한국지능정보시스템학회
02 춘계학술대회 논문집, pp. 435~441

대규모 최적화 문제의 해결을 위한 메타휴리스틱 알고리즘의 병렬화

이용환* · 류광렬*

Parallelization of Metaheuristic Algorithms to Solve the Large-scaled Optimization Problem

Yong-Hwan Lee · Kwang-Ryel Ryu

요 약

전력시스템 등, 산업 전반의 많은 분야에 최적화 문제가 산재해 있다. 또한 이러한 최적화 문제를 해결하기 위한 많은 연구가 있었다. 특정 응용에 국한되지 않고 모든 응용에 적용 가능한 메타휴리스틱 알고리즘은 그 중 많은 비중을 차지하고 있으며, 가장 대표적인 방법은 유전알고리즘과 타부 탐색이다. 그러나 최적화 문제에 속하는 많은 문제들이 탐색공간이 방대하고 많은 제약이 존재하는 대규모 최적화 문제로서 기존의 메타휴리스틱 기법들을 그대로 이용해서는 빠른 시간 내에 최적의 해를 찾아내기 힘들다. 본 논문에서는 대규모 최적화 문제의 하나인 발전기 기동정지 계획 문제를 해결하기 위하여 유전알고리즘과 타부 탐색을 적용하고 그 성능을 분석한다. 그리고 각 방법을 병렬화하여 수행함으로써 병렬화를 통하여 시간상의 이득과 함께 부가 효과로서 집중화와 다각화의 효과를 얻을 수 있음을 보여준다.

Key words : 메타휴리스틱, 유전알고리즘, 타부 탐색, 병렬화, 발전기 기동정지 계획, 최적화

1. 서 론

산업 전반에 걸쳐 많은 최적화 문제들이 존재하고, 그 중 많은 문제들이 방대한 탐색공간과 많은 제약을 지닌 대규모 최적화 문제이다. 이러한 대규모 최적화 문제들은 방대한 탐색공간과 제약에 의한 문제의 복잡성 때문에 최적의 해를 찾는 것이 계산적으로 불가능하다. 따라서 최적의 해를 찾기보다는 최적에 가까운 해를 찾고자 하는 노력이 많이 이루어졌다. 최적화 문제의 현실적 해결을 위하여 hill-climbing 탐색, greedy 탐색과 같은 휴리스틱 기법들과 유전알고리즘, 타부 탐색, ant colony optimization, simulated annealing등의 많은 메타휴리스틱 기법들이 개발되고 사용되었다. 휴리스틱 기법들은 문제에 대한 정보를 이용하여 빠르게 탐색을 하지만 쉽게 국지적 최적해에 빠진다. 따라서 탐색 시간이 다소 길어지더라도 보다 세련된 탐색 전략을 구사하여 전역적 탐색을 할 수 있는 메타휴리스틱 기법에 기반한 방법들이 많이 이용된다.

본 논문에서는 대규모 최적화 문제 중 하나인 발전기 기동정지 계획의 수립을 위하여 최적화 문제에 가장 널리 이용되는 메타휴리스틱 기법인 유전알고리즘(Goldberg, 1989)과 타부 탐색(Glover and Laguna, 1997)을 적용한다. 그러나 두 방법 모두 대규모 최적화 문제를 효율적으로 탐색하기에는 약간의 한계가 있다. 유전알고리즘은 해집단을 기반으로 탐색을 수행하여 전역적 탐색 능력이 우수하지만 확률적 연산으로 인해 오랜 탐색시간이 요구되어진다. 반면 타부 탐색은 비록 적응성 있는 메모리(adaptive memory)를 사용하여 어느 정도의 전역적 탐색 능력을 가지지만, 지역적 탐색을 기초로 하고 있어 전역적 탐색에 한계가 있다. 따라서 본 논문에서는 이러한 단점을 극복하기 위한 방안으로서 각 기법을 병렬로 구현하여 보다 우수한 탐색 능력을 가지도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 대상 문제인 발전기 기동정지계획에 대해서 설명한다. 3, 4장에서는 각 방법의 구현에 대해서 소개한다. 그리고 5장에서는 실 데이터에 대해서 각 방법을 실험하고 비교한다. 6장에서는 결론을 맺고 앞으로의 향후 연구방향에 대해서 토의한다.

2. 발전기 기동정지 계획

발전기 기동정지 계획이란 하나의 전력시스템을

※ 본 연구는 한국과학재단 목적기초연구(과제번호 R01-1999-00216)지원으로 수행되었음.

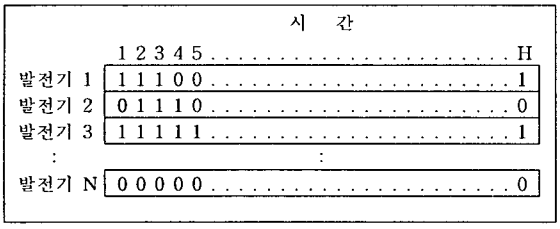
* 부산대학교 컴퓨터공학과

형성하는 여러 대의 발전기에 대해서 안정적인 전력 공급체계를 확보하면서 총 발전비용을 최소화하기 위해 각 발전기별로 일간 또는 수일간의 기동 및 정지시간을 결정하는 작업이다. 전력 수급량이 증가하고 발전기 수가 늘어나면서 종류도 다양해짐에 따라 효율적인 기동정지계획의 수립은 경제성의 관점에서 중요한 문제로 부각되었다. 그러나, 각 발전기마다 고려해야 할 제약조건이 많을 뿐 아니라 여러 발전기 간에 존재하는 제약조건들도 많아 효율적인 발전계획을 도출하는 것이 상당히 어렵다.

발전기 기동정지 계획에서 고려해야 할 제약 조건들은 다음과 같다.

(1) 발전요구량: 매 시간대별로 요구되는 발전량으로 부하수요와 송전손실량을 함께 만족시킬 만큼 전력을 공급하여야 한다.
(2) 운전예비력: 예상하지 못한 전력부하의 증가나 일부발전기의 고장에 대비해 항상 어느 정도의 예비 발전량을 공급하여야 한다.
(3) 발전기의 초기 조건: 계획 시점에서의 각 발전기의 기동정지 상태 및 지속시간 등을 고려하여야 한다.
(4) 최소기동시간, 최소정지시간: 각 발전기는 그 종류에 따라 기동정지 상태를 전환하기 전에 이전상태에서 지속하여야 하는 최소 요구시간이 있다.
(5) 기타 발전기의 최대, 최소 발전 한계량, 상시운전 등의 여러 제약사항이 있다.

(3), (4)번 제약조건은 발전기 별로 반드시 지켜야 하는 절대적 제약으로 어느 하나의 발전기라도 지켜지지 않으면 계획에 따라 발전을 할 수가 없다. 반면 (1), (2)번의 제약은 지키지 않는다 하여 계획대로 발전이 되지 않는 것은 아니다. 그러나 경제성을 위해서는 요구량(발전요구량+운전예비력)보다는 많은 양을 발전하면서 요구량과 실 발전량의 차이는 최소가 되도록 하여야 한다. 실 발전량이 요구량보다 적게 되면 대규모 정전과 같은 사고 발생의 우려가 있다.



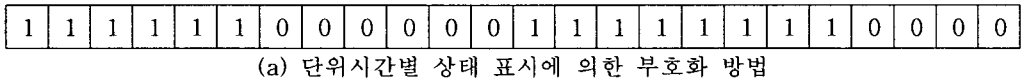
<그림 1> 발전기 기동정지 계획의 표현

발전기 기동정지 계획을 표현하기 위한 한 방안으로서 각 발전기마다 시간별로 기동정지 여부를 이진값(기동 1, 정지 0)으로 나타내어 보면 <그림 1>과 같이 된다. 예를 들어 특정발전기의 24시간 계획이 000000 111111 000000 000000 이라면 0시부터 6시까지 정지하고 6시부터 12시까지 기동, 다시 12시부터 24시까지 정지하는 것을 의미하게 된다. 만약 이 발전기의 최소정지시간이 6시간이고 최소기동시간은 12시간이라면 이 계획은 최소기동시간을 만족하지 못하기 때문에 잘못된 계획이 된다. 최소시간 제약에 의해서 특정 시간의 기동정지 여부를 수정하기 위해서는 주위의 몇 시간이 동시에 수정되어야 한다. 따라서 동시에 여러 시간대의 발전량에 변화가 발생되고 발전비용에도 많은 차이가 난다. 이와 같이 발전기 기동정지 계획은 해의 일부분의 변화에 의해서 평가값이 대폭 변화하는 불규칙적인 탐색공간을 가지고 있어 최적해의 탐색이 매우 힘든 문제이다.

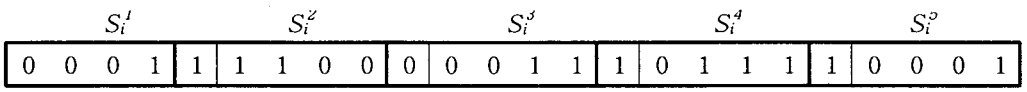
2.1 후보해의 부호화 방법

발전기 기동정지 계획의 수립을 위하여 유전알고리즘과 타부 탐색을 적용하기 위해서는 우선 후보해를 부호화 하여야 한다. 가장 쉽게 생각할 수 있는 방법은 발전기 기동정지 계획의 표현 방법 그대로 단위시간별 상태를 표시하여 부호화하는 방법이다. 즉, <그림 2>(a)와 같이 발전기별로 시간당 하나의 비트씩 할당하여 이진값으로 나타내는 방법이다. 이때 이웃해의 생성은 임의의 비트를 변경하여 생성할 수 있다. <그림 2>(a)에서 최소정지시간과 최소기동시간이 모두 6이라고 하면 그림의 계획은 최소시간 제약을 따르고 있는 계획이 된다. 그러나 이웃해 생성을 위하여 7시의 계획이 변경된다면 최소정지시간을 어기는 해가 된다. 따라서 위의 방법을 따를 경우 유전알고리즘에서 crossover를 하거나 타부 탐색에서 이웃해를 생성하는 과정에서 최소정지시간과 최소기동시간을 지키지 못하는 해가 나타날수 있으며 이러한 최소시간 제약을 만족하는 실행 가능한 해(feasible solution)를 찾는 것이 힘들어 진다. 본 논문에서는 최소정지시간과 최소기동시간을 반드시 지키도록 하는 (Mun *et al.* 1997)의 방법을 개선하여 사용하였다.

<그림 2>(b)와 같이 하나의 발전기 i에 대한 부호화된 스트링(string) S_i는 여러 개의 부스트링(substring)으로 이루어진다. 하나의 부스트링은 부호를 표시하는 부호부 1비트와 지속시간을 나타내는 지속시간부 n_i비트로 나타낸다. 단, 첫 번째 부



(a) 단위시간별 상태 표시에 의한 부호화 방법



(b) 부스트링별 상태지속시간 표시에 의한 방법

<그림 2> 후보해의 부호화 방법

스트링은 부호를 가지지 않는다.

각 부스트링에서 부호부는 기동정지 여부를 나타내고, 지속시간부는 n_i개의 비트가 나타내는 정수값에 발전기 i 고유의 최소정지(기동)시간을 더하여 실제 지속시간을 나타낸다. 각 부스트링의 의미는 부호부가 나타내는 상태를 지속시간만큼 지속함을 의미한다. 첫 번째 부스트링은 계획시점 이전의 상태를 고려하여 해석하여야 한다. 이전상태가 기동상태이면 지속시간만큼 기동을 하고 정지상태이면 정지시간만큼 정지를 한다. 지속시간부의 길이 n_i는 다음의 식으로 구한다.

$$n_i = \lceil \log_2(n_i^{\max}) \rceil$$

단,

$n_i^{\max}$: 발전기 i의 최소정지시간과 최소기동시간 중에서 큰 값

이와 같이 지속시간부의 길이를 결정하게 되면 최소정지시간이나 최소기동시간 이상의 모든 지속시간을 표현할 수 있다. 하나의 후보해는 모든 발전기에 대한 스트링의 집합으로 구성되어진다.

3. 유전알고리즘의 적용

유전알고리즘은 자연 선택(적자생존)과 자연 유전학의 메커니즘을 기반으로 만들어진 효율적인 탐색 방법이다(Goldberg, 1989). 유전 알고리즘은 하나의 해를 부호화 하여 만들어진 개체들의 집단인 해집단을 가지고, 매 세대마다 selection, crossover, mutation의 연산을 확률적으로 수행하여 새로운 해집단을 만들어간다. 해집단은 세대가 지남에 따라 우수한 해들로 수렴되어 간다. 유전알고리즘은 이처럼 해집단을 기반으로 확률적 연산을 통하여 탐색을 수행하기 때문에 전역적 탐색 능력이 뛰어나지만 긴 탐색시간을 요구한다.

발전기 기동정지 계획에 적용 시 적합도(fitness)에 따라 그 값의 비율에 따른 selection을 하지 않고 순위에 따라 적합도를 정규화를 하여 selection을 하도록 하였다. 발전기 기동정지 계획은 후보해에서 약간의 변화에 의해서도 평가값에 큰 변화가 생기는 매우 불규칙적인 해공간을 가지기 때문에 평가값이 좋지 못한 해를 통해서도 우수한 해를 이끌어낼 수 있다. 순위에 의한 정규화를 통해 탐색 초반부터 해집단이 일부 우수한 해들에 의해 빠르게 수렴되어 가는 것을 방지할 수 있고, 또한 해집단이 어느 정도 수렴되고 나면 전체 해집단의 적합도가 유사해져서 random selection이 발생하는 것을 방지하여 수렴후에도 보다 우수한 해로의 탐색을 수행할 수 있다.

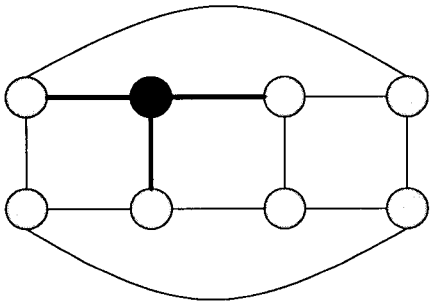
3.1 유전알고리즘의 병렬화

유전알고리즘은 해집단을 기초로 연산을 수행하지만 해집단의 각 개체들은 독립적으로 평가와 유전연산을 수행할 수 있기 때문에 그 자체적으로 병렬성을 내포하고 있다. 대규모 최적화 문제에서 유전알고리즘의 최대 단점인 탐색시간의 문제를 해결하기 위한 하나의 방법은 유전알고리즘을 병렬화하는 것이다. 따라서 본 논문에서도 유전알고리즘을 병렬화하여 보다 빠른 시간에 발전기 기동정지 계획을 수립하고자 한다.

3.1.1 병렬화 모델

유전알고리즘의 병렬화에 대한 연구는 70년대 병렬컴퓨터의 발달과 함께 이루어지기 시작했으며, 병렬 유전알고리즘의 모델은 전역 병렬 모델, 소집단 구조 모델, 세집단 구조 모델, 하이브리드 모델 4가지로 구분된다(Cant-Paz, 1997).

본 논문에서는 프로세서 16개로 구성된 MIMD 병렬컴퓨터를 이용하여 소집단 구조 모델로서 병렬 유전알고리즘을 구현하였다. 소집단 구조 모델은 각 소집단을 독자적으로 진화시키면서 서로 일부 해를 주기적으로 교환하여, 소집단간에 서로 준 독립성이 유지되고, 그에 따라 다양성 유지가 용이해져서 어느 한쪽으로 조기 수렴하는 것을 방지하는 효과가 있다. 따라서 보통의 유전알고리즘보다 탐색 속도면에서 빠를 뿐 아니라 더욱 우수한 해로 수렴할 가능성도 높아지게 된다.



<그림 3> 하이퍼큐브 연결 구조

3.1.2 연결 구조(topology)

병렬 유전알고리즘에서 프로세서간의 연결구조는 성능을 좌우하는 중요한 요소 중 하나이다. 일반적으로 연결구조가 조밀하면 프로세서간의 해의 교환이 과도하게 이루어짐에 따라 소집단 간의 다양성이 유지되지 못하여 해의 질이 떨어지게 된다. 반면 연결구조가 느슨하면 프로세서간의 다양성 유지에는 유리하나 우수한 해의 전달 속도가 저하된다는 문제가 있다. 본 논문에서는 하이퍼큐브의 형태로 구현하였다. <그림 3>에서 보는 바와 같이 하이퍼큐브는 모든 프로세서에서의 링크의 수가 $\log_2 n$(n은 프로세서의 개수)이 되도록 하는 연결구조(예를 들어 프로세서의 수가 8개이면 각 프로세서에서의 링크의 수는 3개가 된다.)로서 지나치게 조밀하지도 지나치게 느슨하지도 않은 적절한 연결성을 지니고 있다.

3.1.3 매개 변수 값 결정

이상의 기본 골격 하에서 실제 병렬 유전알고리즘의 탐색 성능에 영향을 미치는 중요요소들로 각 소집단의 규모, 적정 교환해의 수, 적정 교환 주기, 적정 교환 대상해 등을 들 수 있다. 이들 매개변수들의 적

정 값은 실제 다양한 실험과 분석을 통하여 탐색의 성능을 극대화할 수 있는 방향으로 결정하였다.

4. 타부 탐색의 적용

타부 탐색은 최적화 문제를 해결하기 위해 흔히 사용하는 또 다른 메타휴리스틱 기법이다. 보통의 지역적 탐색법처럼, 타부 탐색은 하나의 후보해에서 시작하여 주어진 종료조건을 만족할 때 까지 반복적으로 이웃해로 이동을 한다. 그러나 타부 탐색은 국지 최적해에 빠지는 것을 피하기 위한 전략들을 사용함으로써 지역적 탐색을 넘어선다. 하나의 해에서 이웃의 다른 해로 이동할 때, 비록 이전의 해보다 좋지 않더라도 이웃해 중 가장 좋은 해로 이동을 하여 국지 최적해를 벗어나기 위한 이동을 시도한다. 그러나 다음의 이동에서 가장 좋은 해들을 계속 선택하다 보면 지난번의 국지적 최적해로 다시 돌아올 확률이 높으므로, 상당수의 이동이 이루어지는 동안 지난번의 국지적 최적해로의 회귀를 방지하는 수단이 필요하다. 이를 위해서, 타부 탐색은 타부리스트라는 단기 메모리(short term memory)에 가장 최근 일정 횟수의 이동내역을 기록하고, 이 리스트에 기록된 해로 이동하는 것을 방지한다. 타부 리스트는 보통 FIFO (first-in-first-out)기반의 큐(queue)의 형태로 구현된다.

4.1 이웃해 생성

타부 탐색은 지역적 탐색법에 기반을 두고 있으므로 이웃해 생성 방법은 중요한 고려 요소 중 하나이다. 일반적으로 이웃해는 현재해의 일부분을 변경하여 생성한다. 주로 특정 위치의 값을 변화시키거나 두 개의 값을 서로 교환하는 방식을 취하고 있으나 발전기 기동정지 계획의 경우는 이러한 방식으로 후보해를 생성하게 되면 실제 복호화 된 계획에서는 현재해와 이웃해 사이에 아주 큰 변화가 나타날 수 있다. 지속시간부의 하위비트가 변할 때는 지속시간이 약간만 변하지만 상위 비트가 변환되게 되면 큰 폭으로 변하게 된다. 따라서 이웃해의 생성은 임의의 비트를 변환하도록 허용하지 않고, 현재해에서 지속시간부를 한시간 증가 또는 감소하는 해만 생성하도록 하였다. 부호부의 값 전환은 허용하였다. 때론 연속된 여러 시간에 대해서 기동정지를 변경함으로써 좋은 해를 찾아낼 수 있다. 그러나 지속시간부의 수정만 가지고는 그러한 해를 찾아내기가 힘들다. 부호부의 값 전환은 연속된 여러 시간에 대한 기동 혹은 정지상태를 반전한 해를 만들어낸다.

<그림 4>에서 (b)는 지속시간을 한 시간 증감하여 생성된 이웃해를 보여주고 있다. (a)의 첫 번째 부스트링의 경우 지속시간부는 1을 나타내고 있기 때문에 한 시간을 증감하면 2 또는 0이 될 수 있다. 따라서 (b)의 그림과 같은 이웃해가 생성된다. 나머지 부스트링에 대해서도 같은 방식으로 이웃해를 생성한다. (c)의 그림은 두 번째, 네 번째 부스트링에서 부호부의 값을 바꾸어 생성된 이웃해를 보여주고 있다. 다른 부스트링에 대해서도 같은 방식으로 이웃해를 생성한다.

4.2 탐색의 다각화

현재해로부터 위의 방법에 의해 이웃해를 생성하고 나면 가장 좋은 해로 이웃해가 이동을 한다. 타부 탐색에서는 타부리스트를 관리하여 이미 방문한 해로의 재 방문을 막고 탐색의 다각화를 유도한다. 본 연구에서는 타부리스트로서 inc_list와 dec_list를 따로 두어 inc_list에는 지속시간을 한 시간 증가시키거나 부호부를 0에서 1로 전환하여 생성된 경우를 삽입하여 추후에 지속시간을 한 시간 감소시키거나 부호부를 1에서 0으로 전환하여 원래의 해로 돌아가는 것을 방지하도록 하고, dec_list는 지속시간을 한 시간 감소시키거나 부호부를 1에서 0으로 전환하여 생성된 경우를 삽입하여 추후에 지속시간을 한 시간 증가시키거나 부호부를 0에서 1로 전환하여 원래의 해로 돌아가는 것을 방지하였다.

타부리스트만 이용해서는 타부 탐색이 국지적 최적해를 벗어나 전력적 탐색을 하도록 이끌기 어렵다. 따라서 보다 전역적 탐색을 이끌기 위한 다

<그림 4> 이웃해 생성

각화 방안이 필요하다. 이를 위하여 각 발전기와 시간대에 대한 빈도수 기반 메모리를 두어 각 발전기와 각 시간대에 대해서 계획 변경 횟수를 기록하여 둔다. 빈도수 기반 메모리의 값은 매 이동마다 계획의 변경이 발생한 발전기와 시간대에 대해서 빈도 값을 증가시켜준다. 탐색이 장시간 동안 진전이 없으면 다각화 전략을 적용하는데 변경 빈도수가 높은 발전기나 시간대의 계획을 수정하여 만들어진 이웃해에 대해서 그 빈도수에 비례하는 벌점을 부여함으로써 상대적으로 많이 탐색하지 않은 공간으로서 탐색을 유도한다.

4.3 비용 함수의 효율적 계산

발전기 기동정지 계획에 의해 산출되는 총 발전비용은 연료비용, 기동비용, 정지비용으로 구성된다. 여기서 기동비용과 정지비용은 발전기마다 주어진 매개변수 값에 따라 발전기의 기동 정지 여부에 따라 결정되는 비용으로 타 발전기와는 무관하게 계산이 가능하다. 그러나 연료비용은 같은 시간에 기동되는 모든 발전기들을 고려하여 각 발전기의 출력이 결정되고 그 출력에 따라 비용이 계산된다. 따라서 연료비용의 경우 하나의 발전기의 계획이 바뀌게 되면 그 시간대의 모든 발전기에 대한 비용이 변하게 된다. 기동비용과 정지비용은 발전기 별로 계산하고, 연료비용은 시간별로 계산한다.

본 논문에서 평가를 위한 비용함수는 연료비용, 기동비용, 정지비용의 합에 최소시간 제약이외의 제약을 어길 시 발생하는 벌점을 합산하여 계산하였다. 타부 탐색 도중 이웃해를 평가하기 위해서 다시 이 모든 값을 처음부터 다시 계산하는 것은 시간적 낭비이다. 따라서 현재해와 이웃해 사이의 변화 부분에 대해서만 재평가를 하는 것이 효율적이다. 발전기 기동정지 계획에서 이웃해를 평가하기 위해서는 기동비용과 정지비용은 변화가 나타난 발전기에 대해서 재 계산을 하면 되고, 연료비용은 변화가 나타난 시간들에 대해서 재 계산을 하면 된다. <그림 5>에서 보는 바와 같이 변화가 나타난 위치에 대해서 가로방향으로 기동비용과 정지비용을 재 계산하고 세로방향으로 연료비용을 재 계산한다. 그러나 본 연구의 실험 시 사용한 데이터는 일부 발전기들의 특성에 의하여 연료비용이 계획의 변화가 나타난 시간의 전후 한시간까지 달라지게 된다. 따라서 전후 1시간에 대해서도 연료비용을 재 계산하였다.

4.4 타부 탐색의 병렬화

타부 탐색은 지역적 탐색을 기반으로 하여 빠른 탐색을 하면서 여러 가지 전략을 사용하여 탐색을 다각화함으로써 전역적 탐색을 할 수 있다. 그러나 대규모 최적화 문제는 복잡하면서 방대한 탐색 공간을 가지기 때문에 일반적인 타부 탐색으로는 적절한 시간 안에 탐색을 끝내기 어렵고 충분히 전역적 탐색을 하기도 힘들다. 따라서 본 연구에서는 타부 탐색도 병렬화를 통하여 이러한 문제점을 해결하고자 하였다.

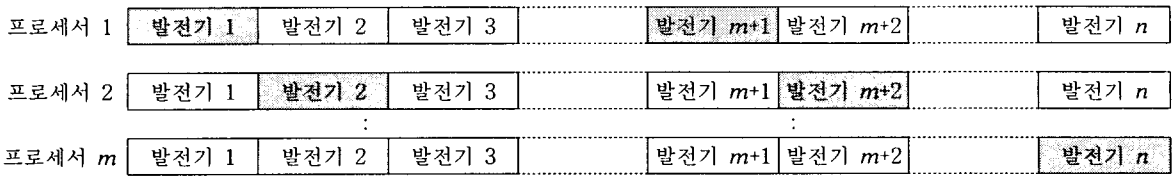
4.4.1 병렬화 모델

타부 탐색의 병렬화 모델은 크게 전역 모델과 분산 모델로 구분할 수 있다. 전역 모델은 하나의 관리프로세서(master)와 여러 개의 작업프로세서(slave)로 구성되며, 관리프로세서가 탐색의 전 과정을 관리하고 이웃해들의 생성 및 평가와 같이 계산량이 많은 작업은 여러 작업프로세서가 나누어 처리하도록 하는 방법이다. 이 모델은 구현이 쉬운 반면 프로세서간의 동기화가 엄격히 유지되어야 한다는 단점이 있으며(Crainic *et al.* 1995), 병렬화를 통한 계산시간 단축 효과 외 다른 이점을 얻기 힘들다.

분산 모델에서는 여러 프로세서가 각기 별도의 탐색영역에 대해 독립적인 타부 탐색을 수행하면서 우수한 해를 서로 교환해 가면서 탐색을 수행한다. 이 모델은 여러 프로세서들로부터 다양한 탐색영역에서 구한 해들을 동시에 받아보고 분석하여 탐색이 이루어지므로 전역적인 탐색이 효과적으로 이루어진다. 실제로 분산 모델의 경우 계산 시간의 단축뿐만 아니라 보다 우수한 해를 찾게 되는 효과를 얻은 경우가 보고된 바 있다(Rego and Roucairol, 1995, Babeau *et al.* 1997). 이 외에도 문제를 분해(problem decomposition)하여 부 문제(subproblem)를 병렬로 풀게 하는 접근 방안 등이 있다(Toulouse *et al.* 1996). 본 논문에서는 타부 탐색의 병렬화 모델로서 분산 모델을 채택하였다.

1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
						:						:							:				
1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0

<그림 5> Incremental Evaluation



<그림 6> 탐색 공간의 분할

4.4.2 탐색 영역의 배분

분산 모델의 적용 시 가장 중요한 요소는 프로세서간의 탐색의 다양화를 유지하는 것이다. 프로세서간의 다양화 유지를 위해서는 다음과 같은 전략을 사용할 수 있다. 첫 번째 방법은 프로세서마다 탐색 전략을 달리하는 것이다. 즉, 타부리스트의 크기, 이웃해 생성방법, 집중화·다각화 전략 등을 프로세서마다 달리하여 서로 다른 방향으로 탐색을 유도하는 방식이다. 두 번째 방법은 직접적으로 탐색공간을 분할하는 방식이다. 즉, 프로세서마다 이웃해 생성범위를 제한하여 특정 영역만 탐색을 하도록 하는 방식이다. 본 논문에서의 다양화 유지 전략으로는 후자의 방법을 이용하였다.

타부 탐색의 병렬화 시 탐색을 수행하는데 소요되는 시간을 프로세서 수에 비례해서 단축하기 위해서는 병렬화하지 않았을 때 적정한 것으로 판단된 생성 이웃해 수를 프로세서의 수로 나눈 만큼 각 프로세서에서 이웃해를 생성하도록 하는 것을 생각할 수 있다. 그러나 각 프로세서에서 너무 적은 수의 이웃해만 생성하게 되면 대상 탐색 영역을 세밀하게 탐색하지 못할 우려가 있다. 대신, 프로세서마다 탐색공간을 제한을 하게 되면 적은 수의 이웃해로도 충분한 탐색을 할 수가 있다. 발전기 기동정지 계획에서의 탐색 영역의 분할은 발전기별 분할과 시간대별 분할을 생각할 수 있다. 발전기 기동정지 계획에서 부스트링들을 수정하여 이웃해를 생성하는데 각 부스트링이 나타내는 시간대는 유동적이기 때문에 시간대별로 탐색 공간을 분할하는 것은 힘들다. 따라서 탐색 영역의 분할은 발전기별로 하였다. <그림 6>은 발전기별로 각 프로세서의 탐색영역을 분할한 예이다. 이처럼 각 프로세서에서의 이웃해 생성 영역을 서로 분할하게 되면, 전체영역에 대해서 생성했던 이웃해를 일부 영역에 대해서만 생성하므로 그 영역에 대해서는 보다 세밀한 탐색이 가능해진다. 그리고 전체 프로세서의 관점에서 보면 각 프로세서가 서로 다른 영역을 탐색하므로 탐색의 다양성 유지가 용이하게 된다.

본 논문에서는 탐색공간의 분할을 위해서 전체 발전기를 일정등분으로 나누어 각 프로세서에 할당하였다. 이때 특정 종류의 발전기들이 한 등분으로 들어가지 않고 각 등분에 고루 분포하도록 분배를 하였다. 이와 같은 방법으로 각 프로세서는 특정 영역에 대해서 집중적으로 탐색을 하면서 일정주기마다 이웃 프로세서로부터 우수한 해를 받아 들여 전체적으로는 다양성을 유지하면서 탐색을 하도록 하였다.

4.4.3 연결구조

병렬 타부 탐색에서의 프로세서들간의 연결구조는 병렬 유전알고리즘과 마찬가지로 하이퍼큐브의 형태로 구현하였다.

5. 실험

각 방법의 성능 평가를 위하여 실제 발전기 152대의 일간 계획을 대상으로 실험을 하였다. 실험환경으로는 Parsytec사의 CC16시스템을 사용하였다. CC16시스템은 PowerPC604 100MHz CPU 16개로 구성된 전형적인 MIMD(multiple instruction streams multiple data streams)시스템이다.

<표 1> 유전알고리즘과 타부 탐색의 성능 비교

	해의 질(10^{10}원)	탐색 소요 시간
우선 순위법	1.0483	< 1초
유전알고리즘	1.0404	62시간
타부 탐색	1.0488	2시간 16분

우선 유전알고리즘과 타부 탐색의 결과를 우선순위법과 비교하였다. 우선순위법은 실제 현장에서 발전기 기동정지 계획을 위해 사용하는 방법으로 각 발전기의 연료효율에 따라 효율이 높은 발전기부터 기동을 시키도록 계획하는 휴리스틱 기법이다. 유전알고리즘과 타부 탐색의 탐색 능력에 대한 공정한 비교를 위해서 총 이웃해 평가횟수는 동일하도록 하였다. <표 1>에서 보는 바와 같이 유전알고리즘은 해의 질에 있어서 우수한 해를 찾아내지만 탐색 소요시간이 매우 긴 것을 알 수 있다. 그러나 타부 탐색은 비록 탐색 소요시간에 있어서 유전알고리즘보다는 빠르지만 그래도 많은 시간이 소요되고 해의 질 또한 우선순위법과 유사한 수준에 머물고 있다. 이는 타부 탐색이 유전알고리즘만큼 충분히 전역적 탐색을 하지 못하기 때문이다.

<표 2> 병렬화를 통한 성능 비교

		해의 질 (10^{10}원)	탐색 소요 시간
우선 순위법		1.0483	< 1초
유 전 알고리즘	직렬	1.0404	62시간
	병렬	1.0387	7시간 30분
타부 탐색	직렬	1.0488	2시간 16분
	병렬	1.0356	30분

다음으로 각 방법의 병렬화를 통한 성능의 변화를 살펴보았다. <표 2>를 보면 유전알고리즘과 타부 탐색 모두 병렬화를 통하여 탐색 시간의 단축과 함께 해의 질 또한 개선된 것을 볼 수 있다. 두 알고리즘 모두 병렬화를 통하여 탐색의 다각화 능력이 강화되었음을 보여주는 것이다. 특히 타부 탐색은 병렬화한 결과 짧은 시간안에 병렬 유전알고리즘보다 우수한 해를 찾아내고 있다. 이는 타부 탐색 자체의 빠른 탐색능력에 병렬화를 통한 프로세서간의 다양화가 결합하여 이와 같은 성능을 보이는 것이다. 또한 프로세서 별로 탐색 공간을 분할함으로써 탐색의 집중화의 효과도 가지고 있다.

6. 결론

본 논문은 대규모 최적화 문제를 위하여 대표적인 메타휴리스틱 기법인 유전알고리즘과 타부 탐색을 병렬화하여 성능의 향상을 가져올 수 있음을 보여주었다. 대규모 최적화 문제 중 하나인 발전기 기동정지 계획에 적용하기 위하여 유전알고리즘과 타부 탐색의 적용방안을 연구하고 또한 각 알고리즘을 병렬화하여 병렬화를 통하여 성능이 향상됨을 실험을 통하여 보여주었다. 또한 타부 탐색의 경우는 전역적 탐색의 한계를 벗어나서 빠른 시간에 우수한 해를 찾아내는 것을 보였다.

각 탐색방법마다 그마다의 장단점이 있다. 따라서 여러 탐색방법들의 장점을 결합한 하이브리드 탐색법에 대한 많은 연구가 있었다. 향후 과제로서 타부 탐색의 지역적 빠른 수렴성과 유전알고리즘의 전역적 탐색능력을 결합한 하이브리드 탐색을 병렬화하여 구현함으로써 그 성능을 검증해보고자 한다.

참고문헌

Badeau, P., M. Gendreau, F. Guertin, J. Y. Potvin and E. Taillard, "A parallel tabu search heuristic for the vehicle routing problem with time window", Transportation Research-C5, pp. 109-122, 1997.

Cant-Paz, E., "A Survey of Parallel Genetic Algorithms", IlliGal Report No. 97003, 1997.

Crainic, T. G., M. Toulouse and M. Gendreau, "Synchronous Tabu Search Parallelization Strategies for Multicommodity Location-Allocation with Balancing Requirements", OR Spectrum, Vol. 17, 1995.

Glover, F. and M. Laguna, *Tabu search* , Kluwer Academic Publishers, 1997.

Goldberg, D. E., *Genetic Algorithms in Search, Optimization & Machine Learning* , Addison-Wesley, 1989.

Mun, K. J., H. S. Kim, J. H. Park, T. W. Park, S. H. Park, K. R. Ryu and S. H. Chung, "A Parallel Genetic Algorithm for the Unit Commitment Problem", Proceedings of the International Conference on Intelligent System Application to Power Systems(ISAP-97), pp. 188-193, Seoul, Korea, 1997.

Rego, C. and C. Roucairol, "A Parallel Tabu Search Algorithm Using Ejection Chains for the VRP", Proceedings of the Metaheuristics International Conference, Breckenridge, Colorado, pp. 253-259, 1995.

Toulouse, M., T. Crainic and M. Gendreau, "Communication Issues in Designing Cooperative Multi-Thread Parallel Searches", *Meta-Heuristics: Theory and Applications* , Kluwer Academic Publishers, Norwell MA, pp. 501-522, 1996.