병렬유전알고리즘을 응용한 대규모 전력계통의 최적 부하배분

論 文
48A - 4 - 6

Optimal Economic Load Dispatch using Parallel Genetic Algorithms in Large Scale Power Systems

金 泰 均* · 金 圭 浩** · 劉 錫 九***
(Tae-Kyun Kim · Kyu-Ho Kim · Seok-Ku You)

Abstract - This paper is concerned with an application of Parallel Genetic Algorithms(PGA) to optimal economic load dispatch(ELD) in power systems.

The ELD problem is to minimize the total generation fuel cost of power outputs for all generating units while satisfying load balancing constraints. Genetic Algorithms(GA) is a good candidate for effective parallelization because of their inherent principle of evolving in parallel a population of individuals. Each individual of a population evaluates the fitness function without data exchanges between individuals. In application of the parallel processing to GA, it is possible to use Single Instruction stream, Multiple Data stream(SIMD), a kind of parallel system. The architecture of SIMD system need not data communications between processors assigned.

The proposed ELD problem with C code is implemented by SIMSCRIPT language for parallel processing which is a powerful, free-form and versatile computer simulation programming language. The proposed algorithms has been tested for 38 units system and has been compared with Sequential Quadratic programming(SQP).

Key Words : Parallel Processing, Genetic Algorithms, Economic Load Dispatch, SQP, SIMD

1. 서 론

대규모 전력계통에서는 계통부하를 여러 발전기들이 나누어 담당하고 있다. 그 결과 각 발전기의 연료 변환률이 서로 차이가 있을 수 있으므로, 각 발전기가 같은 양의 연료를 사용하여도 발전출력은 다를 수 있다. 따라서 총 부하를 각 발전기에 적절히 분담시켜 총 연료비용을 최소화하는 것을 최적경제 부하배분(Economic Load Dispatch : ELD)이라 한다. 총 발전량에 필요한 연료비용을 최소화하기 위하여, 각 발전기의 연료비용은 발전출력의 2차 다항식으로 표현될 수 있으며, 각 발전기 출력의 상하한치와 전력수급 조건과 같은 제약조건이 만족되어야 한다[1].

정식화된 ELD 문제는 Lambda 반복법, Gradient 방법, 신경회로망 그리고 유전알고리즘(Genetic Algorithms : GA)과 같은 최적화 방법을 이용하여 해를 탐색할 수 있는 방안이 연구되었다[1-5]. 특히, Lambda 반복법은 발전 출력의 초기값을 설정하여 시작하고 최적해를 얻을 때까지 수치적인 반복에 의해 반복적으로 값들을 수정해 나간다[1].

신경회로망에서는 연속적, 혹은 이산적인 연료비용 함수를 갖는 발전기들에 대해 ELD 문제를 풀 수 있는 Hopfield 모델이 사용되었다[3,4].

또한, 유전알고리즘은 ELD뿐 만 아니라 전력계통의 여러 수치해석 문제에 적용되어 왔으며, 이 방법은 자연선택과 진화에 근거하여 세대가 변함에 따라 전역적 최적해를 탐색할 수 있는 전역적 탐색기법이다[5,6]. 그러나 유전알고리즘의 결점은 계산량이 많다는 것이다. 즉 최적해를 탐색하는데 있어서 정식화된 문제의 규모에 따라 매우 많은 시간이 걸린다. 이것은 유전알고리즘의 고유한 특성상 한 점에서 해를 찾아가는 것이 아니라 여러 개의 점에서 동시에 해를 찾아나가기 때문이다. 여러 개의 점에서 동시에 해를 탐색하는 즉, 병렬로 전개되는 특성으로부터 유전알고리즘에 병렬처리를 응용할 수 있으며, 이것을 병렬유전알고리즘(Parallel Genetic Algorithms : PGA)이라 한다. 결국, 유전알고리즘은 순차적인 처리를 목적으로 알고리즘의 간단한 조정에 의하여 병렬처리에 응용하기에 적합한 알고리즘이라는 것을 알 수 있다.

유전알고리즘에 있어서 해를 탐색하기 위한 과정에서 초기화, 적합도 평가, 복제, 돌연변이 등은 개체들 사이에 데이터를 교환하지 않고 독립적으로 진행되며, 교차는 선정된 두 개체 사이에 종속적으로 진행된다. 유전알고리즘에 병렬처리를 응용함에 있어서, 예를들어 적합도 평가의 경우 개체들 간에

* 正 會 員 : 韓國電力研究員 先任研究員 · 工博
** 正 會 員 : 安山工科大學 電氣科 專任講師 · 工博
*** 正 會 員 : 漢陽大 工大 電氣工學科 敎授 · 工博
接受日子 : 1999年 1月 4日
最終完了 : 1999年 3月 25日

데이터의 교환 등이 필요 없으므로, 각각의 개체에 각각의 프로세서를 할당시켜 적합도를 평가하게 할 수 있다. 이러한 특성은 병렬처리 구조의 하나인 SIMD(single instruction stream, multiple data stream) 시스템에 구축하는 것이 적합하다. 이 SIMD 시스템은 모든 프로세서가 동일한 명령어를 동시에 실행하여, 각 프로세서에서 주어진 데이터를 처리하는 시스템이다[7].

본 논문에서는 최적경제 부하배분 문제에 병렬유전알고리즘 적용의 가능성과 그 효과를 분석하였다. 병렬처리에 유용한 SIMSCRIPT 언어가 사용되었는데, 이는 매우 강력하고, 자유 형식이며 용도가 다양한 컴퓨터 시뮬레이션 프로그래밍 언어이다.

제안된 알고리즘의 효용성을 입증하기 위하여 SIMSCRIPT 언어를 사용하여 다수의 프로세서를 대상으로 38기 계통에 적용하였으며, Sequential Quadratic programming(SQP)의 결과와 비교하였다.

2. 문제의 정식화

전력계통에서의 최적경제 부하배분은 전력수급조건과 각 발전기출력의 제약을 만족하면서 계통내 발전기의 총 연료비용을 최소화시키는 문제로 표현할 수 있다[1].

목적함수

계통의 연료비용은 발전기 유효전력의 2차 함수의 합으로 표현된다.

$$\text{Min.} \quad \sum_{i\in\Omega}(a_i + b_iP_{Gi} + c_iP_{Gi}^2) \tag{1}$$

제약조건

(i) 등식제약조건

만족되어야할 전력수급조건은 다음과 같다.

$$\sum_{i\in\Omega} P_{Gi} = P_{Demand} + P_{Loss} \tag{2}$$

(ii) 부등식제약조건

각 발전기의 출력은 다음과 같은 제약을 갖는다.

$$P_{Gi}^{min} \le P_{Gi} \le P_{Gi}^{max} \tag{3}$$

여기서, i : 발전기모선 인덱스

a_i, b_i, c_i : i 번째 발전기의 발전비용계수

Ω : 발전가능한 발전기들의 집합

P_{Gi} : i 번째 발전기 출력

P_{Demand} : 총 부하

P_{Loss} : 계통손실

$(.)^{max}$, $(.)^{min}$: 상하한치

최적 경제부하배분을 위한 정식화 문제의 등식제약조건을 식(4)과 같이 페널티 함수를 이용하여 등식제약조건을 갖지 않는 최적화문제로 바꿀 수 있다.

$$\begin{aligned}\text{Min.} \quad P(\mathbf{x}) &= \sum_{i\in\Omega}(a_i + b_iP_{Gi} + c_iP_{Gi}^2) \\ &+ \mu\sum[\sum_{i\in\Omega} P_{Gi} - P_{Demand} - P_{Loss}]^2\end{aligned} \tag{4}$$

μ는 페널티 계수이며, 제약조건이 만족되면서 목적함수 $P(\mathbf{x})$를 감소시키면 원래의 목적함수인 연료비에 대한 함수를 최소화시키는 해를 얻을 수 있다.

3. 병렬유전알고리즘의 응용

최근 병렬처리용 컴퓨터 하드웨어와 소프트웨어 기술이 발달하여 이를 이용한 계산시간의 단축과 이에 소요되는 비용의 감소로 병렬계산 응용 연구가 다방면에서 시작되고 있으며, 전력계통 분야에서도 대규모 시스템 해석에 새로운 기술을 적용하기 위한 연구가 활발히 진행되고 있다.

3.1 유전알고리즘(Genetic Algorithms)

유전알고리즘(Genetic Algorithms : GA)은 자연선택과 자연유전학의 원리를 근거한 최적점탐색 알고리즘으로서, GA는 최적화문제의 목적함수로부터 적합도를 구해 적합도 값에 따라 다음 세대에 생존과 소멸을 결정하는 것으로, 종래의 탐색 알고리즘과는 다른 점이 있다[8]. 첫째, 탐색공간에서 해집단을 이용하여 병렬로 탐색한다. 그러므로 전역적 최적해를 향해 수렴할 가능성이 훨씬 높다. 둘째, 결정적 요소(목적함수) 만을 정보로 사용하고, 도함수나 다른 정보를 필요로 하지 않는다. 따라서 자연계의 복잡한 비선형함수에 적용하기가 수월하다. 셋째, 탐색을 위하여 확률적인 규칙을 사용한다. 탐색공간에서 단순히 랜덤 하게 진행하는 것이 아니라 좀더 빠르게 최적점을 찾기 위하여 결정적 요소를 이용하여 효율적으로 확률적인 선택을 한다. 넷째, 최적화문제의 변수들의 집합을 유한길이의 2진 스트링으로 코딩한다. 더욱 정확한 해를 요구한다면 스트링의 길이는 매우 길 것이다. 긴 스트링에 대한 코딩절차는 많은 계산시간을 필요로 하고 수렴 정도를 감소시킬 것이다.

본 연구에서는 2장의 최적부하 배분문제에 유전알고리즘 응용시, 스트링의 구성을 2진 스트링으로 코딩하는 대신 실수로 사용하였다. 즉, 각각의 발전기출력의 상하한치 범위 내에서 난수를 발생시켜 발전 가능한 발전기의 수만큼 스트링의 길이를 구성하였다. 결국, 식 (1)의 최적부하 배분문제의 목적함수를 n개의 변수로 구성된 $f(x_{i1}, x_{i2}, \cdots, x_{in})$와 같이 나타낼 수 있다. 변수들은 다음의 같은 스트링 $\mathbf{X}_i = (x_{i1}, x_{i2}, \cdots, x_{in}) \in R^n$로 표현할 수 있으며, R^n은 n차원 탐색공간이다. 스트링 $\mathbf{X}_i$를 하나의 개체(individual)라고 하며, 개체들의 집합을 개체군(population)이라고 한다.

다음은 최적부하 배분문제에 대한 유전알고리즘 응용 절차를 나타내었다.

***step 1* 초기화**

개체들의 유전자는 발전기 수만큼 각각의 발전기의 유효전력 출력 상하한치 범위 내에서 랜덤하게 발생시킨다. 그림 1은 유전알고리즘의 개체군 구성도이며, x_{ii}는 i번째 개체의 i번째 유전자를 의미한다.

$x_{11}, x_{12}, \cdots, x_{1n}$
$x_{21}, x_{22}, \cdots, x_{2n}$
$x_{31}, x_{32}, \cdots, x_{3n}$
·····
·····
$x_{i1}, x_{i2}, \cdots, x_{in}$

그림 1 개체군의 구성도

Fig. 1 Configuration of Population

***step 2* 적합도 평가**

식 (5)를 이용하여 적합도를 평가한다. 식 (3)은 각 발전기 출력의 상하한치 범위 내에서 개체의 유전자를 구성하는데 사용되므로 부등식 제약조건에서 고려할 필요가 없다. 식 (4)의 μ는 페널티 계수로서 세대가 변화됨에 따라 ∞로 증가하고 목적함수 $P(\mathbf{x})$를 감소시키면, 제약에 대한 강도 또한 증가하여 결국 제약조건을 만족하면서 원래의 목적함수인 연료비에 대한 함수를 최소화시키는 해를 얻을 수 있다. 식 (4)의 페널티 함수 형태로 변형된 최적 부하배분 문제는 식 (5)와 같이 역을 취하여 목적함수 최소화문제를 적합도 최대화문제로 변형시킬 수 있다.

$$\text{Fitness}(A) = \frac{1}{P(\mathbf{x})} \tag{5}$$

***step 3* 수렴 판정**

a) 만약 최대 세대이면 멈춘다.

b) 그렇지 않으면 ***step 4***로 간다.

***step 4* 복제(reproduction)**

복제란 ***step 2* 적합도 평가**에서 얻은 적합도에 비례하여 개체를 생존시키는 작용자로서, 비선형이고 불연속인 양의 값을 갖는 함수를 적합도 함수로 정의할 수 있다. 정규화된 적합도는 다음 세대에 생존할 확률로서, 적합도가 더 큰 개체는 다음 세대에 더 많은 자손을 발생시킬 가능성이 크다. 즉, 우성개체는 다음세대로 넘어가고 열성인 개체는 소멸된다. 다음세대로 넘어갈 개체를 선택하기 위해서 적합도에 의해 스롯의 크기가 정해지는 룰렛 휠을 사용하였다.

***step 5* 교차(crossover)**

새롭게 복제된 개체들 중에서 교차확률에 따라 교차할 개체를 선정한다. 새로운 개체군을 이루는 각각의 개체들에는 재결합 작용자인 교차 작용자가 적용된다. 본 연구에서는 whole arithmetical crossover를 사용하였다.

이 작용자는 식 (6)과 같이 두 개의 개체가 선정되었을 때 식 (7)과 같이 선형조합(linear combination)으로 정의하는 것으로서, a는 0과 1 사이의 난수이다. 여기서, t는 현세대, $\mathbf{X}_i^t$와 $\mathbf{X}_j^t$는 현세대 t의 i와 j 개체, 즉 부모이고, $\mathbf{X}_i^{t+1}$과 $\mathbf{X}_j^{t+1}$은 t+1 세대로 넘어갈 자손을 의미한다.

$$\begin{aligned}\mathbf{X}_i^t &= (x_{i1}, x_{i2}, \cdots, x_{in}) \\ \mathbf{X}_j^t &= (x_{j1}, x_{j2}, \cdots, x_{jn})\end{aligned} \tag{6}$$

$$\begin{aligned}\mathbf{X}_i^{t+1} &= a\cdot\mathbf{X}_j^t + (1-a)\cdot\mathbf{X}_i^t \\ \mathbf{X}_j^{t+1} &= a\cdot\mathbf{X}_i^t + (1-a)\cdot\mathbf{Y}_j^t\end{aligned} \tag{7}$$

***step 6* 돌연변이(mutation)**

교차를 행한 후 개체의 스트링에서 각각의 유전자는 돌연변이 확률만큼 랜덤한 변화를 한다. 따라서 돌연변이 연산자는 국지적 최적해로 수렴하는 것을 방지하는 작용을 한다. 그러나 돌연변이는 세대가 변하면서 환경에 대한 적응성과는 무관하게 랜덤 하게 발생하므로 돌연변이 확률이 너무 크면(50% 이상) 일반적으로 랜덤 탐색의 경향을 나타낸다. 본 연구에서는 non-uniform mutation을 사용하였다. 유전자 x_{ii}가 돌연변이 대상으로 선정되면 non-uniform 돌연변이는 유전자 값을 식 (8)의 x_{ii}'와 같이 변형시킨다.

$$x_{ii}' = \begin{cases} x_{ii} + \triangle(t, y), & y = x_{ii}^{max} - x_{ii} \\ & : \text{랜덤 digit가 0인 경우} \\ x_{ii} - \triangle(t, y), & y = x_{ii} - x_{ii}^{min} \\ & : \text{랜덤 digit가 1인 경우} \end{cases} \tag{8}$$

$$\triangle(t,y) = y\cdot r\cdot(1 - t/T)^b$$

여기서, r은 0에서 1사이 난수, t는 현세대, T는 최대세대수, b는 non-uniform의 정도를 결정하는 수이다. x_{ii}^{max}와 x_{ii}^{min}은 돌연변이 대상으로 선정된 변수의 상하한치, 즉 최적부하 배분문제의 경우에서는 발전기출력의 상하한치이다. 또한, 랜덤 digit에 따라 구분되는 것은 선택된 유전자 즉, 변수가 그 점과 상한치 또는 하한치의 범위 내에서 변화되도록 하는 방향을 나타내는 것이다. t/T는 초기세대에서는 탐색공간을 균일하게 탐색하고 t가 증가함에 따라 국지적으로 탐색하는 것을 의미한다. ***step 2***로 간다.

3.2 유전알고리즘에 병렬처리 응용

병렬처리란 두 개 이상의 컴퓨터 프로세서가 내부처리 통신망으로 연결되어 단일 문제를 두 개 이상의 소규모 문제로 분할하여 병렬상태로 처리함으로써 계산속도를 향상시키기 위한 전산기술이다. 따라서, 유전알고리즘에 병렬처리를 응용하는 목적은 유전알고리즘의 결점인 개체의 수에 따라 계산량이 많아지는 것을 병렬로 처리함으로써 계산속도를 향상시키는 것이

다.

유전알고리즘의 최적해를 탐색하는데 있어서 정식화된 문제의 규모에 따라 개체의 수와 세대의 수를 크게 설정할 경우 매우 많은 시간이 걸린다. 이것은 유전알고리즘의 고유한 특성상 한 점에서 해를 찾아가는 것이 아니라 여러 개의 점에서 동시에 해를 찾아나가기 때문이다. 여러 개의 점에서 동시에 해를 탐색하는 즉, 병렬로 전개되는 특성으로부터 유전알고리즘에 병렬처리를 응용할 수 있으며, 이것을 병렬유전알고리즘(Parallel Genetic Algorithms : PGA)이라 한다. 결국, 유전알고리즘은 순차적인 처리를 목적으로 알고리즘의 간단한 조정에 의하여 병렬처리에 응용하기에 적합한 알고리즘이라는 것을 알 수 있다[7].

유전알고리즘에서 해를 탐색해 나가는 과정에서 초기화, 적합도 평가, 복제, 돌연변이 등과 같은 작용은 개체들 사이에 데이터 교환이 이루어지지 않으므로, 유전알고리즘의 각각의 개체가 할당되는 프로세서들 사이에 데이터 통신을 하지 않는 병렬처리 아키텍처로 구성한다면 해를 탐색해 나가는 실행시간을 단축시킬 수 있다.

본 연구에서는 M. J. Flynn의 병렬처리 시스템 아키텍처 분류의 하나인 SIMD(single instruction stream, multiple data stream) 시스템을 사용하였다. SIMD 시스템이란 모든 프로세서가 동일한 명령어를 동시에 실행하여, 각 프로세서에서 주어진 데이터를 처리하는 시스템이며, 위에서 언급한 유전알고리즘의 병렬처리 응용 특성에 적합하다는 것을 알 수 있다.

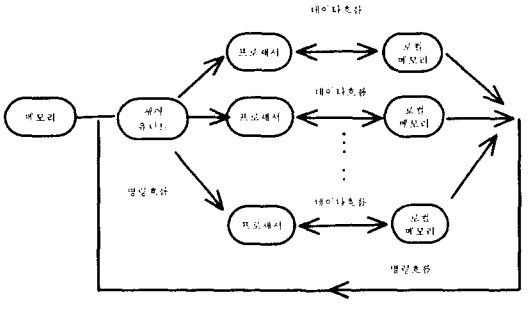

그림 2 SIMD의 개념도
Fig. 2 Concept of SIMD

그림 2는 SIMD에 대한 개념도로서, 하나의 명령흐름을 수행하지만 많은 연산처리 유니트를 포함하며, 각 프로세서는 자신의 데이터를 메모리로부터 가져와 처리할 수 있다. 그림 2에서 알 수 있듯이 모든 프로세서는 중앙유니트에서 나오는 하나의 명령흐름의 제어하에서 동작한다. 예를들어 그림 2에서 유전알고리즘의 적합도 계산에 대하여 병렬처리를 설명하면, 각각의 프로세서의 로컬 메모리에 주 메모리로부터 한 개의 개체씩 할당시킨다. 그리고, 제어 유니트로부터 각 프로세서에 적합도에 평가에 대한 명령을 전달하면 프로세서들은 다른 데이터 값을 가지고 동시에 적합도를 계산한다. 이렇게 계산된 결과는 주 메모리에 전달된다. 그리고 다른 연산을 위하여 계속 실행하게 된다. 그러나, 교차의 경우 개체들 사이에 종속적으로 진행되는데 이것은 교차시킬 대상의 개체들을 선택한 후 각 프로세서에 할당시키면 병렬처리가 가능하다. 교차와 돌연변이는 매 세대마다 교차와 돌연변이 횟수가 각각 달라지게 된다. 이것은 예를들어, 10개의 프로세서 중 5개가 교차 또는 돌연변이에 할당되면, 나머지 5개의 프로세서는 다음 과정으로 넘어가게 하면 해결할 수 있다.

3.3 병렬유전알고리즘(Parallel Genetic Algorithms) 절차

유전알고리즘은 개체의 해집합이 병렬로 전개되는 고유의 특성으로부터, 순차적인 처리를 목적으로 알고리즘의 간단한 조정에 의하여 병렬처리에 응용하기에 적합한 알고리즘이다. 유전알고리즘에 있어서 해를 탐색하기 위한 몇몇 과정에서 초기화, 적합도 평가, 복제, 돌연변이 등은 개체들 사이에 데이터를 교환하지 않고 독립적으로 진행된다. 또한, 교차는 두 개체 사이에 종속적으로 진행되기 때문에 교차시킬 개체를 선정한 후, 각 프로세서에 할당시켜 병렬처리를 할 수 있다. 따라서, 유전알고리즘에 병렬처리를 응용할 경우, 병렬처리 구조의 하나인 SIMD(single instruction stream, multiple data stream) 시스템에 구축하는 것이 적합하다.

SIMD를 기초로 병렬유전알고리즘 응용에 있어서, 프로세서의 수와 같게 개체군(population)을 여러 개의 작은 개체군(sub-population)으로 분할하여 병렬화를 하는 분배 메카니즘(division mechanism)을 이용하여 다음과 같은 방법으로 기존의 유전 알고리즘에 적용한다.

***step 1* 초기화**

SIMD 시스템을 응용하기 위하여 개체군을 프로세서의 수와 같게 여러 개의 작은 개체군으로 분할한 후, 각각의 작은 개체군의 개체들의 유전자는 각각의 발전기의 유효전력출력 상하한치 범위 내에서 랜덤하게 발생시킨다. 그리고, 각각의 작은 개체군을 각각의 프로세서에 할당시킨다.

***step 2* 적합도 평가**

SIMD에 근거한 병렬처리를 이용하여 각각의 프로세서에 할당된 작은 개체군의 개체들에 대한 페널티 함수를 계산하고 적합도를 평가한다. ***페널티 함수 및 적합도 평가는 3.1절 유전알고리즘의 step 2***와 같다.

***step 3* 수렴 판정**

a) 만약 최대 세대이면 멈춘다.
b) 그렇지 않으면 ***step 4***로 간다.

***step 4* 개체의 스트링 조작**

복제와 돌연변이는 개체들 사이에 독립적으로 진행되므로 SIMD를 기반으로 병렬처리에 의하여 향상시킨다. 또한, 교차는 두 개체 사이에 종속적으로 진행되기 때문에 교차시킬 개체를 선정한 후, 각 프로세서에 할당시켜 병렬처리를 한다. ***복제, 교차 및 돌연변이는 3.1절의 step 4, 5 및 6과 같다. step 2***로 간다.

4. 사례연구

병렬유전알고리즘을 최적 부하배분 문제에 적용하여 효율성을 입증하기 위해 38기 계통에 시험하였다[9]. 최적부하 배분

문제는 C 언어로 코딩하였고, PC-IBM pentium II에서 병렬처리에 의한 계산소요시간을 구하기 위해 SIMSCRIPT 언어를 이용하여 구현하였다.

본 연구에서 최적 부하배분 문제에 사용한 병렬유전알고리즘을 위한 파라메타 값들은 다음과 같다.

- 스트링의 표현 : 실수
- 개체군의 크기 : 2000개
- 최대 세대수 : 2000세대
- 교차확률 : 0.9
- 돌연변이확률 : 0.09
- 시험계통의 총 부하 : 8250MW
- 전체계통의 발전기 수 : 38기
- 전력수급에 관한 등식제약조건 허용오차 : 0.001
- 프로세서의 수 : 1, 2, 4, 5, 10, 20개

표 1에는 각 발전기들의 연료비용 계수와 발전출력의 상하한치를 나타내었으며, SQP (Sequential Quadratic Programming)에 의한 결과와 병렬유전알고리즘에 의한 결과를 비교하였다. 또한, 송전손실은 비선형 등식제약조건으로서 목적함수에 포함시켜야하나, 본 연구에서는 최적경제 부하배분 문제에 병렬유전알고리즘 적용의 가능성과 그 효과를 분석하는데 초점을 맞추었고, 실 계통에 온-라인 적용 및 실제 하드웨어 구현을 위해서는 앞으로 고려할 필요가 있다. 결과로부터 알 수 있는 것은 연료비용이 낮은 발전기에서는 발전출력이 상한치에 가깝고, 연료비용이 높은 발전기는 하한치 가까이 즉, 출력이 적다는 것은 알 수 있다. SQP 방법의 총 발전비용은 $12,159,477.29인 반면에, 병렬유전알고리즘(PGA)에 의한 총 발전비용은 $12,159,455.00임을 알 수 있다. 탐색에 의한 총 발전비용을 비교하여 보면 전역적 탐색기법인 병렬유전알고리즘에 의한 결과가 고전적인 방법인 SQP로부터 얻은 결과 보다 더 작다는 것을 알 수 있다.

계산시간의 단축은 병렬 컴퓨터를 사용하고자 하는 주된 목적이므로 병렬 알고리즘을 평가하는 가장 중요한 척도는 계산소요시간(running time)이다. 계산소요시간은 컴퓨터에서 알고리즘에 의해 문제를 해결하는데 작업의 시작순간부터 끝나는 순간까지 소요되는 시간을 의미한다. 본 연구에서는 병렬유전알고리즘을 평가하기 위해서 동일한 문제에 대해서 순차처리 알고리즘을 사용한 경우 즉, 한 개의 프로세서를 사용했을 때의 계산소요시간과 여러 개의 프로세서로 병렬처리 알고리즘을 사용한 경우의 계산소요시간을 비교하는 방법을 사용하였다. 이것은 식 (9)와 식 (10)과 같이 속도 향상률과 효율로 평가하였다.

$$\text{속도향상률}(S_p) = \frac{\text{가장빠른 순차처리 소요시간 } (T^*)}{\text{병렬처리 소요시간 } (T_P)} \tag{9}$$

$$\text{효율}(E_P) = \frac{\text{속도 향상률 } (S_P)}{\text{프로세서수 } (P)} \times 100 \tag{10}$$

표 2는 병렬유전알고리즘을 이용하여 프로세서의 수가 1, 2, 4, 5, 10 그리고 20개 일 때 실행한 계산시간, 속도 향상률과 효율을 나타내었다. 표 2로부터 프로세서 수의 증가에 따라 계산시간이 감소되고, 식 (9)의 속도 향상률과 식 (10)의 효율이 향상됨을 알 수 있다. 각 프로세서의 수와 계산소요시간과의 관계는 그림 3에 나타내었다. 또한, 프로세서의 수에 따른 속도 향상률 및 효율과의 관계는 그림 4와 5에 나타내었다. 표 2 및 그림 3과 4에서 프로세서 수의 증가에 따라 계산소요시간이 프로세서의 수에 거의 비례하여 감소됨을 알 수 있고, 속도 향상률 역시, 향상됨을 알 수 있다. 또한, 효율에 있어서, 병렬처리를 하였을 때 프로세서 수의 증가에 따라 96% 이상이 된다는 것을 알 수 있다. 이것으로부터 유전알고리즘은 순차적인 처리를 목적으로 알고리즘의 간단한 조정에 의하여 병렬처리에 응용하기에 적합한 알고리즘이라는 것을 알 수 있다. 따라서, 유전알고리즘을 전력계통의 제반문제에 적용시, 유전알고리즘의 특성상 계산소요시간이 많이 걸리는 단점을 병렬처리를 응용하여 해결할 수 있다. 특히, 실제 하드웨어를 구현할 경우 실 계통에 온-라인 적용도 가능하다는 것을 알 수 있다.

표 1 38기 시스템에 대한 발전 데이터 및 최적배분 결과
Table 1 Generator data and results of 38 units system

Unit No.	a_i [$/MW]	b_i [$/MW]	c_i [$/MW]	P_i^{min} [MW]	P_i^{max} [MW]	P_i[MW] SQP	P_i[MW] PGA
1	64782.0	796.6	0.3123	220.0	550.0	550.00	550.00
2	64782.0	796.6	0.3123	220.0	550.0	550.00	550.00
3	64670.0	795.5	0.3127	200.0	500.0	500.00	500.00
4	64670.0	795.5	0.3127	200.0	500.0	500.00	499.98
5	64670.0	795.5	0.3127	200.0	500.0	500.00	500.00
6	64670.0	795.5	0.3127	200.0	500.0	500.00	499.99
7	64670.0	795.5	0.3127	200.0	500.0	500.00	500.00
8	64670.0	795.5	0.3127	200.0	500.0	500.00	500.00
9	172832.0	915.7	0.7075	114.0	500.0	383.05	377.67
10	172832.0	915.7	0.7075	114.0	500.0	383.05	377.52
11	176003.0	884.2	0.7515	114.0	500.0	382.78	376.07
12	173028.0	884.2	0.7083	114.0	500.0	406.02	403.01
13	91340.0	1250.1	0.4211	110.0	500.0	240.77	246.50
14	63440.0	1298.6	0.5145	90.0	365.0	148.00	152.25
15	65468.0	1298.6	0.5691	82.0	365.0	132.82	135.70
16	72282.0	1290.8	0.5691	120.0	325.0	138.73	145.79
17	190928.0	238.1	2.5881	65.0	315.0	234.38	236.03
18	285372.0	1149.5	3.8734	65.0	315.0	65.00	65.01
19	271376.0	1269.1	3.6842	65.0	315.0	65.00	65.00
20	39197.0	696.1	0.4921	120.0	272.0	272.00	272.00
21	45576.0	660.2	0.5728	120.0	272.0	272.00	272.00
22	28770.0	803.2	0.3572	110.0	260.0	260.00	260.00
23	36902.0	818.2	0.9415	80.0	190.0	190.00	189.99
24	105510.0	33.5	52.1230	10.0	150.0	14.39	13.58
25	22233.0	805.4	1.1421	60.0	125.0	125.00	125.00
26	30953.0	707.1	2.0275	55.0	110.0	110.00	109.99
27	17044.0	833.6	3.0744	35.0	75.0	75.00	74.98
28	81079.0	2188.7	16.7650	20.0	70.0	20.00	20.00
29	124767.0	1024.4	26.3550	20.0	70.0	20.00	20.01
30	121915.0	837.1	30.5750	20.0	70.0	20.00	20.00
31	120780.0	1305.2	25.0980	20.0	70.0	20.00	20.00
32	104441.0	716.6	33.7220	20.0	60.0	20.00	20.00
33	83224.0	1633.9	23.9150	25.0	60.0	25.00	25.00
34	111281.0	969.6	32.5620	18.0	60.0	18.00	18.00
35	64142.0	2625.8	18.3620	8.0	60.0	8.00	8.00
36	103519.0	1633.9	23.9150	25.0	60.0	25.00	25.00
37	13547.0	694.7	8.4820	20.0	38.0	38.00	37.99
38	13518.0	655.9	9.6930	20.0	38.0	38.00	37.96
Fuel Cost[$]	SQP						12,159,477.29
	PGA						12,159,455.00

본 연구에서는 최적부하 배분문제에 병렬유전알고리즘의 적용 가능성과 그 효과를 분석하였다. 실제로 하드웨어를 구현하

는 대신, 병렬처리의 계산소요시간 계산에 유용한 SIMSCRIPT 언어를 사용하였다. 하드웨어를 이용한 SIMD 시스템을 구현하는데 있어 상호연결 네트워크(interconnection network) 통신 시스템과 공유기억장치(shared memory) 통신 시스템에 적용할 수 있다.

상호연결 네트워크(interconnection network) SIMD 컴퓨터는 각 프로세서가 독립된 메모리를 가지며 다른 데이터에 대해 같은 동작을 동시에 수행할 수 있으며 대표적인 방식이 프로세서 어레이 방식이다. 그림 2에 나타낸 것이 상호연결 네트워크 방식인 프로세서 어레이 방식이다. 공유기억장치(shared-memory) SIMD 컴퓨터는 N개의 프로세서에서 한 개의 메모리를 사용하는 방법으로 공용메모리를 갖는다. 예를들어 프로세서 i가 프로세서 j에 수치를 보낼 때 프로세서 i가 프로세서 j에 알려진 위치에 있는 공유기억장치에 수치를 쓴다. 그리고 프로세서 j가 그 위치로부터 수치를 읽는 방법이며, 그림 2에서 로컬메모리가 제외된다. 두 방식 모두 본 연구에서 적용한 유전알고리즘에 이용할 수 있으며, 이것은 유전알고리즘의 개체군을 프로세서의 수와 같게 여러 개의 작은 개체군으로 분할한 후, 각각의 작은 개체군을 각 프로세서에 할당시켜 같은 명령을 수행해 나가는 것은 같으나, 메모리의 방식은 특성에 따라 서로 다르게 한다.

결국, 본 연구에서 C 언어로 코딩된 최적부하 배분문제를 병렬처리를 위하여 SIMSCRIPT 언어를 사용하여 구현한 SIMD 시스템은 공유기억장치 시스템과 상호연결 네트워크 시스템에 모두 적용할 수 있으며, 향후 실제로 하드웨어를 구현하여 병렬유전알고리즘의 적용 가능성과 그 효과를 비교 및 분석할 필요가 있다.

표 2 계산시간, 속도 향상률 및 효율

Table 2 Running time, Speedup and Efficiency

No. of processor	Running time[sec]	Speedup eq. (9)	Efficiency[%] eq. (10)
1	956.146	1	100
2	495.078	1.931	96.55
4	247.544	3.863	96.58
5	198.037	4.828	96.56
10	99.023	9.656	96.56
20	49.517	19.309	96.55

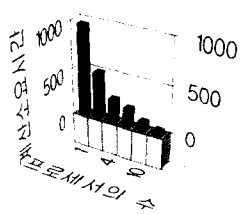

그림 3 프로세서의 수에 따른 계산소요시간

Fig. 3 The running time according to the number of each processor

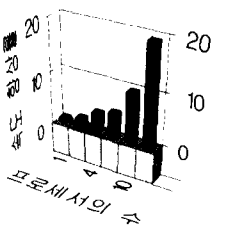

그림 4 프로세서의 수에 따른 속도 향상률

Fig. 4 The Speedup according to the number of each processor

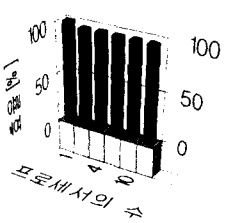

그림 5 프로세서의 수에 따른 효율

Fig. 5 The Efficiency according to the number of each processor

5. 결 론

본 연구에서는 병렬유전알고리즘을 이용한 대규모 전력계통의 최적 부하배분 방안을 제안하였으며, C 언어로 코딩된 최적부하배분 문제를 병렬처리를 위하여 SIMSCRIPT 언어에 의하여 성공적으로 구현하였다.

유전알고리즘은 개체군의 개체들(Multiple Data)이 병렬로 진화하면서 같은 동작(Single Instruction)을 수행한다. 특히, 초기화, 적합도 평가, 복제, 돌연변이 등은 개체들 사이에 데이터를 교환하지 않고 독립적으로 진행되므로, 병렬처리 구조의 하나인 SIMD(single instruction stream, multiple data stream) 시스템과 같은 병렬기기에 구축하는 것이 적합하다는 것을 알 수 있었다. 또한, 교차는 두 개체 사이에 종속적으로 진행되기 때문에 교차시킬 개체를 선정한 후, 각 프로세서에 할당시켜 병렬처리를 할 수 있었다.

결국, 병렬유전알고리즘을 이용하여 전력계통에 있어서 최적 경제부하배분을 위한 계산소요시간을 감소할 수 있으며, 실제 하드웨어를 구현할 경우 실 계통에 온-라인 적용도 가능하다는 것을 알 수 있다.

또한, 최적부하배분을 위한 병렬유전알고리즘 응용의 효용성을 입증하기 위하여 38기 계통에 적용하였으며, 전통적인 지역탐색 방법인 SQP와 비교하여 병렬유전알고리즘이 전역적 최적

해를 찾는데 있어 더 큰 가능성을 가지고 있음을 알 수 있다.

향후 연구로는 병렬유전알고리즘을 이용하여 송전선로의 용량한계를 고려한 장기발전계획을 할 필요가 있다고 사료된다.

감사의 글

이 논문은 1997년도 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음.

Reference

[1] A. J. Wood and B. F. Wollenberg, Power Generation, Operation and Control, John Wiley & Sons, New York, 1984

[2] S. J. Wang, et at., "Short-Term Generation Scheduling with Transmission and Environmental Constraints using an Augmented Lagrangian Relaxation", IEEE Trans. on Power Systems, vol. 10, no. 3, pp1294-1301, 1995

[3] J. H. Park, et al., "Economic load dispatch for piecewise quadratic cost function using Hopfield neural network", IEEE Trans. on Power Systems, vol. 8, no. 3, pp1030-1038, 1993

[4] C. T. Su, G. J. Chiou, "A Fast-Computation Hopfield Method to Economic Dispatch of Power Systems", IEEE Trans. on Power Systems, vol. 12, no. 4, pp1759-1764, November, 1997

[5] G. B. Sheble, et al., "Refined Genetic Algorithm - Economic Dispatch Example", IEEE Trans. on Power Systems, vol. 10, no. 1, pp117-124, February, 1995

[6] Kyu-Ho Kim, Seok-Ku You, "Optimal VAR Dispatching considering Contingency by Genetic Algorithms", Proceeding of the International Conference on Electrical Engineering, vol. 2, pp997-1001, August, 1996

[7] J. Stender, "Parallel Genetic Algorithms: Theory and Applications", 1993

[8] Z. Michalewicz, "Genetic algorithms + Data structures = Evolution Programs", Second Edition, Springer Verlag, 1992

[9] H. T. Yang, et al., "A Genetic Algorithm Approach to Solving Implementation on the Transputer Networks", IEEE Trans. on Power Apparatus and Systems, vol. 12, No. 2, pp661-668, May, 1997

저 자 소 개

김 태 균 (金 泰 均)

1964년 1월 12일생. 1986년 한양대 공대 전기공학과 졸업. 1993년 동 대학원 전기공학과 졸업(공박). 현재 한국전력공사 전력연구원 전력계통연구실 선임연구원

Tel : (042) 865-5831

김 규 호 (金 圭 浩)

1965년 3월 8일생. 1988년 한양대 공대 전기공학과 졸업. 1992년 동 대학원 전기공학과 졸업(석사). 1996년 동 대학원 전기공학과 졸업(공박). 현재 안산공과대학 전기과 전임강사

Tel : (0345) 490-6053

E-mail : kyuho@nail.hanyong.ac.kr

유 석 구 (劉 錫 九)

1938년 10월 31일생. 1961년 한양대 공대 전기공학과 졸업. 1980년 동 대학원 전기공학과 졸업(공박). 현재 한양대 공대 전기공학과 교수. 1996년 당학회 회장

Tel : (0345) 400-5161