한국정밀공학회지 제20권 제4호 (2003년 4월)
Journal of the Korean Society of Precision Engineering Vol. 20, No. 4, April 2003.

클러스터링 컴퓨터 시스템을 이용한 병렬화 유전자 알고리듬의 효율성 증대에 대한 연구

이원창#, 주지한*, 성활경**

A Study for Improvement Effect of Paralleled Genetic Algorithm by Using Clustering Computer System

Won Chang Lee#, Ji Han Ju* and Hwal Gyeong Seong**

ABSTRACT

Among the optimization method, GA (genetic algorithm) is a very powerful searching method enough to compete with design sensitivity analysis method. GA is very easy to apply, since it dose not require any design sensitivity information. However, GA has been computationally not efficient due to huge repetitive computation. In this study, parallel computation is adopted to improve computational efficiency. Paralleled GA is introduced on a clustered LINUX based personal computer system.

Key Words : GA(Genetic Algorithm,유전자 알고리듬), OS(Operating System,컴퓨터운영체제), NIC(Network Interface Card,네트워크 연결카드), PVM(Parallel Virtual Machine,클러스트링 시스템 구축을 위한 라이브러리), LINUX(a kind of computer OS,컴퓨터 운영체제의 하나)

1. 서론

1.1 연구 배경

현대 최적설계분야에서 널리 이용되고 있는 설계민감도이론을 대체할 이론으로 각광받고 있는 유전자 알고리듬은 자연 생태계의 적자생존 원리를 모델로 개발된 전산 프로그램용 알고리듬이다.[1] 유전자 알고리듬은 탐색 공간 전역에 분포된 유전자의 환경 적합도를 평가하여 우수한 개체만을 선택한 후 다음 세대를 구성하는 개체의 복제에 참여시킴으로써 이전 세대보다 더욱 진화된 우수한 특성으로 이루어진 개체인 최적해를 찾는 방법이다. 쉬운 사용방법과 우수한 전역 탐색을 통한 정확한 결과의 도출에도 불구하고 막대한 계산량을 요구하는 저효율성으로 복잡한 기계 구조물 문제의 최적화 기법으로 실제 적용하기를 꺼려하고 있는 것이 현실이다. 본 연구에서는 이러한 유전자 알고리즘의 최대 단점으로 인식되고 있는 저 효율성을 저가격의 IBM 호환 PC에 리눅스 운영체계를 기반으로 하는 시스템을 클러스트링 기법을 이용하여 연결하고 기존의 유전자 알고리듬 중 구조물의 해석과 개체 평가값을 도출하는 부분을 병렬 처리할 수 있는 병렬화 유전자 알고리듬을 작성하고 병렬화

2002년 9월 27일 접수
교신저자, 창원대학교 대학원 기계공학과
Email rainman@sccr.changwon.ac.kr Tel. (055) 263-4956
* 창원대학교 대학원 기계공학과
** 창원대학교 기계공학과

유전자 알고리듬을 이용하여 기존의 유전자 알고리듬이 가지는 저효율성을 개선하고 병렬화된 프로그램의 실효성을 검토하고자 한다. 이와 관련된 연구는 지금까지 전혀 시도 되어지지 않았다. 단지 SA(Simulated annealing algorithm)를 클러스터로 병렬처리를 시도한 적이 있었을 뿐[2] 유전자 알고리듬을 병렬화 시킨 사례는 찾아볼 수 없었다.

2. 유전자 알고리듬의 이해

2.1 유전자 알고리듬의 개념 및 용어[3]

자연계에 존재하는 생물은 개체의 독특한 특성을 지배하는 정보를 유전자에 담아 자손에게로 전달하여 기본 형질을 유지한다. 즉 자손은 부모의 유전정보를 나누어 받음으로써 새로운 개체이나 종의 특성을 잃지 않는 개체로 다시 태어나는 것이다. 이렇게 수 천, 수 백만 년을 이어져 오는 유전법칙을 수학적으로 변형시킨 것이 유전자 알고리듬이다. 유전자 알고리듬은 적자생존법칙을 응용한 탐색 알고리듬으로써 초기에 적당한 크기의 모집단을 생성시킨 후 모집단에 속한 각 개체에 대하여 주어진 환경에 대한 적합도(fitness)를 평가함으로써 개체의 진화형태와 모집단의 진화방향이 결정된다.

Table 1 Basic terms of genetic algorithm

용 어	설 명
Chromosome	염색체(유전정보를 갖는 개체)
Genes	유전자(정보전달매체)
Locus	유전자의 위치(설계변수의 종류)
Alleles	유전자값(설계변수의 값)
Phenotype	표현형(사상된 해)
Genotype	유전자형(역사상된 해)
Crossover	교차
Mutation	돌연변이
Population	모집단
Fitness	적합도
Encoding	사상
Decoding	역사상

유전자 알고리듬(genetic algorithm:GA)에서 염색체는 일차원의 배열로서 표현되고, 그 위에서의 위치가 유전자좌가 된다. 각 위치에서 갖게 되는 값은 유전자이고, 유전자형은 스트링상에 표현된 값의 패턴이다.

각 개체는 적합도의 크기에 비례하여 진화에 참여할 기회가 차등적으로 주어지므로 적합도가 낮은 개체는 도태되고 적응도가 높은 개체는 더 많은 부분에서 진화과정에 직,간접적으로 영향을 주며 참여하게 되는 것이다. 이러한 과정을 반복시킴으로써 점점 우수한 개체로 진화되어 목적으로 하는 최적해에 도달하게 되는 것이다.

적합도(fitness)란 주어진 환경에 대한 상대적인 적응정도라고 할 수 있다. 최적설계를 실행하기 위해서는 실수로 정의된 설계변수들을 유전자에 담기 위해 정수로 변환시키는 사상(encoding)과정을 거쳐 정수로 이루어진 유전자의 염색체(chromosome)로 변환된다. 염색체는 곧, 유전정보를 갖고 있는 하나의 개체임을 알 수 있다. 따라서, 역사상(decoding)과정을 통해 언제든지 염색체는 원래의 값 즉, 실수인 설계 변수 값으로 바꾸어 줄 수가 있으며 이를 통해 진화된 개체의 적합도가 계산되게 된다. 설계변수를 유전정보를 갖고 있는 염색체로 변환시키는 과정은 실수인 설계변수를 이진코드를 사용하여 비트(bit)당 0 혹은 1의 값을 갖는 m개의 비트(bit)로 구성되는 스트링(string)으로 표현되는 개체를 만드는 것으로써 m개의 설계변수에 대해 n 비트의 스트링(string)은 m그룹 1차원 배열로 개체를 생성시킨 후 난수 생성기(random number generation)를 통해 N개의 염색체를 무작위 선택함으로서 모집단이 구성된다.

이와 같이 구성된 설계변수들은 유전자 알고리듬의 기본 조작 과정을 거치면서 점점 진화되어 최적화된 시스템으로 개선되어 간다.

2.2 유전자 알고리듬의 기본 조작

2.2.1 복제 및 도태

주어진 환경에 대한 적응정도를 평가하는 것이 적합도(Fitness)이다. 적합도가 큰 개체들은 적합도의 크기에 따라 차등 적용되는 확률에 따라 자신의 유전자를 다음세대에 복제한다. 반대로 적합도가 낮은 개체는 도태하게 된다. 교배를 통한 유전자의 복제 및 도태는 적합도의 크기에 따라 이루어지므로 세대가 지날수록 초기 모집단의 전체 적합도 또한 향상되어 진다.

$$P_s = \frac{\Lambda_i}{(\sum_{i=0}^{n}\Lambda_i / n)} = \frac{\Lambda_i}{\overline{\Lambda}} \qquad (1)$$

$P_{s(i)}$: i번째개체가선택될확률

Λ_i : i번째개체의적합도(*Fitness*)

$\overline{\Lambda}$: 모집단(*Population*)의평균적합도

여기서, 우수한 유전자를 가지는 개체를 다음 세대로 복제하기 위해 적합도가 가장 우수한 개체(엘리트 개체)를 교차과정 없이 그대로 다음세대로 복제하는 엘리트전략과 우수한 두 개체를 이용하여 새로운 개체(교배쌍)를 복제하는 교배전략이 있다.

복제 및 도태를 통한 설계 개선은 적합도 분포에 의한 개체수에 많은 영향을 받는다. 만약 적합도가 높은 개체의 수가 적을 경우 국소해로 수렴될 경우가 많다. 반면 개체수가 너무 많게 되면 계산 량이 많아지므로 효율이 떨어지는 문제에 부딪칠 수 있다. 항상 효율성과 정확성을 고려해야만 하는 것이다.

2.2.2 교차(Crossover)

교차는 부모가 되는 그 개체의 유전자 일부분을 상호 교환시켜 새로운 유전자를 갖는 2개의 개체를 생성시키는 과정이다. 교차조작에 의해 부모개체의 일부 유전정보를 전수 받은 새로운 개체가 생성된다. 이와 같은 교차의 방법으로는 1점교차(one-point crossover), 다점교차(multi-point crossover) 및 균일교차(uniform crossover)가 대표적이며 이외에도 염색체상의 임의의 여러 개소에서 교차가 이루어지는 분할교차(segmented crossover)와 염색체가 연속적인 값을 취할 때 교배에 참여하는 두 개체의 중간값만을 인자로 계승받는 혼합교차(blended crossover)가 있다.

2.2.3 돌연변이(Mutation)

돌연변이란 교배에 참여한 두 부모 개체의 형질과는 상관없이 정해진 확률에 따라 임의 위치의 비트(bit)값을 변이 시키는 조작을 돌연변이라 한다. 초기 모집단 생성 후 복제 및 도태에 의한 교차만으로는 광범위한 최적개체의 탐색에 한계가 있다. 이와 같은 한계를 극복하고 한정된 탐색범위를 벗어나 새로운 탐색공간으로의 전이가 가능토록 함으로써 국소 최적치 수렴 구역을 탈출할 수 있도록 하기 위한 인위적인 조작을 돌연변이라고 할 수 있다.

따라서 낮은 돌연변이율은 국소 최적해를 제공하나 효율이 좋아지는 반면 너무 높은 돌연변이율은 수렴시간이 길어질 뿐만 아니라 심할 경우 수렴이 어려운 경우도 있다. 따라서, 대체로 0.5~1 % 사이의 돌연변이율(mutation probability)을 사용하게 된다. 이것은 문제에 따라 조금씩 다르게 적용될 수 있다.

2.3 유전자 알고리듬의 처리순서

유전자 알고리듬의 전산처리는 다음과 같은 과정으로 수행된다. 단, 처리과정중 유전자의 표현에서 유전자와 해 사이에는 1대1 대응이어야 하며 교차에 의해 생성된 개체는 양친의 형질이 계승되어야 한다.

1단계. 초기 모집단 생성

개체의 특성을 이진코드를 사용한 1차원 배열의 스트링(string)으로 표현하고, 초기 개체수 m에 대한 염색체 길이(chromosome length)를 결정한다. 이를 난수 발생에 따라 무작위로 선택하여 초기 모집단(population size) P_o를 형성하다.

$$P_o = \{ C_1, C_2, C_3, \cdots, C_m \} \qquad (2)$$

2단계. 적합도 평가

이진코드의 1차원배열로 표현된 개체의 적합도(fitness)를 각 개체별로 평가한다.

3단계. 새로운 모집단 P_t 생성

확률론적으로 적합도(fitness)가 높은 개체는 보다 많이 복제(reproduction)되어 교배율에 해당하는 수만큼의 교배쌍이 선택된다. 선택된 스트링(string)을 교차(crossover) 시키는 조작에 의해 새로운 모집단 P_t를 형성한다.

$$P_t = \{ C_1, C_2, C_3, \cdots, C_m, C_{m+1}, \cdots, C_{m+k} \}^T \qquad (3)$$

4단계. 돌연변이조작

새로 구성된 모집단 P_t중 초기에 주어진 돌연변이율의 확률에 따라 무작위로 선택된 개체의 염색체중 특정위치 비트(bit)를 변이 시킨다.

5단계. 차세대 모집단 P_{t+1} 구성

새로 구성된 모집단 P_t의 각 개체에 대한 적합도를 계산한다. 이를 토대로 모집단 P_t의 평균 적합도를 계산한 후 각 개체의 적합도를 평가한다. 평가된 각 개체의 적합도로부터 선택확률을 결정한다. 이를 기초로 다시 m개의 개체를 선택하여 다음세대의 모집단 P_{t+1}를 구성한다.

6단계. 최적화 평가

상기 과정을 거치면 한 세대(Generation)가 끝난다. 초기에 주어지는 조건의 세대 수 만큼 혹은 수렴기준을 만족할 때까지 상기 과정을 반복시키면 설계변수는 원하는 최적해로 수렴하게 된다.

3. 병렬화 유전자 알고리듬

3.1 유전자 알고리듬의 병렬화

앞에서 살펴본 유전자 알고리듬[3]은 우수한 병렬화 가능성을 포함하고 있다. 초기 모집단을 구성한 후 모집단을 구성하고 있는 임의의 개체들을 보면 모든 개체가 각 개체 서로간에 대하여 상호 독립된 성격을 가지고 있다는 것을 알 수 있다. 그리고, 외부 혹은 내장된 해석 프로그램을 이용하여 설계 모델을 해석 할 때에도 각 개체들의 설계 값은 서로 연관 관계가 전혀 없다는 것을 알 수 있다. 이는 개체 수 만큼의 서로 다른 결과값을 만들어 내고 있으며 만들어진 결과값을 이용하여 각 개체들의 우수성을 비교 할 수 있는 객관적이고 상호 독립적인 평가값을 도출한다. 이러한 과정은 한 세대에 속해 있는 모든 개체의 수 만큼 반복하게 되는 것이다. 모든 개체에 대한 평가 작업이 끝이 나면 각 개체들의 평가값을 기준으로 우수한 개체와 열등한 개체를 선별하고 초기 입력값에 기준하여 새로운 모집단을 구성할 새로운 개체들을 생성 시킨다. 이상과 같은 과정을 하나의 단위로 하여 최적화가 이루어 졌다고 판단될 때까지 같은 연산을 반복하게 되는 것이다. 여기서, 우리는 모집단을 이루는 개체의 평가가 상호 독립적이며 개체 서로 간에 어떠한 종속관계나 연계성이 없음을 알 수 있다. 이 부분을 병렬화를 통해 동시에 연산과 평가가 이루어진다면 유전자 알고리듬의 효율성의 증대에 기여할 수 있을 것이다[4]. 부연 설명 하자면 현재와 같은 순차적 알고리듬에서는 만약 각 세대의 모집단이 10개의 개체로 구성되어 있다면 실제 모델을 해석하는 가장 많은 시간이 소요되는 부분인 각 개체의 설계값을 기반으로 설계 모델을 해석하고, 해석 후 나온 결과값을 이용하여 다시 평가 자료를 만드는 부분을 순차적으로 10번 반복해야 할 것이다. 하지만, 이 부분을 10개의 프로세스로 병렬화 시키면 1번의 연산 시간만에 10번의 계산이 이루어지고 이는 모든 개체의 평가 자료가 순차적 방법보다 이론상 10배의 효율성 증대를 가져올 것이라는 것을 알 수 있다. 또한 단지 해석 부분과 평가 부분만을 병렬화 시킨 것이므로 정확도는 순차적 방법과 완전히 동일하다.

한 세대의 유전자 알고리듬을 간단히 보면 다음과 같은 3단계를 반복한다는 것을 알 수 있다.

1. n개의 개체를 가지는 모집단 생성
2. n번의 개체 해석과 적응도 평가
3. 각 개체의 적합도 및 최적화 여부 평가

위의 3번 과정에서 설계 조건이 만족 되면 알고리듬은 끝이 난다. 하지만 그렇지 않으면 다시 1번 과정으로 되돌아가서 1, 2, 3번 과정을 반복하게 되는 것이다. 여기서, 1번과 3번 과정은 기계 구조물의 해석 시간이 포함된 2번 과정에 비해 거의 무시할 만한 수준의 연산 시간이다. 특히, 최적화할 모델이 복잡하고 거대해 질수록 이러한 경향은 더욱 심화된다. 본 연구에서는 2번 과정을 효율적으로 병렬화 하는 것이다.

3.2 병렬 처리를 위한 시스템 구성

본 연구를 위한 클러스트링 시스템은 IBM 호환 PC(펜티엄 120MHz, 메모리 64MB)에 리눅스 운영체계를 기반으로 노드를 구성하였으며 노드간 통신을 위해 beowulf project의 산물중 하나인 PVM(parallel virtual machine:PVM)[5]을 이용하여 구현 하였다. 유한 요소 해석툴은 Fortran77 언어를 이용하여 자체 제작한 PladeFEM을 이용하였

다.[6] 또한, 유전자 알고리듬의 각 단계 또한 Fortran77 언어를 이용하여 모듈화 시켜 재사용이 용이하도록 하였다. 그리고, 각 단계의 데이터 변환 및 노드간 통신을 위한 메시지 처리용 프로그램은 C++을 이용하여 완성 시켰다.

구성된 시스템을 간단히 살펴보면 1대의 NFS(network file system) 서버와 1대의 부하 분산 서버(load balance)와 다수의 노드라 불리는 연산 전용 컴퓨터들로 구성되어 있다.[7] 모든 노드들에서 사용되고 생성되는 데이터의 취합은 NFS서버에서 이루어지면 병렬화 준비 단계 및 각 노드들에게 연산을 위한 프로세스(process) 할당은 부하 분산 서버가 담당한다. 물론 NFS서버와 부하 분산 서버 또한 연산을 위한 노드에 포함 될 수 있다. 그리고, 각 노드들은 모델 해석을 위한 해석용 툴을 가지고 있으며 연산시 모두 서로 간에 일체의 간섭 없이 독립적으로 연산이 이루어지며 오직 연산 시작시와 연산 종료시에만 부하 분산 서버와 각 노드에 할당되어 있는 프로세스간 통신을 통해 새로운 작업을 할당 혹은 제한 당하게 된다. 모든 노드들과 NFS서버는 100Mbps NIC(network interface card:NIC)를 장착하고 있으며 100Mbps 스위칭 허브로 상호 연결되어 있다. 필요에 의해 2개 이상의 NIC를 장착하여 통신 속도를 높일 수도 있다. 그리고, 필요에 의해 노드는 100개 이상 추가로 연결 할 수도 있다. 본 연구에서는 부하 분산 서버와 NFS서버 외 두 대의 노드를 사용하였다. 다음은 간략한 시스템 구성도 이다.

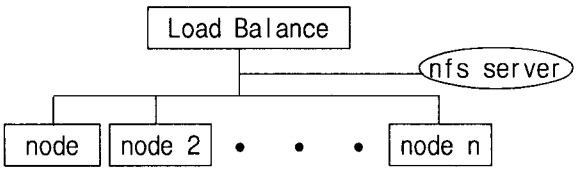

Fig. 1 Clustering system for paralleled genetic algorithm

3.3 3차원 구조물의 적용예

유전자 알고리듬을 3차원 구조물에 적용한 예제는 트러스(truss)요소 구조물에 대한 기하학적 치수 및 토폴로지 최적설계를 수행하기 위한 것으로서 25요소 트러스 구조물(twenty-five member transmission tower)을 예제화 하여 적용하였다.

3.3.1 트러스 구조물

25요소 트러스 구조(twenty-five member transmission tower)[8]에 대한 예제는 Fig. 2 의 구조를 모델로 하였으며 목적함수로는 구조물의 총질량이 사용되었고 제한조건식은 구조물에 정하중이 작용 될 때의 변위량(displacement)으로 하였다. 즉,

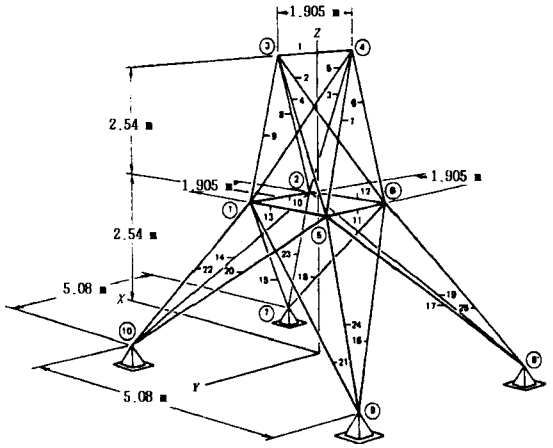

Fig. 2 Twenty-five member transmission tower mode

$$\text{Minimize} \quad f(\vec{x}) = \sum_i \rho A_i L_i \tag{4}$$

Subject to

$$\begin{aligned} g_{j1}(\vec{x}) &= |u_j| - \overline{u_j} \le 0 \\ g_{j2}(\vec{x}) &= |v_j| - \overline{v_j} \le 0 \\ g_{j3}(\vec{x}) &= |w_j| - \overline{w_j} \le 0 \end{aligned} \tag{5}$$

이다. 이때 u_j, v_j, w_j 는 j 번째 절점에서의 각 방향 변위이며 $\overline{u_j}$, $\overline{v_j}$, $\overline{w_j}$ 는 각 방향 변위의 상한치이다. 또한 ρ 는 재질의 밀도이고, A_i 및 L_i 는 i 번째 부재의 단면적과 길이이다.

위의 최적화 문제는 유전자 알고리듬을 적용시

키기 위하여 1차적으로 unconstrained optimization problem 으로 변환 시킨다.

$$F' = f + r\left[\sum_{j=1}^{n}\left(g_{j1}^{2} + g_{j2}^{2} + g_{j3}^{2} \right)\right] \qquad (6)$$

여기서, r은 벌칙 paramater이고, $\{g\}$ 함수는 아래와 같이 정의된다.

$$g = \begin{cases} 0, & g \le 0 \\ g, & g > 0 \end{cases} \qquad (7)$$

F'은 최적화가 될 수록 작은 값을 취하게 되나, 유전자 알고리듬에서 판단의 기준이 되는 적응도(fitness)는 최적화 될수록 크게 된다. 따라서,다음과 같이 적응도가 정의된다.

$$F = \frac{C_1}{C_2 + F'} \qquad (8)$$

단, C_1 및 C_2 는 적응도의 크기를 조절하기 위해 사용되는 상수이며 본 연구에서 C_1으로는 5000이 사용되었고 C_2 로는 1이 사용되었다. 하중 및 물성(material property)으로는 Table 2가 사용되었다.

Table 2 Design data for 25-member transmission tower

Node	Load, kN (kips)		
	X	Y	Z
1	2.225(0.5)	0.0(0.0)	0.0(0.0)
2	2.225(0.5)	0.0(0.0)	0.0(0.0)
3	4.450(1.0)	44.5(10.0)	-2.225(-5.0)
4	0.0(0.0)	44.5(10.0)	-2.225(-5.0)
Modulus of elasticity = 68.95 GPa(10^4 ksi) Material density = 27.15 kN/m^3 (0.10 lb/in^3) Upper limit on cross-sectional areas = None Displacement limits at all nodes and in all directions = 0.00889 m(0.35 in)			

25개의 트러스 부재는 Table 3과 같이 7개의 그룹을 형성하고 각 그룹을 2진 코드의 11비트(bit)로 배열(string)시켜 총 77비트의 배열을 1개의 개체로 하는 유전자 알고리듬 문제로 표현하였다. 이 경우 각 그룹에서 선택될 설계변수는 Table 4에서와 같이 0.1부터 4.0 사이의 연속적인 값으로부터 선택 된다.

Table 3 Binary strings assigned to design group

Design group	Design variables	Binary string (bit)
1	x_1	11
2	x_2, x_3, x_4, x_5	11
3	x_6, x_7, x_8, x_9	11
4	x_{10}, x_{11}, x_{12}, x_{13}	11
5	x_{14}, x_{15}, x_{16}, x_{17}	11
6	x_{18}, x_{19}, x_{20}, x_{21}	11
7	x_{22}, x_{23}, x_{24}, x_{25}	11
Total		77 bits

Table 4 Selectable sizing variables(cross-sectional area) for each topology

Topology (x)	Cross-sectional area, (in^2)						
	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	Group 7
1	0.1 ~ 4.0						
Binary string (bit)	11	11	11	11	11	11	11

특히 초기 모집단의 크기는 난수 발생기(random number generator)에 의해 생성되며 모집단의 크기는 160으로 하였고 유전자 연산을 위해 교차확률(crossover probability)은 0.85 로서 2점 교차를 사용하였으며 돌연변이 확률(Mutation probability)은 0.01로 하여 우수한 유전자가 생성되도록 하였다. 위의 확률값들은 통상적으로 사용되어지는 범위 안에서 몇 번의 예비 테스트를 통하여 잡은 값들이다.

4. 결과 및 고찰

연산과정을 통해 적합도 함수를 최대로 하면서 구조물의 무게를 최소로 하기 위한 수렴값은 Fig. 3 의 이력 곡선을 보면 100세대 이후부터 나타났다.

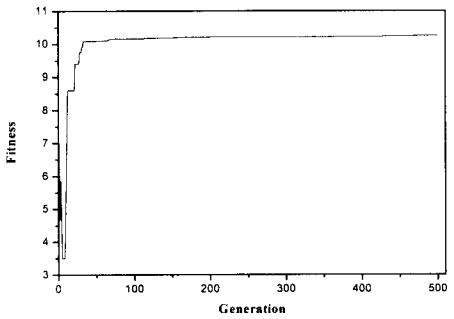

Fig. 3 Generaion-history of "most fitness" design for twenty-five member transmission tower

기존의 순차 방식과 본 연구에 사용된 병렬화 방식을 사용하여 얻어진 구조물의 최적화된 상태에서 발생된 최대변위와 무게는 Table 5와 같다.

Table 5 Comparison of topology results after 500 generations (traditional and paralleled genetic algorithm)

Design group	Optimum topology, m^2 (in^2)	
	traditional GA	paralleled GA
1	0.0670E-04(0.0104787)	0.0670E-04(0.0104787)
2	4.70000E-04(0.728627)	4.70000E-04(0.728627)
3	18.90095E-04(2.929653)	18.90095E-04(2.929653)
4	3.22116E-04(0.499281)	3.22116E-04(0.499281)
5	4.51927E-04(0.700488)	4.51927E-04(0.700488)
6	4.85209E-04(0.752076)	4.85209E-04(0.752076)
7	22.38204E-04(3.469223)	22.38204E-04(3.469223)
Max. Displ. m (inch)	8.57504E-04(0.03376)	8.57504E-04(0.03376)
Weight kN (lb)	2.086(468.7)	2.086(468.7)

해석 결과는 기존의 순차 방식과 본 연구에서 사용한 병렬화 방식이나 전혀 틀리지 않음을 알 수 있다.

유전자 알고리듬의 병렬화의 이점은 해석 시간의 단축을 통한 유전자 알고리듬의 효율을 높이는 것이다. Table 6은 유전자 알고리듬의 계산에 1대만을 사용하였을 때의 총 연산 시간과 3대를 사용 하였을 때의 총 연산 시간을 나타낸 것이다.

Table 6 Comparison of computing time

Reference	system	
	only one pentium computer	3 node pentium clustering compute
No. of Design Variables	7	7
Computational Time	9 hours 45 minutes for 500 generation (160 chromosome per Generation)	6 hours 10 minutes for 500 generation (160 chromosome per Generation)
Final Weight, kN (lb)	2.086(468.7)	2.086(468.7)

본 연구에서 관심을 가지는 효율성 부분에서 3대의 연산 노드를 사용 하였을 뿐인데도 대략 37% 정도의 효율이 향상 되었음을 알 수 있다. 연산 노드를 늘린다면 더욱 큰 폭으로 연산 효율이 증대 될 것이다. 마지막으로 여기서 사용된

Table 7 Comparison of topology results(ANSYS and PladeFEM)

Design group	topology, m^2 (in^2)	
	ANSYS 5.6	PladeFEM
1	0.06451E-04(0.010)	0.06451E-04(0.010)
2	13.21029E-04(2.0476)	13.21029E-04(2.0476)
3	19.33221E-04(2.9965)	19.33221E-04(2.9965)
4	0.06451E-04(0.0100)	0.06451E-04(0.0100)
5	4.42128E-04(0.6853)	4.42128E-04(0.6853)
6	10.46255E-04(1.6217)	10.46255E-04(1.6217)
7	17.23351E-04(2.6712)	17.23351E-04(2.6712)
Max. Disp. m (inch)	0.010409(0.40982)	0.010418(0.41017)
Max. average stress MPa (ksi)	109.637(15.901)	109.492(15.880)
Weight kN (lb)	2.445(549.533)	2.445(549.533)

PladeFEM(자체개발한 유한요소 해석용 프로그램)의 신뢰성 확보를 위해 Fig. 2의 25 요소 트러스 모델을 상용해석 프로그램인 .ANSYS 5.6과 PladeFEM으로 해석후 결과를 비교하였다. Table 7에 해석에 적용된 토폴로지와 간단한 해석 결과를 나타내었다. 각 절점에서의 변위와 요소의 평균 응력도 거의 같다고 보아도 무방한 수준임을 알 수 있다. 무게는 정확하게 같음을 알 수 있다.

5. 결론

본 연구에서는 우수한 전역 검색 기능과 정확한 결과를 가지는 유전자 알고리듬의 효율성을 높이기 위해 IBM 호환 PC에 리눅스 운영체계를 기반으로 한 단일 컴퓨터를 ethernet network으로 연결하여 구현한 클러스터를 이용하여 유전자 알고리듬의 병렬화를 시도해 본 결과 다음과 같은 결론을 얻었다.

1. 리눅스 클러스터에 병렬화된 유전자 알고리듬을 이용한 결과 기존의 단일 컴퓨팅 시스템에서 연산시 9시간 45분이 소비되던 문제를 6시간 10분만에 처리함으로써 효율이 37% 향상된 결과를 얻을 수 있었다. 본 연구 결과 유전자 알고리듬의 효율을 병렬화를 통해 개선 할 수 있다.

2. 유전자 알고리듬을 이용한 최적화 기법을 저렴한 IBM 호환 PC기반의 클러스터로 대신할 수 있다.

본 연구에서는 유전자 알고리듬의 병렬화에 따른 효율성 증대 여부와 저가이면서도 고성능의 최적설계 시스템 구현에 초점을 맞추었다.

현재 마땅한 리눅스용 구조 해석용 툴이 없는 문제를 해결하고 연산 노드당 시스템 규격을 높이고, 연산 노드를 더 많이 추가 장착하도록 시스템을 구성 한다면 더욱 효율을 높일 수 있을 것이다. 뿐만 아니라 연산중 예상치 못한 상황에서 일부의 시스템이 다운되는 상황에서도 중단되지 않고 꾸준히 해석이 가능하도록 fail over 기능을 구현하여 지금까지 전산 프로그램을 이용한 최적화 분야에서 제외 되었던 아주 크고 복잡한 댐의 형상 설계와 같은 실제모델의 해석에 적용할 수 있도록 발전시킬 계획이다.

참고문헌

1. Goldberg, D. E., "Genetic algorithm in search optimization, and machine learning," Addison-Wesley, pp. 1-25, pp. 136-137, 1989.
2. Park, H. S., Sung, C. W., "Optimization of Steel structures using distributed annealing algorithm on a cluster of personal conputers," International journal Computer and Structures, Vol. 80, pp. 1305 -1316, 2002.
3. 백운태, 성활경, "유전자 알고리즘에 의한 드릴링 머신의 설계 최적화 연구," 한국정밀공학회지, 제14권, 제12호, pp. 25-27, 1997.
4. 김종현, "병렬컴퓨터구조론," 생능출판사, pp. 27 -121, 1996.
5. Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., Sunderam, V. and Kowalik, J., "PVM: Parallel Virtual Machine A Users' Guide and Tutorial for Networked Parallel Computing," The MIT Press, pp. 11-43, 1994.
6. Cook, R. D., Malkus, D. S. and Plesha, M. E., "Concepts and Applications of Finite element analysis," WILEY, pp.31-57, 1989.
7. Kirch, O., Dawson, T., "Linux Network Administrator's Guide," O'reilly, pp. 365-374, 2000.
8. Haig, E. J., Arora, J. S., "Applied Optimal Design," A Wiley Interscience Publication, John Wiley & Sons, Inc., pp. 245-248, 1995.