병렬 적응 진화알고리즘을 이용한 발전기 기동정지계획에 관한 연구

論 文
55A-9-3

A Parallel Adaptive Evolutionary Algorithm for Thermal Unit Commitment

金 亨 洙† · 趙 德 桓* · 文 景 俊** · 李 和 錫*** · 朴 俊 灝§ · 黃 琪 鉉§§
(Hyung-Su Kim · Duck-hwan Cho · Kyeong-Jun Mun · Hwa-Seok Lee · June-Ho Park · Gi-Hyun Hwang)

Abstract - This paper is presented by the application of parallel adaptive evolutionary algorithm(PAEA) to search an optimal solution of a thermal unit commitment problem. The adaptive evolutionary algorithm(AEA) takes the merits of both a genetic algorithm(GA) and an evolution strategy(ES) in an adaptive manner to use the global search capability of GA and the local search capability of ES. To reduce the execution time of AEA, the developed algorithm is implemented on an parallel computer which is composed of 16 processors. To handle the constraints efficiently and to apply to parallel adaptive evolutionary algorithm(PAEA), the states of thermal unit are represented by means of real-valued strings that display continuous terms of *on/off state of generating units and are involved in their minimum up and down time constraints*. And the violation of other constraints are handled by repairing operator. The procedure is applied to the 10 ~ 100 thermal unit systems, and the results show capabilities of the PAEA.

Key Words : Parallel Adaptive Evolutionary Algorithm, Unit Commitment, Genetic Algorithm, Evolution Strategy, Parallel Computer

1. 서 론

우리나라는 경제성장에 따른 급격한 전력수요의 증가에 대응하기위해 단계적으로 발전소를 건설해 왔다. 이에 따라 전력계통의 규모도 점점 대형화되어 경제적이고 효율적인 전력계통 운용의 필요성도 더욱 증가하고 있다. 전력계통 운용은 계획, 감시 및 제어 등을 포함하는 복합적인 분야로서 각 분야에 대한 종합적인 연구가 필요하며, 특히 발전기 기동정지계획[1]은 전통적으로 매우 중요한 역할을 담당해왔으며 현재까지도 많은 연구가 진행되고 있다. 최근에는 전력시장의 변화에 따라 새로운 시장 환경을 고려하여 발전기 기동정지계획을 수립하는 방법[2]도 제시되고 있다. 종래에는 단일 또는 독점적인 발전사업자에 의해 계통이 운영되었으므로 발전기 기동정지계획의 목적은 주로 발전기의 운전비용을 최소화하는 것으로 정의되었다. 그러나 최근의 전력시장은 각 지역별 시장상황 및 여러 발전사업자간의 경쟁요소의 도입 등으로 인하여 종래와는 다르게 정의되고 있다. 수직적 전력산업 구조에서는 주로 단일의 계통 운영자에 의해서만 발전기 기동정지계획이 수립되었으나 이제는 계통운영자뿐만 아니라 전력시장에 참여하는 각 발전사업자들도 자신에게 유리한 참여를 위해 독자적인 발전기 기동정지계획을 수립해야 할 필요성이 생겼다. 따라서 앞으로 발전기 기동정지계획의 중요성 또한 더욱 커질 것으로 생각된다.

본 논문에서는 비용 최소화 관점에서의 발전기 기동정지계획의 방법을 제안한다. 비록 각 시장상황에 따라 조금씩 달라지지만 대부분의 계통 운영자는 여러 제약과 계통의 안전도 등을 고려하면서 비용최소화 관점에서 계통을 운영하기 때문에 제안한 방법은 여전히 유효하다고 할 수 있겠다. 수학적으로 볼때 발전기 기동정지계획 문제는 많은 변수를 포함하는 복잡한 조합 최적화 문제로 볼 수 있다. 지금까지 발전기 기동정지계획 문제를 해결하기 위해 많은 방법들이 제안되었는데, 수치적인 방법으로는 우선순위법[3], 동적 계획법[4], Branch and Bound[5], 혼합 정수법[6], 라그랑지 미정계수법[7] 등이 있다. 우선 순위법은 발전기의 우선 순위를 정하여 그 순서에 따라 발전기의 기동 및 정지를 결정하는 것으로 구현이 용이하나 근사적인 해를 구한다는 단점이 있다. 동적 계획법은 이론적으로 최적을 해를 구할 수 있지만 실제로 기억용량과 계산시간 등으로 인해 대형 계통에 대해 적용하는 불가능하다. 라그랑지 미정계수법은 승수를 이용하여 제약조건을 주 문제의 목적함수에 결합시켜 듀얼(Dual) 문제를 구성하고 이를 통해 주(Primal) 문제의 해를 구하는 방법으로 지금까지 대형 시스템에 가장 효과적인 방법으로 알려져 있다. 그렇지만 때로는 해가 수렴하기 어렵고, 준최적해를 구한다는 문제점을 가지고 있다. 이와같이 수치적 방법들은 실제 운용상에 발생하는 다양한 제약 조건의 고려가 어렵고 전역 최적해를 구하기가 어려운 문제점이 있어, 최근에는 확률적인 최적화 방법인 유전알고리즘(genetic

† 교신저자, 正會員 : 釜山大 工大 電氣科 post-doc · 工博
E-mail : kimhsu@pusan.ac.kr
* 學生 會員 : 現代 重工業 硏究員
** 正 會 員 : 韓國原子力硏究所 Senior Researcher · 工博
*** 正 會 員 : 巨濟大學 電氣科 副敎授 · 工博
§ 正 會 員 : 釜山大 電子電氣情報컴퓨터工學部 敎授 · 工博
§§ 正 會 員 : 東西大 컴퓨터工學部 助敎授 · 工博
接受日字 : 2006年 5月 30日
最終完了 : 2006年 8月 10日

algorithm: GA)과 Simulated Annealing을 이용한 방법[8], 경험적인 최적화 방법인 Tabu 탐색법을 이용하는 방법[9] 등이 연구되고 있다. 이러한 방법은 실질적인 제약조건의 고려가 용이하여 기존의 수치적인 방법보다 더 나은 해를 구할 수 있는 반면에, 많은 계산량과 시간을 요구함으로서 실 계통에 적용하기가 어려운 단점이 있다. 최근에는 컴퓨터 기술의 발달에 힘입어 병렬 프로세서 및 병렬 알고리즘을 이용한 방법[10]이 제안되었는데, 많은 프로세서를 동시에 이용함으로서 전역 최적해를 찾을 가능성은 증가시키면서 최적해를 탐색하는데 소요되는 수행시간을 줄일 수 있는 방법으로 그 이용이 점점 증가하고 있다.

본 논문에서는 발전기 기동정지계획 문제를 해결하기 위하여 병렬 프로세서 및 병렬 적응 진화알고리즘(parallel adaptive evolutionary algorithm: PAEA)을 이용한 방법을 제안하였다. 적응 진화알고리즘은 유전알고리즘의 전역탐색과 진화전략(evolution strategy: ES)의 국부탐색을 결합한 방법으로서, 실수형 문제에 적용이 용이하지만 본 논문에서는 이산적 문제인 발전기 기동정지계획에도 적용이 가능하도록 하였다. 또한 적응진화알고리즘을 병렬화하였는데, 각 프로세서는 독립적으로 해집단을 생성, 실행하면서 일정 세대 후에는 인접 프로세서와의 해를 공유함으로서 전역 탐색 성능 및 계산속도를 향상시킬 수 있도록 하였다. 그밖에 교배와 돌연변이를 수행한 후, 제약조건을 위배한 스트링에 대해서는 수정연산자를 도입하여 제약조건을 만족하도록 스트링을 재구성함으로서 탐색영역내에서 보다 효과적인 탐색이 가능하도록 하였다.

제안한 방법의 유용성을 입증하기 위해 참고문헌[14]에 있는 10대~100대 발전기 계통에 적용하여 기존의 방법과 비교·검토하였다. 사례 계통에 적용한 결과, 본 논문에서 제안한 방법은 계산속도의 향상과 더불어 발전 비용면에서 우수한 해를 구함을 알 수 있었다.

2. 발전기 기동정지계획의 정식화

일간 또는 수일간의 발전기 기동정지계획을 수립하기 위해서는 대상발전기의 조합을 결정하여 총 발전비용을 최소화하는 것으로 목적함수를 설정할 수 있다. 발전기 비용은 연료비용, 기동비용, 정지비용 등으로 구성된다. 여기서 연료비용은 열 소비율과 연료가격에 의해 계산되며, 기동비용은 발전기가 정지한 시간에 대한 함수로 표현된다. 그리고 정지비용은 각 발전기에 대한 일정한 값으로 주어진다. 이때 만족해야 할 제약조건으로는 부하 제약조건, 발전량 제약조건, 발전 예비력 제약조건 및 최소발전기 기동/정지 시간 등이 있으며, 그 밖에 운전원 제약, 상시운전 및 정지제약, 연료량, 연료소비율 및 계통 안전도에 대한 조건이 포함될 수 있다. 본 논문에서 고려한 화력 발전기 기동정지계획 문제를 풀기 위한 목적함수 및 제약조건은 아래와 같다.

(1) 목적함수 : 목적함수는 발전기 연료비용함수($F_{i,t}$), 기동비용($SC_{i,t}$) 및 정지비용($SD_{i,t}$)으로 구성되어 있으며 식 (1)과 같다. 여기서, 기동비용은 발전기의 온도와 압력에 대해 시간함수 또는 상수로 표현할 수 있으며, 본 논문에서는 식 (2)와 같이 일정상수로 가정하였다. 이때 냉각기동비용은 가열기동비용보다 높게 책정하였다. 발전기의 정지비용은 식 (3)과 같이 일반적으로 상수로 표현한다.

$$Min\ F_T = Min \sum_{t=1}^{T}\sum_{i=1}^{N}(F_{i,t} + SC_{i,t} + SD_{i,t}) \tag{1}$$

여기서, $F_{i,t}$: t시간에서 발전기 i의 연료비용

$SC_{i,t}$: t시간에서 발전기 i의 기동비용

$SD_{i,t}$: t시간에서 발전기 i의 정지비용

T : 발전기 기동정지계획 시간[h]

N : 총 발전기 수

$F_{i,t} = a_i + b_i P_i^t + c_i (P_i^t)^2$

a_i, b_i, c_i : 발전기 i의 비용함수의 계수

$$sc_{i,t} = \begin{cases} \alpha_i & \text{발전기 } i\text{의정지시간이냉각시간보다작을때} \\ \beta_i & \text{이외의경우} \end{cases} \tag{2}$$

여기서, α_i : 발전기 i의 가열기동비용 계수

β_i : 발전기 i의 냉각기동비용 계수

$$SD_{i,t} = \delta_i \tag{3}$$

여기서, δ_i : 발전기 i의 정지비용상수

(2) 제약조건 : 본 논문에서 사용한 제약조건은 부하 제약조건(load demand), 발전량 제약조건(generation limit), 발전 예비력 제약조건(spinning reserve) 및 최소발전기 기동/정지 시간으로 구성되며 아래와 같다.

- 부하 제약조건

$$\sum_{i=1}^{N} P_i = P_d\ ,\quad i = 1, 2, 3, \ldots, T \tag{4}$$

여기서, P_i : 발전기 i의 발전량[MW]

P_d : 부하량[MW]

- 발전량 제약조건

$$P_i^{Min} \leqq P_i \leqq P_i^{Max} \tag{5}$$

여기서, P_i^{Min} : 발전기 i의 최소발전용량[MW]

P_i^{Max} : 발전기 i의 최대발전용량[MW]

- 발전 예비력 제약조건

$$\sum_{i=1}^{N} P_i^{Max} u_{t,i} \geqq P_d + R_t \tag{6}$$

여기서, R_t : 예비발전기 용량[MW]

$u_{t,i}$: t 시간에서의 발전기 i의 on/off 상태

- 최소발전기 기동/정지 시간

$$(T_{t-1,i}^{on} - MT_i)(u_{t-1,i} - u_{t,i}) \geqq 0 \tag{7}$$

$$(T_{t-1,i}^{off} - MDT_i)(u_{t,i} - u_{t-1,i}) \geqq 0$$

여기서, MT_i : 발전기 i의 최소발전기 기동 시간[h]

MDT_i : 발전기 i의 최소발전기 정지 시간[h]

$T_{t,i}^{on}$: t시간까지 발전기 i의 운전지속시간

$T_{t,i}^{off}$: t시간까지 발전기 i의 정지지속시간

$u_{t,i}$: t시간에서 발전기 i의 운전상태(0: 정지, 1: 운전)

화력발전기 기동정지계획 문제는 전력경제급전 문제를 수반하고 이를 해결하기 위해서 계산이 간단하고 빠른 시간내에 경제급전 문제를 풀 수 있는 λ-iteration 방법을 이용하였다. λ-*iteration* 방법을 이용하여 발전 출력량을 구하는 과정은 다음과 같다.

단계 1) 초기 λ를 임의로 설정한다.

단계 2) 최대, 최소 발전출력을 고려하여 계획된 발전기별 발전량을 구한다.

단계 3) 제약조건으로서 소비부하량과 발전량의 합이 일치하는지 확인한다.

단계 4) 허용오차 이내로 수렴할 때까지 λ값을 변화시키면서 위의 단계 2)~단계 3)를 반복한다.

3. 적응진화 알고리즘

본 논문에서는 해집단을 다음세대로 진화시킬 때, 유전알고리즘과 진화전략을 동시에 사용하고, 적합도에 따라 복제하는 과정에서 유전알고리즘과 진화전략이 적용될 해집단의 비율이 적응적으로 변경되는 적응진화 알고리즘[12]을 사용하였고, 그 흐름도는 그림 1과 같다. 적응진화 알고리즘의 과정은 초기해집단을 구성할 때 각 스트링에 대해서 태그변수 0(유전알고리즘의 개체) 또는 1(진화전략의 개체)을 임의로 대응시킨 후, 각 개체의 적합도를 계산하고 룰렛(roulette wheel)을 사용하여 적합도에 따라 복제한다. 복제된 개체는 태그변수에 따라 유전알고리즘의 해집단과 진화전략의 해집단으로 분리한 후, 유전알고리즘의 해집단에 대해 교배 및 돌연변이 연산을 수행하고 진화전략의 해집단에 대해서는 돌연변이 연산만을 수행한다. 이때 부모해집단에서 적합도가 가장 높은 개체는 유전알고리즘의 해집단과 진화전략의 해집단에 각각 하나씩 복제하도록 하는 엘리티즘을 사용하였다. 적응진화 알고리즘의 주요 과정은 개체의 구분(초기화), 평가 및 복제, 최소해집단 보장, 유전알고리즘과 진화전략의 연산, 엘리티즘으로 구성되어 있으며, 각각의 특성은 다음과 같다.

(1) 개체의 구분(초기화): 초기 해집단을 구성할 때 각 스트링에 대해서 태그변수 0 또는 1을 임의로 대응시킨다. 태그변수 0은 유전알고리즘을 적용할 개체이고 태그변수 1은 진화전략을 적용할 개체이다.

(2) 평가 및 복제: 각 스트링을 평가함수에 따라 적합도를 평가한다. 복제 방법은 적합도에 비례하여 복제하는 룰렛휠을 사용하였다. 복제 후 태그변수가 0인 개체들은 유전알고리즘의 교배 및 돌연변이 과정을 적용하여 자손을 생성하고 그 자손에는 태그변수 0을 대응시킨다. 그리고 태그변수가 1인 개체들은 진화전략을 적용하여 자손을 생성하고 그 자손에는 태그변수 1을 대응시킨다.

(3) 최소 해집단수의 보장: 세대가 진행됨에 따라 유전알고리즘 혹은 진화전략 중 어느 하나의 역할이 너무 커지는 것을 방지하기 위해 유전알고리즘과 진화전략의 해집단은 전체 해집단에 대해 일정비율이상의 개체를 대응시키도록 하였다.

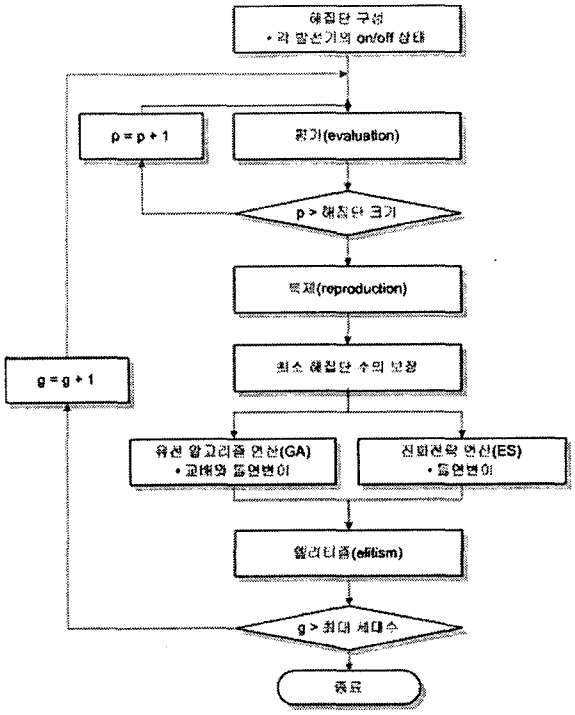


그림 1 적응진화 알고리즘의 흐름도

Fig. 1 Flowchart of adaptive evolutionary algorithm

(4) 교배(crossover) 및 돌연변이(mutation): 교배는 부모염색체의 유전자 정보를 서로 교환하여 새로운 정보를 가진 자손염색체를 생성하는 과정이며 돌연변이는 부모염색체 유전자의 형태변화로 새로운 유전정보를 발생하는 메커니즘이다. 본 논문에서는 실행시간이 단축되고 수학적 기법이 첨가될 수 있는 실변수형 유전알고리즘을 이용하였다. 또한 진화전략은 부모 해집단으로부터 자손 해집단을 생성한 후 부모는 모두 제거되고 자손 해집단을 다음세대의 부모 해집단으로 선택하는 방법을 사용하였다.

(5) 엘리티즘: 전체 해집단에서 적합도가 가장 높은 개체를 유전알고리즘의 해집단과 진화전략의 해집단에 각각 하나씩 복제하도록 하였다. 이때 유전알고리즘의 해집단에 대해서는 태그변수를 0으로, 진화전략의 해집단에 대해서는 태그변수를 1로 두었다.

4. 병렬 적응진화 알고리즘

4.1 병렬 컴퓨터 시스템

본 논문에서 사용한 병렬 컴퓨터 시스템은 메시지 전달 기반의 MIMD(Multiple Instruction stream Multiple Data stream) 시스템인 독일 Parsytec사의 CC/16(Cognitive Computer)이다. CC/16 시스템은 그림 2에서 보는 것처럼, crossbar switch로 연결된 16개의 노드로 구성되어 있으며, 각 노드는 PowerPC 604(100Mhz) 프로세서 및 로컬 메모리

를 가진다. 하나의 스위치는 4개의 프로세서 및 I/O를 위한 하드디스크(HD)와 연결되어 있으며, 각 프로세서와 스위치, 스위치와 스위치간의 통신은 데이터 전송속도가 75Mbyte/s인 Parsytec사의 HS(High Speed) link로 연결되어 있으며, 전체 시스템의 I/O 서비스를 제공하는 I/O서버가 4개 있다.

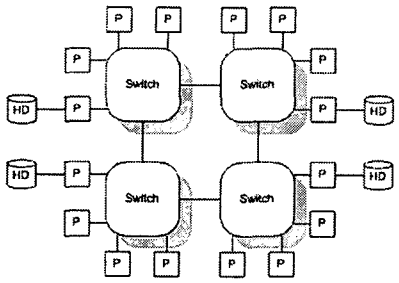


그림 2 병렬 컴퓨터 시스템 구조
Fig. 2 Structure of the parallel computer system

프로세서망은 데이터의 분배형태 및 결과를 반송하는 과정과 밀접한 관계에 있다. 프로세서망의 구성방식, 토폴로지(topology)는 특히 프로세서망의 규모 및 노드간의 통신량 등을 고려하여 적합한 형태로 구현해야 하며, 프로세서의 결합 차수 및 노드간의 거리로서 토폴로지의 특성을 나타낼 수 있다. 본 논문에서의 구현한 토폴로지는 하이퍼큐브 연결구조[13]이다. 이 구조는 적은 프로세서 개수를 가지는 병렬처리 시스템에서 차수를 줄이면서 노드간의 거리를 효과적으로 최소화하는 프로세서망 구조로서, 현재에는 많은 병렬처리 시스템이 하이퍼큐브 구조 및 이를 기초로 한 복합구조로 구성되어 있다. 2진 n-큐브 구조인 하이퍼큐브는 하이퍼큐브 내의 노드의 수와 각 노드들을 연결하는 통신 노드의 수를 결정하는 n차원 파라메터에 의해 그 특성이 달라지게 된다.

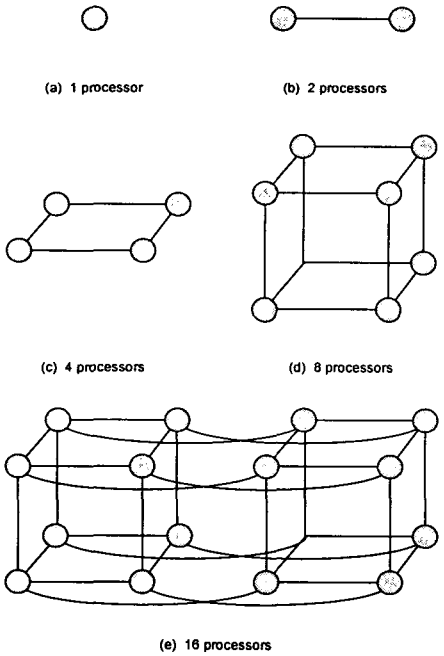


그림 3 하이퍼큐브 구조
Fig. 3 Hypercube structure

그림 3은 프로세서 개수의 증가에 따른 하이퍼큐브 구조를 나타낸 것이다. 전체 노드의 수는 차원의 수를 n이라 할 때 2n 개이고, 각 노드와 직접 연결된 노드의 수는 n개가 된다. 직접 연결되지 않은 노드로 데이터를 전송할 때에 중간노드를 경유하게 되는데, 가장 멀리 떨어져 있는 노드간의 거리는 n이 된다.

4.2 병렬 적응 진화알고리즘의 구성

병렬 처리란 다수의 프로세서들이 여러 개의 프로그램들 또는 프로그램의 분할된 부분들을 분담하여 동시에 처리하는 기술이며, 병렬처리를 위해서는 한 프로그램을 여러 개의 작은 부분들로 분할하는 것이 가능해야 하며, 분할된 부분들을 병렬로 처리하였을 때와 전체 프로그램을 순차적으로 처리하였을 때의 기능이 동일하여야 한다.

적응 진화알고리즘은 해집단을 운용하는 알고리즘으로써 수행과정중에 각 스트링마다 반복수행하는 절차가 내포되어 있으므로 병렬처리하기에 용이하다. 병렬처리는 같은 크기의 해집단에 대해 프로세서의 수가 많을수록 실행속도를 향상시킬 수가 있게 되며, 병렬처리 대상이 되는 알고리즘의 구성요소에 따라 최적해의 탐색성공률 등의 수행특성을 개선할 수 있다.

본 논문에서는 다중 장치 구조에 근거하여 적응 진화알고리즘의 병렬화 모델을 구성하였으며, 그림 4는 병렬화 모델의 구성도를 나타낸 것이다. 다중 장치 구조의 병렬처리방법은 동일한 알고리즘의 수행단위가 공간적으로 여러 개 존재하여 여러 장치가 동시에 동일한 처리 과정으로 실행되므로 공간적 병렬성(spatial parallelism)을 가지게 된다. 또한, 주기적으로 이웃하는 프로세서와 데이터의 입출력을 수행하게 되고, 각 프로세서의 초기 탐색공간의 차별화에 따라 각기 다른 수행특성으로 구한 해를 공유하는 과정을 통하여 전역해의 탐색 성공 가능성을 높이게 된다. 본 병렬화 모델 구조는 본 논문의 다중 컴퓨터 시스템과 같은 성긴 구조의 병렬컴퓨터에서 최적의 성능을 보이는 구조이며, 적은 프로세서 개수에서도 좋은 결과를 가져올 수 있는 장점을 가지고 있다.

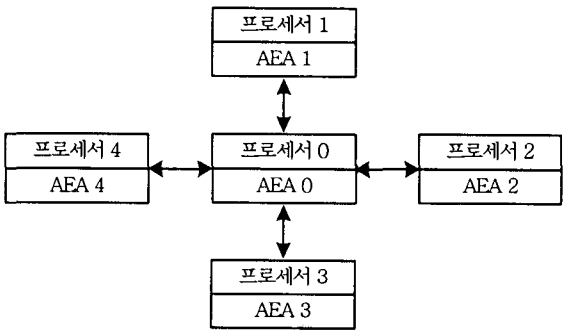


그림 4 적응 진화알고리즘의 병렬화 모델
Fig. 4 Parallel adaptive evolutionary algorithm model

그림 4에서 각 프로세서에 할당되어진 해집단의 초기값은 임의의 실수값으로 생성된 스트링으로 이루어지며, 모든 해집단의 요소는 다르게 구성되어진다. 이와같이 여러 해집단을 동시에 생성, 시뮬레이션함으로써 단일 프로세서의 결과

에 비해 초기값에 따른 영향을 크게 줄일 수 있다. 이러한 해집단은 적응 진화알고리즘의 연산과정을 거쳐서 국지적인 탐색을 수행하게 된다. 이후 각 해집단은 이미 선정된 최적의 교환주기에 의해 일정세대마다 일정 조건에 따라 교환해를 선별한 후 각 프로세서의 양방향 채널을 통해 이웃하는 프로세서와 해를 교환하게 된다. 이를 해의 이주(migration) 과정라고 하며 이러한 해집단의 변화를 통해 전체 영역에 대한 탐색의 국지화를 방지하게 된다. 그림 4의 화살표는 교환하는 해의 경로를 나타낸 것이다. 이러한 적응 진화알고리즘을 병렬화하기 위해서 본 논문에서는 이주주기, 이주스트링 선별, 해집단내에서 대체스트링 선별 순으로 이주과정을 진행하였다.

(1) 이주주기 확인: 이주주기는 해집단의 빌딩블록(building block) 형성 시기를 고려하여 결정되어야 한다. 빌딩블록 형성전 해의 이주가 발생하면 이주해의 효율성이 낮아질 수 있으며, 이를 방지하기 위해 실험을 통한 통계를 이용하여 결정된 이주주기에 따라 해의 교환이 이루어진다.

(2) 이주스트링의 선별: 이주 스트링은 이웃 해집단의 해의 탐색을 돕는 역할을 하며 이러한 선별과정은 해집단내에서 적합도가 높은 스트링을 이주하는 방법, 혹은 이주스트링의 개수만큼 스트링을 추가 생성하여 이주하는 방법이 있다. 본 논문에서는 이주과정 수행시에 주변의 프로세서간에 결과를 서로 비교하여 상대적으로 가장 적합도가 높은 스트링을 이주해로 선택하였다.

(3) 해집단 내의 대체스트링 선별: 해집단 크기를 유지하기 위해서 이주될 스트링과 대체할 스트링을 선별하여야 한다. 여기에서는 해집단 내의 가장 낮은 적합도의 스트링을 제거하고, 타 해집단으로부터 이주된 우수해는 태그변수를 0으로 지정한 후 유전알고리즘의 스트링에 포함하였으며 엘리티즘을 통해 해집단 내에 가장 적합도가 높은 스트링은 진화전략의 스트링에도 반영되어진다.

이러한 교환과정은 적응 진화알고리즘 수행시 일정세대마다 이주가 수행되고, 따라서 모든 해집단은 독립적으로 격리되지 않게 된다. 또한 데이터 전송 과정이 전체 알고리즘 수행과정에 대해서 그 수행빈도가 낮으므로 이주과정에서 발생할 수 있는 시간지연이 크지 않고, 이주해의 규모가 큰 경우에도 좋은 성능을 유지할 수 있다.

4.3 PAEA를 이용한 발전기 기동정지계획

제안한 병렬 적응 진화알고리즘을 이용하여 화력 발전기 기동정지계획 문제를 해결하기 위해서는 스트링 구성이 중요한 과제이다. 기존의 스트링 구성방법은 발전기 기동 및 정지 상태를 시간별로 표시하는 비트형 방법으로서, 발전기 대수가 증가함에 따라 탐색해야 할 변수의 수가 기하학적으로 증가하는 문제점이 있었다. 따라서 본 논문에서는 이러한 문제점을 해결하기 위해서 발전기 기동 및 정지 상태를 실수(정수)로 표현하여 스트링을 구성하였다. 그리고 PAEA의 연산자인 교배 및 돌연변이를 수행한 후, 제약조건을 위배한 스트링을 제약조건을 만족하도록 수정하는 수정연산자를 도입하였다.

(1) 스트링의 구성: 발전기 기동정지계획은 그림 5의 (a)와 같이 각 비트의 0과 1로 매 시간별 발전기의 기동 및 정지상태 표현이 쉽게 가능하지만, 문제의 변수가 많게 되고 제약조건을 고려하기 힘들어 결과적으로 계산량 및 계산소요시간이 늘어나게 된다. 이를 개선하기 위하여 스트링을 발전기와 최소 기동 및 정지시간을 제외한 on-off의 상태지속기간을 나타내는 2차원으로 구성하였으며 제안한 방법으로 구성된 스트링은 그림 (b)와 같다. 이때 그림 5(a) 및 (b)의 표현은 서로 변환이 가능하다.

그림 5(b)의 스트링 요소인 S_{ij}(i 발전기의 j번째 요소)는 on/off상태와 상태지속기간으로 구성되어 있으며, 이러한 방법은 일반적으로 교배, 돌연변이 과정에서 제약조건을 위반하는 스트링을 감소시켜 해의 탐색능력을 크게 향상시키게 된다.

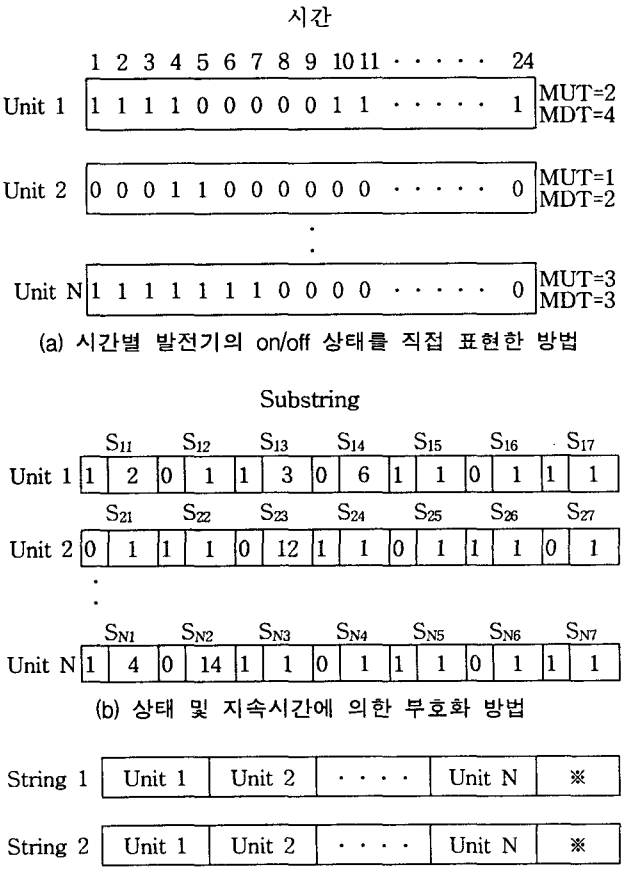


그림 5 스트링의 구성 방법
Fig. 5 String structure

이러한 스트링은 실수형 변수로 구성되며, 알고리즘의 평가과정 수행시 스트링 요소는 정수형 형태로 재구성한 뒤 임시변수에 할당하여 평가함수에 적용하게 된다. 기동정지계획에 있어서는 해에 대한 사전정보가 주어지지 않으므로 상태지속기간의 최소값이 0이며 최대값이 24인 랜덤한 초기값을 사용한다. 그림 5에서 N은 발전기의 수, M은 스트링

개수, MUT와 MDT는 최소 운전 및 정지시간, ※는 GA 또는 ES의 스트링을 나타내는 태그변수를 나타낸다.

(2) 적합도 함수: 적합도를 평가하기 위해서는 문제에 맞는 적절한 적합도 함수를 구성하여야 한다. PAEA는 스트링의 적합도가 최대화되도록 진화하므로 비용함수를 최소화하는 화력발전기의 기동정지계획 문제는 식 (8)과 같이 표현하였다.

$$Fitness = \frac{a}{b+F_T} \tag{8}$$

여기서, F_T : 총 발전비용 (연료+기동+정지비용)

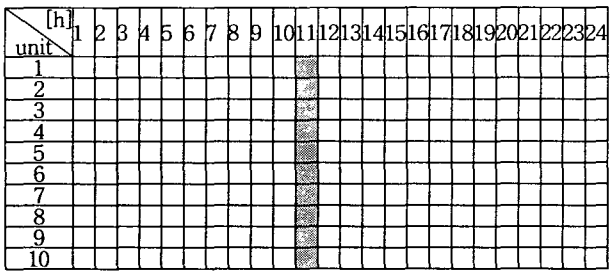
a, b : 임의의 상수

(3) 수정연산자(repairing operator): 본 논문에서는 발전기 최소 운전 및 정지 시간을 만족하도록 스트링을 구성하였지만, PAEA의 진화연산자인 교배와 돌연변이를 수행하면 제약조건을 위배하는 스트링이 발생할 수 있다. 따라서, 교배와 돌연변이 과정 후 제약조건을 위배한 스트링에 대해서 수정연산자를 적용하였다. 수정연산자는 스트링의 수정 과정을 통해 제약 조건을 위반한 해의 구조를 제약조건을 만족하도록 변경하는 방법으로 특히 발전기 기동정지 계획과 같은 조합 최적화 문제에 있어 해의 탐색능력을 개선시킬 수 있다.

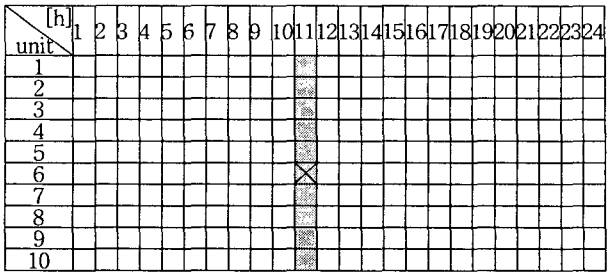
적용한 방법은 우선 PAEA의 평가과정에서 제약조건의 위반여부를 확인하고 발전예비력 제약조건이 위배되는 스트링에 대해 수정연산자를 적용하였다. 이는 특정 시간대의 on 상태 발전기들에 대해 최대출력값의 합이 식 (6)의 발전예비력을 만족하지 못하는 경우에 해당되며, 이때 제약조건을 만족시키기 위해 해당시간에 대한 off상태 발전기에 대해 연료단가가 가장 낮은 발전기부터 차례로 스위치를 on상태로 변경하여 발전기를 추가 기동한다. 여기서 연료단가는 식 (9)와 같이 적용하였다.

$$\text{연료단가} = \frac{\text{최대출력시의 발전비용}}{\text{발전기의 최대 출력량}} \tag{9}$$

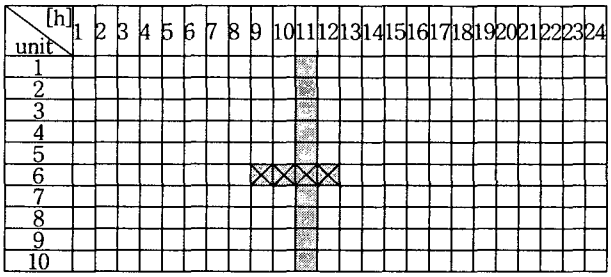
그러나 해의 수정과정에서 발전기의 최소 기동 및 정지시간 제약조건을 위배할 수 있으므로 추가적으로 최소 기동 및 정지시간 제약조건을 검토하고 이를 위반시 제약조건을 만족하도록 해를 수정하였다. 수정연산자의 전 과정을 그림 6에 나타내었다.



(a) 발전예비력 제약조건을 위배하는 시간대 검출



(b) 연료단가가 낮은 발전기를 우선 추가 기동하도록 스트링 수정



(c) 대상 발전기에 대해 최소 기동 또는 정지시간 제약조건을 만족하도록 재수정

여기서, ⊠ : 발전기 기동정지상태가 달라진 구역

그림 6 수정연산자

Fig. 6 Repairing operator

5. 사례 연구

5.1 PAEA의 시뮬레이션 조건

본 논문에서는 제안한 PAEA의 유용성을 검증하기 위해 참고문헌[14]의 사례에 적용하여 24시간의 예상 전력수요에 대한 발전기 10대, 20대, 40대, 60대, 80대, 100대의 기동정지계획을 도출하여 이를 종래의 방법과 비교·검토하였다. 화력 발전기 기동정지계획에 사용된 PAEA의 파라미터 값을 표 1에 나타내었다.

표 1 PAEA의 시뮬레이션 계수

Table 1 Simulation parameters of the PAEA

파라메터 / 발전기대수	해집단 크기	교배확률	돌연변이 확률	δ	C_d	C_i
10	30	0.35	0.05	10	0.935	1.040
20	30	0.35	0.05	10	0.935	1.040
40	30	0.35	0.05	10	0.940	1.040
60	30	0.35	0.01	10	0.945	1.035
80	30	0.35	0.01	10	0.955	1.035
100	30	0.35	0.01	10	0.965	1.030

발전기 10대에 대한 전력수요는 표 2에 나타내었으며, 발전예비력은 전력수요의 10%로 가정하였다. 표 3은 발전기 10대에 대한 모의계통의 데이터를 나타낸 것이며, 큰 규모의 계통 적용시에는 표 3의 발전기 세트가 다시 추가되도록 모의계통을 구성하였으며, 그에 따라 전력 수요도 배수적으로 증가하도록 하였다. 또한 PAEA은 실행할 때마다 결과가 달라지게 되므로 동일한 파라미터에 대해 10번씩 프로그램을 실행한 후 평균시간을 실행시간으로 하였다. 각 프로세서는

매 20세대마다 각 해집단에서 가장 우수한 해를 서로 공유하도록 하였다. 이는 각 프로세서의 독립적인 해의 탐색특성을 유지하면서 특정 해집단이 조기수렴하는 현상을 방지하도록 여러 차례의 실험을 통하여 결정된 수치이다. 성능평가를 위하여 사용된 병렬 컴퓨터 시스템은 프로세서가 16개이며, 각 프로세서의 속도가 100Mhz인 Parsytec사의 CC/16이다.

표 2 시간별 부하수요
Table 2 Power demand

시간	1	2	3	4	5	6	7	8	9	10	11	12
부하	700	750	850	950	1000	1100	1150	1200	1300	1400	1450	1500
시간	13	14	15	16	17	18	19	20	21	22	23	24
부하	1400	1300	1200	1050	1000	1100	1200	1400	1300	1100	900	800

표 3 기동정지계획문제에 사용된 발전기 파라메터
Table 3 Generator parameters for unit commitment
(a, b, c : 연료비용 계수)

Unit# / 파라메터	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5	Unit 6	Unit 7	Unit 8	Unit 9	Unit 10
P_{max}[MW]	455	455	130	130	162	80	85	55	55	55
P_{min}[MW]	150	150	20	20	25	20	25	10	10	10
a[$/h]	1000	970	700	680	450	370	480	660	665	670
b[$/MWh]	16.19	17.26	16.60	16.50	19.70	22.26	27.74	25.92	27.27	27.79
c[$/MWh^2-h]$\times 10^{-3}$	0.48	0.31	2.00	2.11	3.98	7.12	0.79	4.13	2.22	1.73
최소운전시간[h]	8	8	5	5	6	3	3	1	1	1
최소정지시간[h]	8	8	5	5	6	3	3	1	1	1
가열기동비용[$]	4500	5000	550	560	900	170	260	30	30	30
냉각기동비용[$]	9000	10000	1100	1120	1800	340	520	60	60	60
냉각기동시간[h]	5	5	4	4	4	2	2	0	0	0
초기상태[h]	8	8	-5	-5	-6	-3	-3	-1	-1	-1

5.2 시뮬레이션 결과

(1) 해집단 개체수의 변화

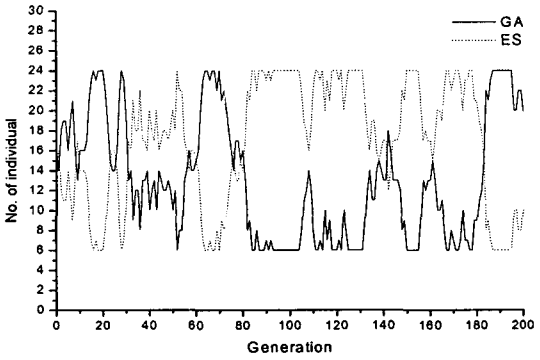
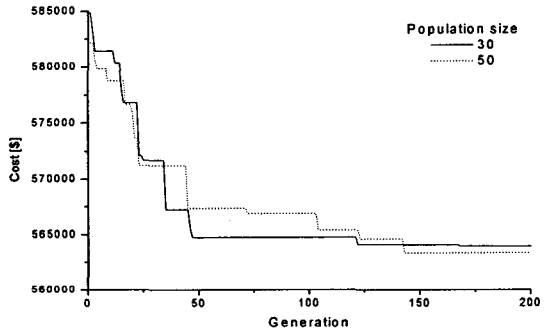


그림 7 매 세대별 GA, ES의 개체 수
Fig. 7 Number of Indiviuals of GA and ES in each generation

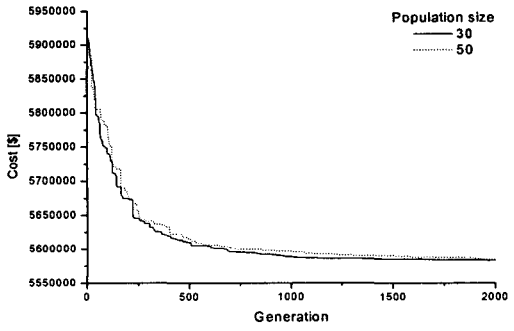
그림 7은 발전기 10대에 대해 세대별 유전알고리즘 및 진화전략의 개체수를 나타낸 것으로, 초기 세대에서는 유전알고리즘의 스트링 수가 증가하고 세대수가 진행됨에 따라 진화전략의 스트링 수가 더 많아짐을 알 수 있다. 얼마 후에는 다시 유전알고리즘의 스트링 수가 증가하게 되는데, 이러한 과정이 주기적으로 반복됨으로서 제안한 방법은 적응적으로 해를 탐색해 나감을 알 수 있다.

(2) 해집단에 따른 성능 비교

그림 8은 각 사례에 대해 제안한 방법을 적용한 후 세대수에 따른 발전비용 추이를 나타낸 것이다. 그림 8에서 실선은 각 프로세서의 해집단의 수가 30일 때, 점선은 각 프로세서의 해집단의 수가 50일 때를 나타내었다. 그림 9에서 보는 바와 같이 각 프로세서의 해집단을 구성하는 스트링 개수가 30개와 50개일 때의 수렴 특성이 유사함을 알 수 있다. 이는 매 세대마다 적응적으로 유전알고리즘 및 진화전략의 개체의 비율을 조절하는 진화특성을 감안할 때, 해집단의 크기가 작더라도 내부의 적응적인 개체수 변화에 따라 효율적인 탐색이 가능함을 알 수 있다.



(a) 발전기 10대



(b) 발전기 100대

그림 8 각 사례에 대해 제안한 방법으로 구한 비용곡선
Fig. 8 Cost curve using the proposed method in each case

표 4는 해집단 크기에 대한 발전 비용 및 평균 실행시간을 나타내었다. 표 4에 나타난 결과로써 스트링개수가 50개인 경우가 30개인 경우에 비해 실행시간이 약 1.7~2배 정도의 시간이 요구됨을 알 수 있었으며, 이는 상대적으로 많은 스트링의 평가에 의해 실행시간이 길어졌음을 나타낸다. 즉, 해집단 크기를 선택함에 있어서 빌딩블록을 형성할 수

있는 적절한 크기의 해집단을 운용하였을 때 실행시간을 최소화하면서 효율적인 탐색이 가능함을 알 수 있다.

표 4 해집단 크기에 대한 발전 비용 및 평균 실행시간 (10회시행)

Table 4 Total generation cost and computation time in population sizes

분류 / 발전기대수	스트링개수(30개)		스트링개수(50개)	
	발전비용	실행시간 (평균값)	발전비용	실행시간 (평균값)
10	563583	1:41	563316	3:22
20	1123037	6:53	1122774	17:19
40	2242508	18:13	2242806	31:31
60	3349024	58:44	3351686	1h43min
80	4463728	1h56min	4462379	3h22min
100	5583193	3h4min	5584463	5h28min

(3) 프로세서 개수에 따른 성능 비교

전체 해집단의 크기가 동일한 상태에서 프로세서 개수가 2개, 4개, 8개, 16개인 경우에 대하여 그 결과를 비교하였다. 각 프로세서 개수는 하이퍼큐브 구조를 형성하기 위한 개수로써 각각 1~4 차원 하이퍼큐브 구조로 구성되며, 이는 그림 3의 (b)~(e)와 같다. 이 경우 하이퍼큐브의 차원수는 곧 이웃하는 해집단의 개수가 된다. 그림 9는 발전기 10대에 대하여 프로세서수에 따른 수행시간을 나타낸 것이다. 프로세서 개수가 2개일 때를 기준으로 하여 8개인 경우에 75%의 수행시간이 감소하였으며, 16개인 경우에 88%의 수행시간이 감소하였다. 따라서 제안한 병렬 적응진화알고리즘은 프로세서 개수가 증가함에 따라 비례적으로 계산시간이 개선됨을 알 수 있다.

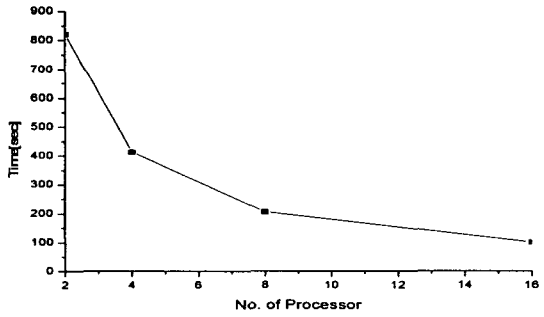


그림 9 프로세서 개수에 따른 수행시간

Fig. 9 Execution time by the change of the number of processors

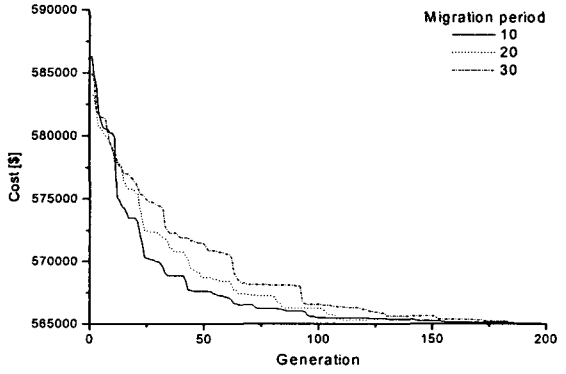
(4) 이주주기에 의한 성능 비교

PAEA의 파라미터의 성능비교를 위해 화력발전기 10대 계통에 대해 이주주기를 10, 20, 30세대로 하였을 때의 총 발전비용 및 계산시간을 표 5에 나타내었다. 이주주기가 10세대인 경우, 초기세대에서는 좋은 탐색성능을 보였으나, 점점 발전비용이 천천히 감소함을 보여준다. 이는 잦은 해의 교환으로 인하여 각 프로세서에 할당된 해집단이 동일해져서, 탐색 방향 또한 같아진 결과이다. 또한 이주주기가 30세대인 경우는 전체적으로 발전비용이 천천히 감소하는 현상을 보였으며, 이 경우에는 해가 효율적으로 인접 프로세서와 교환되도록 이주주기를 줄여야 한다. 본 시뮬레이션의 결과를 토대로 하여 발전비용 및 계산시간을 검토한 결과, 이주주기가 20세대인 경우에 대해서 가장 효율적인 탐색이 이루어졌음을 알 수 있다. 또한 이주주기가 10세대인 경우 잦은 해의 이동으로 인하여 통신소요시간이 이주주기가 20세대인 경우보다 1.1배 늘어났으며, 대규모 계통에 적용시 스트링 크기의 대형화에 따른 데이터 전송량의 증가로 인하여 더 큰 시간지연이 발생할 수 있음을 예상할 수 있다. 이주주기가 30세대인 경우에서는 발전비용 및 수행시간에 있어서 낮은 성능을 보였는데, 이는 전자의 두 경우와 비교하여 효율적인 해의 이동이 이루어지지 못하고 국지적 탐색이 강조됨으로서 오히려 전역 탐색 능력이 떨어졌음을 유추할 수 있다.

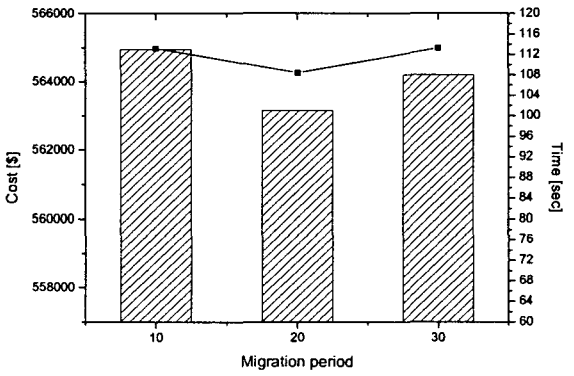
표 5 이주 주기에 대한 성능 비교 (10회 시행)

Table 5 Simulation results by the change of the migration period

이주주기 / 구분	10세대	20세대	30세대
발전비용[$]	563596	563583	564069
평균시간[sec]	113	101	108



(a) 이주주기의 변화에 대한 발전비용



(b) 이주주기 변화에 대한 총비용 및 실행시간

그림 10 이주주기의 변화에 대한 시뮬레이션 결과

Fig. 10 Simulation results by the change of the migration period

(5) 기존의 방법과 성능 비교

일반적인 수치해석법인 라그랑지 미정계수법(LR)을 기준

으로 하여 참고 문헌[14,15]에서 제안한 유전알고리즘 및 진화프로그래밍(evolutionary programming: EP) 방법과 본 논문에서 제안한 PAEA의 결과를 서로 비교하였다. 표 6은 발전기 대수별로 본 논문에서 제안한 PAEA로 찾은 가장 우수한 해, 가장 나쁜 해, 실행시간 및 세대수를 나타내었다. 표 7은 기존의 방법과의 발전비용을 비교하였다. 표 7에서 보는 바와 같이, 제안한 방법은 기존의 방법보다 더 낮은 비용을 구함을 알 수 있었고, 표 8은 LR 방법을 통해 얻어진 발전비용을 기준으로 하여 각 방법에 대해 식 (10)을 사용하여 해의 개선정도를 비교한 것이다.

$$\text{성능개선정도}(\%) = \frac{\text{LR 결과} - \text{비교대상 결과}}{\text{LR결과}} \times 100 \qquad (10)$$

표 6 각 사례에 대한 시뮬레이션 결과(10회시행, 이주주기=20)

Table 6 Simulation results in each case

분류 / 발전기대수	Best	Worst	Difference	Time	세대수
10	563583	565838	0.40	1:41	200
20	1123037	1124542	0.21	6:53	400
40	2242508	2245447	0.25	18:13	500
60	3349024	3361548	0.45	58:44	1000
80	4463728	4478186	0.32	1h56min	1500
100	5583193	5596814	0.19	3h4min	2000

표 7 기존의 방법과의 발전비용 비교(10번 시행, 단위 [$])

Table 7 comparison with total costs of conventional and the proposed methods

방법 / 발전기대수	LR	GA	EP	PAEA
10	565825	565825	564551	563583
20	1130660	1126243	1125494	1123037
40	2258503	2251911	2249093	2242508
60	3394066	3376625	3371611	3349024
80	4526022	4504933	4498479	4463728
100	5657277	5627437	5623885	5583193

표 8 LR방법을 기준으로 하였을 때의 해의 개선정도 비교 [%]

Table 8 Comparisons with improvement rate of conventional and the proposed method standardized on LR

방법 / 발전기대수	GA	EP	제안한 방법 (Best 기준)
10	0.00	0.22	0.39
20	0.39	0.45	0.67
40	0.29	0.41	0.70
60	0.51	0.66	1.32
80	0.46	0.60	1.38
100	0.52	0.59	1.31

표 9는 본 논문에서 제안한 방법과 기존의 방법과의 실행시간을 비교하였다. 각 방법에 대하여 제안한 방법이 유전알고리즘 방법보다 1.4~2.4배, 500Mhz의 CPU를 사용한 EP 방법을 100Mhz로 환산하여 비교할 경우에 대해서는 2.5~5배 더 빠른 수행특성을 보임을 알 수 있었다.

표 10과 11에 10대의 발전기에 대하여 기동정지계획에 의한 각 발전기의 기동정지 상태 및 경제급전 결과를 나타내었다. 이 표를 통해, 실제로 해가 모든 제약조건을 만족하면서 유용한 해를 얻었음을 알 수 있다. 또한 표 12에 110대의 발전기 계통에 대한 기동정지계획 결과를 나타내었다.

표 9 수행시간 비교 (10번 시행, 평균시간)

Table 9 Execution time comparison

방법 / 발전기대수	GA (genetic algorithm)	EP (evolutionary programming)	PAEA
10	3:41	1:40	1:41
20	12:13	5:40	6:53
40	44:57	19:36	18:13
60	1h37min	37:47	58:44
80	2h47min	59:44	1h56min
100	4h22min	1h42min	3h4min
*테스트 환경	HP Apollo 720 Workstation (CPU 100Mhz)	HPC160 Workstation (CPU 500Mhz)	Parsytec CC/16 (CPU 100Mhz)

표 10 발전기 10대의 기동 및 정지상태

Table 10 On/off states of the 10-generator system

시간 / Unit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1																								
2																								
3																								
4																								
5																								
6																								
7																								
8																								
9																								
10																								

표 11 10대의 발전기에 대한 시간별 부하수요, 발전량

Table 11 Power demand, power generation for the 10 generators

시간별 부하수요		Unit	Unit	Unit	Unit	Unit	Unit	Unit	Unit	Unit	Unit	총발전량
시간	부하	1	2	3	4	5	6	7	8	9	10	[MW]
1	700	455	245									700
2	750	455	295									750
3	850	455	370			25						850
4	950	455	455			40						950
5	1000	455	390		130	25						1000
6	1100	455	360	130	130	25						1100
7	1150	455	410	130	130	25						1150
8	1200	455	455	130	130	30						1200
9	1300	455	455	130	130	85	20	25				1300
10	1400	455	455	130	130	162	33	25	10			1400
11	1450	455	455	130	130	162	73	25	10	10		1450
12	1500	455	455	130	130	162	80	25	43	10	10	1500
13	1400	455	455	130	130	162	33	25	10			1400
14	1300	455	455	130	130	85	20	25				1300
15	1200	455	455	130	130	30						1200
16	1050	455	310	130	130	25						1050
17	1000	455	260	130	130	25						1000
18	1100	455	360	130	130	25						1100
19	1200	455	455	130	130	30						1200
20	1400	455	455	130	130	162	38		10	10	10	1400
21	1300	455	455	130	130	100	20		10			1300
22	1100	455	455		130	40	20					1100
23	900	455	420			25						900
24	800	455	345									800

표 12 발전기 100대의 기동 및 정지상태

Table 12 On/off states of the 100 generator system

시간 / Unit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
20	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
21	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
22	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
23	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
24	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
25	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
26	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
27	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
28	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
29	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
31	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
33	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
34	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
35	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
36	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
37	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
38	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
39	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
40	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
41	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
42	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
43	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
44	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
45	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
46	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
47	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
49	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
50	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
51	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
52	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
53	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
54	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
55	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
56	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
57	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
58	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
59	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
61	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
62	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
63	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
64	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
65	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
66	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
67	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
68	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
69	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
70	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
71	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
72	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
73	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
74	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
75	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
76	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
77	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
78	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
79	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
80	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
81	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
82	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
83	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
84	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
85	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
86	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
87	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
88	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
89	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
90	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
91	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
92	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
93	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
94	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
95	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
96	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
97	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
98	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
99	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
100	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

6. 결 론

현재 전력산업은 경쟁요소의 도입에 의해 커다란 변화를 맞고 있다. 기간산업으로서의 중요성에 따라 전력계통의 효율적인 운용방안의 중요성도 더욱 커지고 있다. 본 논문에서는 화력 발전기 기동정지계획 문제를 해결하기 위하여 효율적이면서 실 문제에 적용이 용이한 적응 진화알고리즘을 제안, 적용하였다. 특히 적응 진화알고리즘을 병렬화함으로서 기존 방법으로 계산하였을 때 발생하는 시간적인 제약과 준 최적해에 수렴하는 현상을 개선하여 보다 실질적인 대규모의 대상 문제에 접근할 수 있도록 하였다. 이와함께 적응 진화알고리즘에 적합하도록 실수형 스트링 구성을 제안하였으며, 해가 제약조건을 만족하도록 수정연산자를 도입하였다.

제안한 방법의 유용성을 검토하기 위해 참고문헌[14]에 있는 사례계통에 적용하여 기존의 결과와 비교, 분석하였다. 병렬 적응 진화알고리즘의 매 세대별 특성을 분석한 결과, 단일 프로세서에서 수행하였을 때와 동일하게 적응적으로 GA 및 ES 개체수가 변화하는 것을 확인하였다. 10~100대의 발전기 계통에 대해서 기존의 방법과 비교한 결과, 총 발전비용면에서 제안한 방법이 수치적 방법인 LR 방법으로 구한 해의 비용보다 0.39~1.38%의 비용을 절감하였으며, 확률적 기법인 GA, EP 방법보다는 각각 0.39~0.92%, 0.17~0.78%의 비용을 절감하였다. 동일한 CPU 속도로 환산한 경우, 제안한 방법이 실행시간면에서 GA 방법보다 1.4~2배, EP 방법보다 2.5~5배 정도 빠르게 결과를 구할 수 있었다. 이와 같이 제안한 방법은 병렬 프로세스와 적응 진화 알고리즘에 의해 기존의 방법보다 우수한 탐색 성능을 보임을 알 수 있었다.

감사의 글

본 연구는 한국과학재단의 지역대학 우수과학자 지원연구에 의하여 이루어진 연구로서, 관계부처에 감사드립니다. (R05-2003-000-12380-0)

참 고 문 헌

[1] A. J. Wood, B. F. Wollenberg, *Power Generation, Operation, and Control*, John Wiley & Sons, 1996.

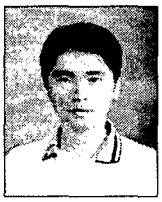
[2] S. J. Wang, S. M. Shahidehpour, D. S. Kirschen, S. Mokhtari, G. D. Irisarri, "Short-term Generation Scheduling with Transmission and Environmental Constraints Using an Augmented Lagrangian Relaxation", *IEEE Transaction on Power Systems*, Vol. 10, No. 3, pp.1294-1301, August 1995.

[3] E. Khodaverdia, A. Bamellar, and R. M. Dunnet, "Semirigorous Thermal Unit Commitment for Large Scale Electrical Power Systems", *Proceedings IEEE Part C*, pp.157-164, 1986.

[4] Hobbs, G. Herman, S. Warner, J. B. Sheble, "An Enhanced Dynamic Programming Approach for Unit Commitment", *IEEE Transaction on Power Systems*,

Vol. 3, No. 3, pp.1201-1205, 1988.
[5] A. L. Cohen, and M. Yoshimura, "A Branch-and-Bound Algorithm for Unit Commitment", *IEEE Trans. PAS-102*, No. 2, pp.444-449, 1983.
[6] T. S. Dillon, K. W. Edwin, H. D. Kochs, and R. J. Taud, "Integer Programming Approach to the Problem of Unit Commitment with Probabilistic Reserve Determination", *IEEE Trans. on PAS-97, No. 6*, pp.2154-2166, 1978.
[7] J. F. Bard, "Short-term Scheduling of Thermal-electric Generators Using Lagrangian Relaxation", *Oper. Res.*, Vol. 36, No. 5, pp.756-766, 1988.
[8] Rajan, C.C.A.; Mohan, M.R.; Manivannan, K., "Refined simulated annealing method for solving unit commitment problem", Neural Networks, 2002, IJCNN, Vol. 1, pp.333-338, May 2002.
[9] A. H. Mantawy, Y. L. Abdel-Magid, S. Z. Selim, "Unit Commitment by Tabu Search", *IEE Proceedings - Generation, Transmission and Distribution*, Vol. 145, No. 1, pp.56-64, 1998.
[10] H. T., Yang, P. C., and Huang, C. L., "A Parallel Genetic Algorithm Approach to Solving the Unit Commitment Problem: Implementation on the Transputer Networks", *IEEE Transactions on Power Systems*, Vol. 12, No. 6, pp.1062-1069, 1997.
[11] S. O. Orero, and M. R. Irving, "Large Scale Unit Commitment Using a Hybrid Genetic Algorithm", *Internaltional Journal of Electrical Power & Energy Systems*, Vol. 19, No. 1, pp.45-55, 1997.
[12] 황기현, "새로운 적응진화 알고리즘을 이용한 전력계통 퍼지 안정화장치의 설계", 박사논문, 부산대학교, 2000.
[13] R. Tanese, "Parallel Genetic Algorithm for a Hypercube", *Proceedings of the 2nd International Conference on Genetic Algorithms*, pp.177-183, 1987.
[14] S. A. Kazarlis, A. G. Bakirtzis, and V. Petridis, "A Genetic Algorithm Solution to the Unit Commitment Problem", *IEEE Transaction on Power Systems*, Vol. 11, No. 1, pp.83-92, 1996.
[15] K. A. Juste, H. Kita, E. Tanaka, and J. Hsegawa, "An Evolutionary Programming Solution to the Unit Commitment Problem", *IEEE Transaction on Power Systems*, Vol. 14, No. 4, pp.1452-1459, 1999.

저 자 소 개



김 형 수 (金 亨 洙)

1972년 1월 26일생. 1994년 부산대 공대 전기공학과 졸업. 1997년 동 대학원 전기공학과 졸업(석사). 2002년 동 대학원 전기공학과 졸업(박사). 현재 부산대학교 BK Post Doc.
Tel : 051-510-3188, Fax : 051-513-0212
E-mail : kimhsu@pusan.ac.kr

조 덕 환 (趙 德 桓)

1974년 10월 30일생. 2000년 부산대 공대 전기공학과 졸업. 2002년 동 대학원 졸업(석사). 현재 현대중공업 기계전기연구소 시스템제어연구실 선박 IT팀 연구원.
Tel : 031-289-5229, Fax : 031-289-5250
E-mail : t2move@naver.com

문 경 준 (文 景 俊)

1972년 10월 25일생. 1994년 부산대 공대 전기공학과 졸업. 1996년 동 대학원 졸업(석사). 2005년 동 대학원 전기공학과 졸업(공박). 현재 한국원자력연구소 양성자기반공학기술개발사업단 선임연구원.
Tel : 042-868-4663, Fax : 042-868-8131
E-mail : kjmun@kaeri.re.kr

이 화 석 (李 和 錫)

1966년 7월 10일생. 1991년 부산대 공대 전기공학과 졸업. 1993년 동 대학원 전기공학과 졸업(석사). 1997년 동 대학원 전기공학과 졸업(공박). 현재 거제대학 전기과 부교수.
Tel : 055-680-1604, Fax : 055-681-3993
E-mail : hslee@.koje.ac.kr

박 준 호 (朴 俊 灝)

1955년 9월 17일생. 1978년 서울대 공대 전기공학과 졸업. 1980년 동 대학원 전기공학과 졸업(석사). 1987년 동 대학원 전기공학과 졸업(공박). 현재 부산대 공대 전자전기정보컴퓨터공학부 교수.
Tel : 051-510-2370
E-mail : parkjh@pusan.ac.kr

황 기 현 (黃 琪 鉉)

1968년 3월 1일생. 2000년 부산대대학원 전기공학과 졸업(박사). 현재 동서대학교 유비쿼터스 컴퓨터그래픽스응용 지역기술혁신센터 전임교수
Tel : 051-320-1938
E-mail : hwanggh@gdsu.dongseo.ac.kr