, 4-Layer에서는 400개의 프로세싱 요소가 필요하나 본 논문에서 사용하는 유전자 알고리즘에서는 앞의 두 경우 모두 10개의 정수로 염색체를 구성한다.

기존의 개체집단(population)으로부터 다음 세대의 개체집단은 roulette 기법을 이용하여 각 염색체의 비용함수의 크기에 반비례하여 선택한다. 즉, 작은 비용 함수를 갖는 염색체일수록 다음 집단의 구성원으로 선택될 확률이 높아진다.

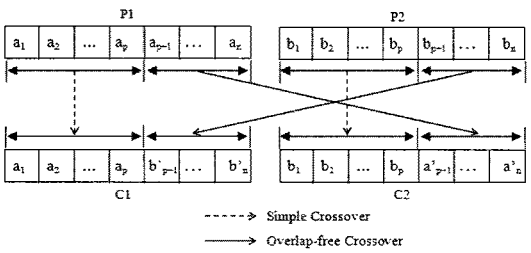


그림 2. 교차 연산
Fig. 2 Crossover Operation

본 논문에서는 그림 2에서와 같은 교차 연산자를 사용한다. 부모 *P1*과 *P2*로부터 교차 지점을 임의로(randomly) 선택하여 새로운 자식 *C1*과 *C2*를 생성한다. 제안된 교차 연산은 2 단계로 구성된다.

1. 단순 교차 : 첫 번째 단계에서는 부모 *P1*과 *P2*로부터 유전자를 복사하여 자식 *C1*과 *C2*를 생성한다. 그러나 기존의 단순 교차 연산과는 다르게 교차 지점으로부터 많은 유전자를 포함한 부분을 복사한다. 이렇게 함으로서 급격한 유전자 변화를 막을 수 있다.
2. 겹침 없는 교차: 두 번째 단계에서 나머지 유전자를 채운다. *C1*의 결정되지 않은 유전자는 *P2*의 유전자 부분으로부터 채워진다. 이 단계에서 새로이 채워지는 유전자가 이미 채워진 *C1*의 유전자와 수직 또는 수평적으로 겹침이 발생하는지를 조사한다. 만약 겹침이 발생하면 겹침이 없는 트랙 즉 유전자를 찾아서 배정한다. 겹침이 없는 트랙을 발견할 수 없을 때는 *P2*의 유전자를 그대로 복사한다. 이러한 방법으로 자식 *C2*도 부모 *P1*으로부터 염색체를 완성한다.

집단으로부터 인접한 2개의 염색체는 교차 확률, $P_{crossover}$,로 교차 연산을 수행한다. 교차 연산 후에 염색체내의 각 유전자는 $P_{mutation}$의 확률로 돌연변이 연산을 수행한다.

유전자 알고리즘에서는 온도 변화에 따른 열 평형상태에 대한 개념이 없다. 따라서 평균장 어닐링에서의 접근 특성을 유지하기 위하여 시뮬레이티드 어닐링에서와 같이 Metropolis Criterion을 사용하여 새로운 상태변환을 결정하도록 기존의 유전자 알고리즘을 변형하였다. 즉, 평균장 어닐링에서의 열 평형상태를 유지하기 위하여 유전 연산자에 의하여 생성된 새로운 상태는 Metropolis Criterion에 의하여 수락 또는 거절 한다. 다음 식에서 ΔC는 이전 상태로부터 새로운 상태로의 변화에 따른 비용변화를 의미하며, 이전 상태의 비용 값으로부터 새로운 상태의 비용 값을 빼서 얻어진다. T는 현재 온도를 의미한다.

$$\Pr[\Delta C \ is accepted] = \min\left(1, \exp\left(\frac{\Delta C}{T}\right)\right) \qquad (5)$$

본 논문에서 염색체 표현은 네트의 순서대로 할당되는 트랙번호의 순서를 문자열로 표현한다. 예를 들어, 문자열, "1,0,2,0,1"는 네트가 트랙에 배정된 상태로 0번 네트는 1번 트랙, 1번 네트는 0번 트랙, 2번 네트는 2번 트랙, 3번 네트는 0번 트랙, 4번 네트는 1번 트랙에 배정된 상태를 표현한다.

MFA에서 SGA로의 개체집단(population) 생성은 MFA에서 사용되는 $N\times M$ 스핀 행렬과 같은 확률로 랜덤하게 개체(염색체)를 생성한다. 예로서, 5개의 트랙에 배당되는 임의의 네트 i의 스핀 값(스핀 행렬의 i-행)이 0.2, 0.4, 0.1, 0.1, 0.2라면 SGA의 염색체를 표현하는 문자열의 i번째 문자가 0, 1, 2, 3, 4가 될 확률이 각각 0.2, 0.4, 0.1, 0.1, 0.2가 되도록 난수 발생기를 이용하여 문자열을 생성한다. 반대로 SGA 개체집단(population)으로부터 MFA 스핀 행렬 생성은 SGA 집단에서의 네트의 트랙 배정 비율로 스핀 행렬을 생성한다. 예로서 개체집단의 크

기인 염색체의 수가 10에서 네트-i에 배당된 트랙이 각각 0,1,0,1,0,1,0,1,0,0이면 네트-i의 스핀 값, 즉 i-행의 값은 0.6, 0.4, 0.0, 0.0,…로 설정된다.

이외에 선택(selection) 연산자는 집단의 전체 비용에 대한 개체의 비용의 반비례로 난수 발생기를 사용하여 개체를 선택하며, 생식(reproduction) 연산자는 선택 연산자에 의하여 선택된 개체를 가지고 집단의 크기만큼의 새로운 집단을 생성한다. 교차(crossover) 연산자는 개체의 순서대로 2개의 개체를 선택하여 유전자 즉, 네트의 트랙 배정 상태를 나타내는 문자열의 부분을 서로 교환한다. 교환될 태스크의 선택은 전체 태스크수의 1/4이하로 제한하였으며, 선택은 랜덤하게 이루어진다. 돌연변이(mutation) 연산자는 임의의 개체를 선정하여 교환 및 이동연산을 수행한다. 교환 확률은 0.1, 이동확률은 0.9로 설정하였다. 교환 연산자는 한 개체에서 두 네트를 임의로 설정하여 두 네트에 배정된 트랙을 서로 교환한다. 이동 연산자는 2개 이하의 네트를 임의로 선정하여 네트에 배정된 트랙을 무작위하게 변화시킨다. 또한 집단에서 최적의 비용을 가진 개체는 항상 집단에 남아 있도록 최적 유전자를 보전한다.

본 실험은 600Mhz의 개인용 컴퓨터로 구성된 MPI 환경에서 수행하였다. 각 개인용 컴퓨터는 MPI 환경에서 노드로 사용된다. 노드의 크기는 P로 정의한다. 각 노드의 집단(sub-population)의 크기는 네트의 수(N)로 정하였다. 따라서 전체 집단(global population)의 크기는 네트 크기와 노드 크기의 곱($N\times P$)으로 표현된다. 비용함수는 MFA와 같은 1차식 비용함수, 식 (1)을 사용하였다.

분산 SGA의 알고리즘을 살펴보면, 각 노드는 MFA의 스핀행렬로부터 개체를 랜덤하게 생성한 다음에 각 노드의 개체집단과 그에 따른 비용(fitness)을 계산한 후 다른 모든 노드에게 브로드캐스트하여 모든 노드가 전체 개체집단에 대한 복사본을 가진다. 각 노드는 생식 연산자를 사용하여 전체 개체집단으로부터 네트의 크기(N)만큼 개체를 생성한다. 각 노드는 순차 유전자 알고리즘을 병렬로 수행한다. 노드의 개체집단에 대하여 독립적으로 유전 연산자가 수행되고 평가되어 진다. 이처럼 분리된 진화가 발생하는 기간을 한 시대(epoch)라 하며 한 epoch 동안 발생하는 generation의 수를 epoch 길이라 정의한다. 각 노드는 epoch 길이 동안 독립적으로 유전 연산자를 수행한 후에 동기화를 이루어 전체 개체집단을 형성한다. 본 실험에서 epoch 길이는 네트의 수를 노드의 수로 나눈 값, N/P로 미리 정하였다. max_epoch는 유전자 알고리즘을 수행하는 동안 발생하는 epoch의 수로서 한 epoch에 한 개의 동기화가 이루어지므로 동기화의 개수를 나타내며, 본 실험에서는 노드의 개수 P로 정하였다.

다음은 각 노드에서 수행되는 변형된 분산 유전자 알고리즘의 의사코드를 보여 준다.

```
MFA 스핀 행렬로부터 노드의 개체집단(P_sub)을 생성한다.
for max_epoch times begin
 P_sub에 대한 fitness를 계산한다.
 for generation=1 until epoch_length begin
  노드의 개체집단으로부터 개체를 선택한다.
   //select 연산
  다음 노드의 개체 집단을 생성한다.
   //reproduce 연산
     for 새로운 개체집단으로부터 2개의 개체를 교대로
         선택한다. begin
   교차 연산자를 적용한다. //crossover 연산
   비용변화( ΔC)를 계산한다.
   if exp(- ΔC/T) > random[0,1] then
    새로운 개체를 수락한다.
  end
  for 모든 개체에 대하여 begin
   돌연변이 연산자를 적용한다. //mutation 연산
   비용변화( ΔC)를 계산한다.
   if exp(- ΔC/T) > random[0,1] then
    새로운 개체를 수락한다.
  end
 end
 P_sub를 다른 모든 노드에게 브로드캐스트한다.
   //전체 개체집단(P_global) 생성
 새로운 P_sub를 랜덤하게 선택한다.
 최적의 비용을 가진 개체를 보전한다.
end
```

3.3 분산 합성 알고리즘(MGA)

제안된 분산 합성 알고리즘은 채널배선 문제에 대한 최적화 알고리즘으로 분산처리가 가능하도록 수정하였다. MFA에서는 각 네트의 트랙 배정이 확률로 표현되므로 확률에 따라 유전자 알고리즘을 적용하기 위한 개체집단을 만들 수 있다. 따라서 온도 변화에 따라 먼저 MFA를 적용하여 열 평형상태에 도달하였을 때의 스

핀행렬로 개체집단을 만들어 유전자 연산을 수행한다. 다음, 개체집단 내의 유전자 비율에 따라 다음 온도의 어닐링을 위한 스핀행렬을 생성한다. 이와 같이 평균장 어닐링과 유전자 알고리즘을 충분히 낮은 온도에서 시스템이 동결(freeze)될 때까지 위의 연산을 반복 수행한다.

변수 및 문제, 노드 구조 초기화한다.
스핀 행렬을 생성 한다.
초기온도 T_0를 설정하여 $T = T_0$ 세팅한다.
while $T \geq T_f$(=최종 온도) **begin**
    MFA 수행
    스핀 행렬로부터 SGA 개체집단 생성
    SGA 수행;
    SGA 개체집단으로부터 MFA 스핀 행렬 생성
    $T = \alpha T$ 로 온도 감소
end

제안한 분산 MGA를 구현하기 위해 냉각스케줄(cooling scheduling)은 주어진 문제의 특성과 목적함수에 따라 민감하게 작용한다. 초기 온도(T_0)는 태스크 수(N) 만큼 평균장 어닐링을 수행하였을 때 비용의 변화가 허용치 ε(= 0.5) 이하일 확률이 95%이상일 때의 온도로 설정한다. 최종온도(T_f)는 네트의 수 N회의 온도 변화 동안 연속해서 비용의 변화가 ε/1000 내에 존재할 때의 온도로 설정한다. 임의의 온도 t에서의 Markov 체인 길이(L_t)는 각 온도에서의 평형상태에 도달하기 위한 상태변환의 회수로 연속해서 네트 수만큼 비용의 변화가 허용치 ε(=0.5)내에 존재할 때의 상태변화 수로 설정한다. 온도 감소율은 $T_{t+1} = \alpha T_t$로 온도 감소에 쓰이는 상수 α는 실험적으로 구해진 값으로 0.9로 정하였다.

Ⅳ. 시뮬레이션 결과

제안된 MGA는 최적화 알고리즘으로서 일반적으로 수행시간이 오래 걸리므로 수행시간을 단축하기 위하여 MPI 환경을 이용하여 분산처리 하였다. 시뮬레이션 환경은 10Mbps의 인터넷으로 연결되었으며 Linux 운영체제의 600Mhz의 개인용 컴퓨터로 구성된 MPI 환경이 사용 되었다. 실험 결과 교차 연산 확률이 0.6일 때 최적해에 빠르게 접근하였다. 반면에 최적해 접근 특성은 돌연변이 연산 확률이 변함에 따라 크게 변하였다. 이는 돌연변이 연산 확률이 커지면 무작위한 탐색으로 최적해 접근이 어려워지며, 반대로 돌연변이 연산 확률이 작아지면 지역 최적해로부터 탈출하지 못하는 것으로 이해된다. 표 2는 다양한 채널배선 문제에 대한 유전 연산자의 최적 확률 값을 나타낸다.

표 2. MGA에서의 유전 연산자 변수
Table. 2 Genetic Parameters of MGA Algorithm

# of nets	# of cols.	# of tracks	# of layers	population size	Prob of crossover	prob of mutation
10	12	3	2	32	0.6	0.05
21	43	6	2	128	0.6	0.05
45	90	8	2	128	0.6	0.01
47	84	9	2	128	0.6	0.01
54	103	9	2	128	0.6	0.01
55	119	9	2	128	0.6	0.01
61	128	10	2	256	0.6	0.005
72	175	11	2	256	0.6	0.005

MGA 합성 알고리즘의 근간이 되는 MFA 및 GA는 알고리즘 자체의 특성으로 인하여 병렬화가 쉽게 이루어진다. MFA에서의 스핀 행렬과 SGA에서의 개체집단은 MPI에서의 노드 수에 따라 균등하게 나누어서 배정되어 각 노드에서 MFA 및 SGA 연산을 독립적으로 병렬로 진행하였다. 따라서 분산 알고리즘은 순차 알고리즘에서의 최적해 품질을 유지하면서 수행시간을 줄일 수 있었다. 표3과 그림 3은 서로 다른 채널배선 문제에서의 병렬화 특성을 나타낸다.

표 3. MGA에서의 한 세대 당 평균 수행시간 (msec)
Table. 3 The Average Execution Time per one generation in msec

# of nets \ # of nodes	1	2	4	8	16	32
10	7	8	16	27	35	NA
21	45	27	28	30	37	40
45	613	310	171	96	60	55
47	160	100	69	58	56	55
54	180	101	75	64	58	52
55	191	102	80	66	57	56
61	481	325	204	139	118	103
72	2138	1089	584	311	198	148

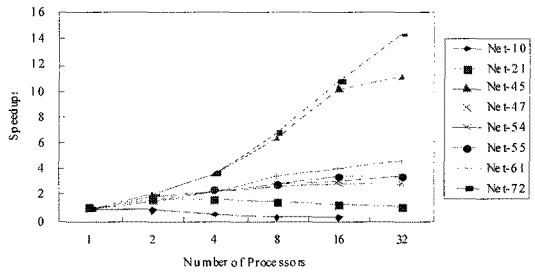


그림 3. 여러 채널배선 문제에서의 수행시간 단축
Fig. 3 Speedups of Different Nets

병렬화로 인한 수행시간 단축(parallel speedup)은 문제의 크기 및 개체집단의 크기에 비례하였다. 이는 문제의 크기가 증가할수록 각 온도에서 평형상태에 도달하는 시간이 오래 걸리고 제안된 분산 알고리즘은 각 온도마다 동기화가 이루어지므로 복잡한 문제에서 병렬화의 효과가 크게 나타난 것으로 분석된다. net-45와 net-72 문제의 경우 문제 자체가 최적해에 도달하는데 오랜 시간이 걸리므로 이에 따라 분산화에 의한 수행시간 단축도 다른 문제에 비하여 크게 나타났다. 이는 제안된 분산 합성 알고리즘(MGA)이 수행 시간이 오래 걸리는 복잡한 문제에 유용하다는 것을 보여준다. 참고로 그림 4는 대표적인 채널배선 문제의 해를 보여준다.

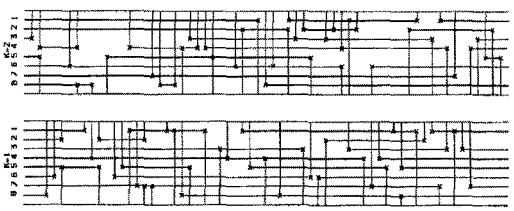


그림 4. 채널배선 문제에 대한 해의 예
Fig. 4 The Solutions of benchmark problems

V. 결 론

본 논문에서는 분산 평균장 유전자 알고리즘(MGA)이라 불리는 MFA와 GA의 장점을 결합한 새로운 합성 알고리즘을 제안하였다. 제안된 알고리즘은 여러 채널배선 벤치마크 문제[13]에 대하여 최적해를 성공적으로 구하였다. 본 알고리즘은 설계 자동화의 분할(partitioning) 및 배치(placement) 문제에 대해서도 손쉽게 적용할 수 있을 걸로 예상한다.

채널배선 문제의 최적해를 구하는데 제안된 합성 알고리즘(MGA)은 GA만 단독으로 사용할 때의 최적해 품질을 유지하면서 수행시간은 단축되었다. 반면에 기존의 MFA만을 사용하였을 때보다는 더 우수한 최적해를 구할 수 있으나 수행 시간이 오래 걸리었다. 따라서 본 논문에서는 MPI 환경을 이용하여 MGA를 분산처리하였다. 합성 알고리즘의 기본 알고리즘인 MFA 및 GA의 내재적 병렬화로 인하여 MGA의 분산화는 직접적이며 손쉽게 이루어 졌다. 분산 MGA는 문제의 크기가 증가할수록 즉, 복잡한 문제에 대하여 수행시간 단축의 효과가 크게 나타났으며, 분산 환경의 노드수를 증가하여 병렬화의 정도를 높여도 최적해의 품질이 유지됨을 보였다. 제안된 합성 알고리즘은 기존의 MFA 및 GA가 사용되는 광범위한 NP 문제에 적용 가능할 것으로 기대된다.

참고문헌

[1] Bultan, T. and C. Aykanat, "A New Mapping Heuristic Based on Mean Field Annealing," Journal of Parallel & Distributed Computing, 16, pp. 292-305, 1992.

[2] Salleh, S. and A. Y. Zomaya, "Multiprocessor Scheduling Using Mean-Field Annealing," Proc. of the First Workshop on Biologically Inspired Solutions to Parallel Processing Problems (BioSP3), pp. 288-296, March 30, Orlando, Florida, 1998.

[3] Yonghuai Liu, "A mean field annealing approach to accurate free form shape matching," Pattern Recognition, Volume 40, Issue 9, pp.2418-2436, September 2007.

[4] Pau Bofill, Roger Guimera, Carme Torras, "Comparison of simulated annealing and mean field annealing as applied to the generation of block designs," Neural Networks, Volume 16, Issue 10, pp. 1421-1428, December 2003.

[5] Zomaya, A.Y. and Teh Y.-W. "Observations on Using Genetic Algorithms for Dynamic Load-Balancing,"

IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 9 , pp. 899-911, Sept. 2001.

[6] Metin Ozkan, Ahmet Yazici, Muzaffer Kapanoglu, Osman Parlaktuna, "Hierarchical oriented genetic algorithms for coverage path planning of multi-robot teams with load balancing," GEC '09: Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation June 2009.

[7] 홍철의, 박경모, "평균장 어닐링과 유전자 알고리즘을 결합한 부하균형 기법,"정보과학회 논문지:시스템 및 이론, 제33권, 제8호, pp. 486-494, 2006.

[8] Susmita, S. and Bhargab, S. and Bhattacharya, B., "Manhattan-diagonal routing in channels and switchboxes," ACM Trans on DAES, Vol. 09, No. 01. 2004.

[9] Yang, C. and Chen, S. and Ho, J. and Tsai, C., "Efficient routability check algorithms for segmented channel routing," ACM Trans on DAES, Vol. 05, No. 03, 2000.

[10] Ning, X., "A Neural Network Optimization Algorithm For Channel Routing." Journal Of Computer Aided Design And Computer Graphics. Vol. 14, No. 1, 2002.

[11] Chen, S., "Bubble-Sort Approach to Channel Routing," IEE Proceedings Computer and Digital Techniques. Vol. 147, No. 6, 2000.

[12] W. Kim, C. Hong and Y. Kim, "Asynchronous Distributed Genetic Algorithm for Optimal Channel Routing," International Conference on Computational and Information Science, December, Shanghai, China, 2004.

[13] Funabiki, N. and Takefuji, Y., "A Parallel Algorithm for Channel Routing Problem." IEEE Trans. of CAD, Vol. 11, No. 4, pp.464-474, 1992.

저자소개

홍철의(Chuleui Hong)

1989년 미국 New Jersey Institute of Technology 전산학(석사)
1992년 미국 Univ. of Missouri - Rolla 전산학(박사)
1992년~1997년 한국전자통신연구원 선임연구원
1997년~현재 상명대학교 컴퓨터과학부 교수
※관심분야: 분산시스템 및 알고리즘, 최적화 기법, 멀티미디어 응용