최대 시스템 신뢰도를 위한 최적 중복 설계: 유전알고리즘에 의한 접근

김재윤*† · 신경석**
*전남대학교 경영학부 · **전남대학교 산업공학과

Optimum redundancy design for maximum system reliability: A genetic algorithm approach

Jae Yun Kim* · Kyoung Seok Shin**
*School of Business Administration, Chonnam National University
**Department of Industrial Engineering, Chonnam National University

Key Words : reliability, redundancy, genetic algorithm

Abstract

Generally, parallel redundancy is used to improve reliability in many systems. However, redundancy increases system cost, weight, volume, power, etc. Due to limited availability of these resources, the system designer has to maximize reliability subject to various constraints or minimize resources while satisfying the minimum requirement of system reliability. This paper presents GAs (Genetic Algorithms) to solve redundancy allocation in series-parallel systems. To apply the GAs to this problem, we propose a genetic representation, the method for initial population construction, evaluation and genetic operators. Especially, to improve the performance of GAs, we develop heuristic operators (heuristic crossover, heuristic mutation) using the reliability-resource information of the chromosome. Experiments are carried out to evaluate the performance of the proposed algorithm. The performance comparison between the proposed algorithm and a pervious method shows that our approach is more efficient.

1. 서 론

오늘날 생산되는 제품, 특히 전자통신 시스템은 관련기술의 급속한 발달로 인해 그 기능과 구조가 점차 복잡화되면서 고장 발생의 기회는 많아지고 있는 반면 사용자는 보다 고품질, 고신뢰도의 시스템을 요구

† 교신저자 jaeyun@chonnam.ac.kr

하고 있다. 이로 인하여, 현재 거의 모든 분야에서 신뢰도에 대한 고려는 점점 더 큰 역할을 하고 있다. 신뢰도(reliability)는 주어진 기간 동안 주어진 조건하에서, 사용자의 요구조건에 부합하는 기능을 성공적으로 수행할 수 있는 확률을 말한다. 여기서 주어진 기간은 정상적인 동작이 요구되는 기간, 주어진 조건은 기계적, 열적, 전기적인 조건을 포함하는 물리적인 환경을 뜻하며, 기능은 정상적인 동작 및 비정상적인 동작의 정의를 포함한 것을 말한다.

시스템의 신뢰도는 1) 시스템의 구성 부품들의 신뢰도를 증가시키거나, 2) 신뢰도가 낮은 부분에 중복(redundancy) 설계를 사용하거나, 3) 고장이 발생했을 때 다른 부품으로 그 기능이 전환될 수 있는 대기(standby) 구조를 설계하거나, 또는 4) 잘 계획된 유지/보수 일정을 행하는 것 등에 의해 개선될 수 있다(Kuo and Rajendra, 2000). 본 연구에서는 이들 중에서 주어진 자원(비용, 부피, 중량, 전력 등)의 제약이 존재하고, 시스템의 각 부분에 추가될 수 있는 다양한 종류의 부품들이 있을 때, 이들 부품들의 최적 할당을 통해 시스템 신뢰도를 최대로 하는 최적 중복수를 결정하는 문제(ORDP: Optimum Redundancy Design Problem)를 다룬다. 복잡하고 대형화된 구조를 갖는 최신 시스템에서는 ORDP에서와 같이 한정된 자원을 이용하여 시스템의 요구기능을 만족하는 동시에 신뢰도를 높일 수 있는 노력들이 중요하게 요구된다.

본 연구에서 다루는 ORDP는 조합최적화(combinatorial optimization) 문제이다(Coit and Smith, 1996). 따라서, 이를 효과적으로 해결할 수 있는 방법론 개발을 위하여 많은 연구들이 수행되었다. ORDP 해결을 위한 접근 방법들은 크게 최적화 기법(exact algorithm), 발견적 기법(heuristic algorithm), 메타휴리스틱(metaheuristics) 등 3가지로 분류될 수 있다. 최적화 기법으로는 동적계획법(dynamic programming), 분지한계법(branch and bound method), 열거법(enumeration method) 등을 이용한 연구들이 수행되었다. 이러한 접근 방법은 변수의 수가 증가하면 요구되는 계산량이 지수적으로 증가하므로, 대형 시스템 또는 여러 제약을 갖는 문제를 해결하지 못한다는 단점을 갖는다(Lee *et al.*, 2002). 발견적 기법에 의한 해는 최적해를 보장하지 못한다. 또한, 일반적으로 ORDP의 결정변수들은 정수값을 갖는데 발견적 기법으로 구한 해는 정수가 아니므로, 정수값으로의 변환이 요구된다. 이러한 이유로 최근에는 유전알고리즘(genetic algorithm)이나 시뮬레이티드 어닐링(simulated annealing) 등과 같은 메타휴리스틱에 의한 연구가 수행되고 있다. 메타휴리스틱은 비교적 적은 시간에 수용가능한 근사 최적해를 찾으며, 문제 크기가 크더라도 효과적으로 해결할 수 있다는 장점을 갖는다(Michalewicz, 1999). ORDP에 대한 구체적인 연구현황들은 Kuo and Rajendra(2000)에 잘 정리되어 있으므로, 이를 참조할 수 있다.

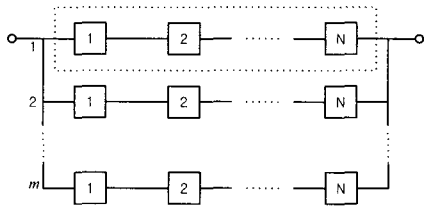
본 연구에서는 ORDP의 해결을 위한 방법론으로 유전알고리즘을 채택한다. 유전알고리즘으로 ORDP를 해결하고자 할 때 중요하게 다루어져야할 사항은 문제의 잠재해를 적절히 표현하는 방법과 문제에서 고려하는 여러 제약들을 효과적으로 처리하는 방법을 개발하는 것이다(Coit and Smith, 1996; Gen and Cheng, 1997). 본 연구에서는 이에 대한 방법과 함께, 알고리즘의 탐

색 성능을 향상시키기 위해 개체가 갖는 고유 정보를 이용하는 발견적 유전연산자를 새롭게 제안한다.

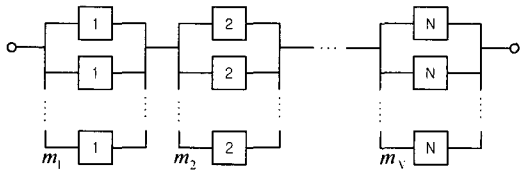
2. 문제정의

중복설계란 동일한 기능을 가지는 여분의 수단을 준비하여 신뢰도를 높게 하는 것이다. 중복설계는 구성품이 여분의 구성 요소를 가지고 있기 때문에 그 구성품의 고장이 반드시 전체의 고장을 일으키지 않는다는 특징을 갖는다. 고도의 신뢰도가 요구되는 특정 부품에 여분의 구성품을 더 설치함으로써 해당 부분의 신뢰도를 높일 수 있다. 중복설계에는 처음부터 여분의 구성품이 주구성품과 함께 작동하는 병렬 중복(parallel redunduncy) 설계와 여분의 구성품은 대기상태에 있다가 주구성품이 고장나면 그 기능을 인계받아 계속 수행하는 대기 중복(standby redundancy) 설계가 있다. 이중에서 중복설계의 가장 대표적인 모델은 병렬구조이다. 시스템의 병렬 구조는 크게 부분별 중복을 갖는 형태(parallel-series 구조)와 부품별 중복을 갖는 형태(series-parallel 구조)의 두 가지로 분류될 수 있다. 부분별 중복 구조는 <그림 1> (a)와 같이 구성품들로 이루어진 어느 한 부분이 하나 또는 그 이상의 중복으로 구성된 구조를 말한다. 반면, 부품별 중복구조는 <그림 1> (b)와 같이 시스템을 이루는 각 구성품이 중복으로 이루어진 구조를 말한다.

흔히 시스템에서 중복을 고려하는 단위를 '단계(stage)'라는 용어로 표현한다. 즉, 단계는 서브시스템(subsystem)이나 기능블럭(functional block)과 같이 신뢰성 구조를 구성하는 기본 단위를 말한다. 예를 들어, <그림 1>의 시스템은 N개의 단계를 갖는 구조이다. 만약, 시스템에서 단계의 수를 N, 중복으로 추가되는 단위 또는 시스템의 수를 m이라 하면, m을 증가시킬수록 시스템의 신뢰도는 증가한다. 그리고 부품별 중복구조의 신뢰도가 부분별 중복구조의 신뢰도보다 더 높다.



(a) 부분별 중복 구조



(b) 부품별 중복 구조

<그림 1> 시스템의 중복 구조

각 단계에 중복으로 추가될 수 있는 구성품들은 동일한 기능을 수행해야 한다. 따라서 그 구성품들은 동일한 것일 수도 있고, 기능은 같지만 신뢰도, 비용, 중량 및 부피가 모두 다를 수 있다. 특히, 대부분 표준화가 되어 있는 부품들은 여러 회사에서 제조되고 있다. 이들 부품들은 동일한 기능을 수행하지만 제조 회사마다 신뢰도, 비용, 중량 및 부피가 모두 상이하다.

이제 본 연구에서 다루는 ORDP의 수리모형에 관하여 알아보자. ORDP는 신뢰도 요구수준의 제약을 갖고, 신뢰도 개선을 위

해 소요되는 자원의 총합을 최소화하는 문제로 모형화 될 수 있다. 반대로, 자원의 제약을 갖고 신뢰도를 최대로 하는 문제로도 변환될 수 있다. 본 연구에서는 일반적인 문제로써 중복으로 추가될 수 있는 부품들은 동일한 기능을 갖지만 비용, 중량, 부피, 전력 등이 서로 다른 여러 종류가 존재하는 경우, 시스템 신뢰도를 최대로 하기 위해 어떤 종류의 부품을 얼마만큼 추가할 것인가를 결정하는 문제를 다룬다. 결국 이 문제를 해결함으로써 최적 중복수를 결정하여 각 단계의 최적 신뢰도값을 결정할 수 있게 된다. 이때, 고려하는 시스템의 중복구조는 <그림 1> (b)와 같은 부품별 중복구조이다. ORDP는 <그림 2>와 같이 모형화될 수 있으며, 수리모형에 사용되는 기호들의 정의는 다음과 같다.

식 (1)은 목적함수로 시스템을 부품별 중복구조로 설계할 때, 각 단계에 추가되는 부품에 따른 시스템 신뢰도를 나타낸다. 식 (2)는 각 부품이 추가될 때의 시스템 비용, 무게, 부피, 전력 등 자원 제약을 의미한다. 이 제약식은 고려하는 자원이나 기술적인 제약에 따라 추가될 수 있으며 선형이 아닌 비선형으로도 모형화될 수 있다. 식 (3)은 각 단계에 추가될 수 있는 부품 개수의 하한과 상한을 나타내는 제약과 결정변수의 정수제약을 나타낸다.

R_s : 시스템의 신뢰도
R_i : 단계 i의 신뢰도, i = 1, 2, ···, N
N : 단계의 수
n_i : 단계 i에서 사용 가능한 부품의 종류
r_{ij} : 단계 i에 추가되는 부품 j의 신뢰도
g_{lij} : 단계 i에 추가되는 단위부품 j의 자원 l, l = 1, 2, ···, L
b_l : 가용한 자원 l의 양, l = 1, 2, ···, L
x_{ij} : 단계 i에 사용되는 부품 j의 수
u_i : 단계 i에 추가될 수 있는 부품들의 상한

$$\text{max.} \quad R_s = \prod_{i=1}^{N} R_i = \prod_{i=1}^{N}\left\{1 - \prod_{j=1}^{n_i}(1-r_{ij})^{x_{ij}}\right\} \tag{1}$$

subject to

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} g_{lij} \cdot x_{ij} \leqq b_l,\ l = 1,2,\cdots,L \tag{2}$$

$$1 \leqq \sum_{j=1}^{n_i} x_{ij} \leqq u_i,\ i = 1,2,\cdots,N,\ x_{ij} \geq 0\text{인 정수} \tag{3}$$

<그림 2> ORDP의 수리모형

3. ORDP를 위한 유전알고리즘

유전알고리즘은 자연계의 적자생존과 유전법칙의 생물학적 진화과정에 기초한 일종의 확률적 탐색기법이다(Holland, 1975). 유전알고리즘에서는 문제의 잠재해를 개체(individual)로 표현하고, 여러 개체로 이루어진 모집단(population)을 운영하여 해공간

을 탐색한다. 모집단의 운영은 다양한 해공간의 탐색을 가능하게 한다. 모집단의 크기는 흔히 세대가 진행되는 동안 일정하게 유지한다. 유전알고리즘은 매 세대마다 개체의 적응도(fitness)를 평가하고, 적자생존의 원리에 의해 즉, 적응도에 따라 모집단의 개체들을 확률적으로 선별(selection)하여 다음 세대의 자손을 생산하는데 참여시킨다. 새로운 자손의 생산은 선별된 개체들에 대하여 교차(crossover) 및 돌연변이(mutation)와 같은 유전 연산자에 의해 이루어진다. 이러한 일련의 과정을 종료조건이 만족될 때까지 반복한다.

유전알고리즘은 다른 탐색기법들과 비교하여 다음과 같은 장점들을 갖는다. 첫째, 유전알고리즘은 하나의 점(해)을 이동하면서 탐색하지 않고 모집단을 이용하여 여러 점을 동시에 변화시키면서 해공간을 탐색한다. 이로써 해가 부분 최적(local optimal)에 조기 수렴되는 문제를 줄일 수 있다. 둘째, 유전알고리즘은 목적함수 값만을 사용하고 미분값이나 그 밖의 다른 방법을 필요로 하지 않는다. 따라서 최적화 문제에서 목적함수나 제약식의 변화에 쉽게 적응할 수 있다. 셋째, 모집단 내의 좋은 해들이 갖는 공통적인 정보들을 추출하여 이용할 수 있기 때문에 효율적인 해의 탐색이 가능하다. 즉, 유전알고리즘은 다양한 해공간의 탐색(exploration)과 해에 포함된 정보의 효율적인 이용(exploitation)이 적절히 조화된 탐색기법이라고 할 수 있다(김여근외, 1997).

3.1 개체표현

유전알고리즘을 주어진 문제에 적용하기 위해서는 먼저 문제의 특성을 잘 반영할 수 있도록 해를 개체로 표현해야 한다. 개체표현은 유전연산과 관련되어 알고리즘의 성능에 크게 영향을 준다. 따라서, 개체의 표현은 자연스러워야 하며, 해의 정보가 유전연산자에 의해 쉽게 추출되어 자손에 전파될 수 있도록 중요한 정보가 개체에 명확히 표현되어야 한다.

본 연구에서 개체 표현은 각 단계를 유전 인자(gene)로 두고, 인자는 각 단계에서 가용할 수 있는 복수개의 부품종류로써 이루어진다. 설명을 위해 다음과 같이, 단계의 수 N=3, 부품의 종류는 각각 n_1=3, n_2=2, n_3=4, 그리고 u_i=5, i=1,2,3인 문제로써 예를 든다. 각 부품은 서로 다른 신뢰도, 그리고 비용, 무게, 부피의 자원을 가지며 그에 대한 상세 정보는 <표 1>에 주어져 있다.

이 문제에 대한 하나의 해는 <그림 3> (a)와 같은 개체로 표현될 수 있다. 그림에서 개체는 단계를 구분하는 3개의 그룹으로 이루어져 있다. ( )로 표시되는 이 그룹은 순서대로 단계 1, 단계 2, 단계 3을 각각 나타낸다. 첫번째 ( )의 첫번째 인자값 2는 단계 1에 첫번째 종류의 부품 2개를 중복으로 추가한다는 의미이다. 세번째 ( )의 세번째 인자값과 같이 0이면 단계 3에서 세번째 종류의 부품은 사용하지 않음을 뜻한다. 따라서 각 단계에서 추가되는 중복의 수는 각 단계에 해당하는 인자값들의 합이 된다. 예를 들어, 단계 1에는 4개(=2+1+1)의 부품이 병렬로 구성됨을 의미한다. <그림 3> (a)의 개체 표현은 <그림 3> (b)와 같은 시스템을 구성하는 것으로 해석된다.

이러한 표현방법은 무엇보다도 개체로 표현하는 부호화(encoding)와 부호화된 개체를 가능해로 나타내는 해석(decoding)이 용이하다. 목적함수가 각 단계에 중복으로

추가되는 부품들의 신뢰도와 개수의 함수이므로, 부품의 종류와 개수를 직접 나타내는 이와 같은 표현은 해를 평가하는 데 소요되는 시간을 줄일 수 있다. 그리고 개체가 단계별로 구분되며, 각 단계를 하나의 유전인자처럼 다루어 유전연산(교차, 돌연변이)이 이루어진다. 따라서 각 단계가 가지고 있는 좋은 정보를 유지하고 효율적으로 자손에 전파할 수 있다. 이 표현이 갖는 또 하나의 장점은 개체의 발견적 정보를 이용하는데 유리하다. 이 발견적 정보로써 좋은 해로의 탐색을 유도하여 알고리즘의 탐색 성능을 향상시킬 수 있다.

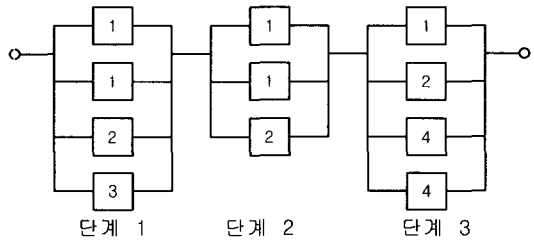
<표 1> 예제 문제의 상세 정보

자원	단계1 부품 종류			단계2 부품 종류		단계3 부품 종류				가용한 자원
	1	2	3	1	2	1	2	3	4	
비용	5	4	9	7	7	5	6	9	4	50
무게	3.2	3.6	2.4	2.8	3.2	3.2	3.6	2.4	2.8	20
부피	6	12	30	3	15	15	12	24	24	150
부품신뢰도	0.93	0.91	0.95	0.85	0.87	0.90	0.91	0.95	0.93	

p1: ((2, 1, 1) (2, 1) (1, 1, 0, 2))

단계1 단계 2 단계 3

(a) 개체 표현의 예



(b) (a)에서 보인 개체의 시스템 구조

<그림 3> ORDP를 위한 개체 표현 및 해석

3.2 초기 모집단

유전알고리즘에서는 복수개의 개체들로 이루어진 모집단을 운용한다. 따라서 알고리즘을 수행하기 위해 먼저 초기 모집단이 생성되어야 한다. 초기 모집단을 생성하는 방법은 임의 생성 방법과 문제의 특성을 이용한 발견적 기법을 사용하는 방법이 있다. 본 연구에서 초기 모집단은 각 단계에 중복으로 추가될 수 있는 부품 개수의 상한을 초과하지 않는 범위 내에서 즉, <그림 2>의 식 (3)을 만족하면서 임의로 부품의 종류와 개수를 선택하여 인자에 부여하는 임의생성 방법을 사용하였다. 이 때 각 단계에 추가되는 여러 부품 중 어떤 종류의 부품이 얼마만큼 선택할 것인지를 결정해야 한다. 따라서, 초기 모집단의 개체는 다음과 같은 절차를 통해 모집단의 크기만큼 반복 생산한다. 아래와 같은 방법으로 생성된 개체는 항상 식 (3)을 만족하게 된다.

step 1: i = 1로 둔다.

step 2: 단계 i에 대응되는 각 인자들에 대해 범위 [0, 1]을 갖는 n_i개의 난수 a_j, j = 1, 2, ⋯, n_i를 발생시킨다.

step 3: 발생된 난수 a_j로써 다음 식에 의해 각 인자(부품 종류)가 할당될 수 있는 비율을 결정한다.

$$m_j = \left[\frac{a_j}{\sum_{j=1}^{n_i} a_j} \times u_i \right]^+,$$

$[x]^+$는 x에 가장 가까운 정수

step 4: j = 1부터 n_i까지 차례로 [0, m_j]의 정수 난수를 발생하여 그 값을 x_{ij}로 한다. 단, $\sum_{j=1}^{n_i} x_{ij} > u_i$이면 x_{ij} = 0으로 둔다.

step 5: i = i+1, i>N이면 종료하고, 그렇지 않으면 step 2로 간다.

3.3 적응도평가와 선별

적응도는 개체의 생존능력을 나타낸다. 최적화문제에서 적응도는 목적함수에 의해 측정되며, 적응도 평가함수로는 최적화문제의 목적함수 자체를 흔히 사용한다. 그러나 앞에서 언급한 임의 생성 방법에 의해 생성한 초기 모집단은 비가능 개체를 포함할 수 있다. 즉, 초기 모집단을 구성하는 개체들은 식 (3)을 만족할 수 있으나, 식 (2)를 만족하는 것은 보장하지 못한다. 유전알고리즘에서 비가능 개체를 다루기 위해 흔히 벌금함수(penalty function)를 사용한다. 본 연구에서도 이와 같은 방법을 채택하여 사용한다. 이 방법은 개체의 각 제약 조건에 대한 위반정도를 측정하고, 개체의 적응도 평가시 이를 반영하여 탐색방향을 조정하는 방법이다. 가능개체만으로 모집단을 운용하면 매우 협소한 영역만을 탐색하게 되어 효율적이지 못하고, 유전알고리즘의 특징인 해가 갖는 정보의 효율적인 이용을 방해할 수도 있다. 따라서 모집단의 가능 개체와 비가능 개체가 상호 조화되어 해 영역을 탐색하는 것이 바람직하다(Michalewicz, 1999).

ORDP는 배낭(knapsack) 문제와 유사하다. 즉, 추가되는 부품이 많을수록 시스템 신뢰도는 커지고, 소요되는 자원 역시 증가한다. 따라서 위반정도와 목적함수(시스템 신뢰도) 값을 고려하여 벌금함수를 설계하는 것이 바람직하므로, 제안한 알고리즘에서는 배낭 문제에 흔히 이용되는 벌금함수를 사용하였다(Olsen, 1994). 개체의 위반정도는 허용된 자원의 최대량과 실제 개체가 갖는 자원 소요량과의 거리에 의해 구해진다.

실제로, 알고리즘에서 개체의 적응도는 다음의 과정을 통하여 평가된다. 개체의 자원 l에 대한 정규화된 소요량이 G_l, l = 1, 2, …, L이라 하고, 각 제약에 대한 벌금을 P_l이라 하자. 그러면 벌금 P_l은 식 (4)와 같이 부과된다. 그러면 개체의 적응도(F_{ind})는 목적함수 값에 벌금을 반영해 줌으로써 구할 수 있다. 즉, 개체의 적응도는 식 (5)와 같다. 식 (5)에서 R은 제약을 무시하고 구한 개체의 신뢰도이다.

$$P_l = \begin{cases} 1, & G_l \le 1, \\ \dfrac{1}{G_l}, & otherwise. \end{cases} \qquad (4)$$

$$F_{ind} = \prod_{l=1}^{L} P_l \cdot R \qquad (5)$$

개체의 적응도가 구해지면 적응도에 따라 다음 세대를 위한 부모개체를 선별한다. 선별은 개체의 적응도에 의해 다음 세대에 생존하는 부모개체를 모집단으로부터 선택

하는 과정이다. 선별은 확률바퀴(roulette wheel) 선별, 순위(ranking) 선별, 토너먼트(tournament) 선별 등이 있다. 본 연구에서는 토너먼트 선별을 사용한다. 토너먼트 선별은 모집단의 개체를 임의로 나열하고 k개씩을 차례로 비교하여 적응도가 가장 높은 개체를 선별하는 방법이다. 이 과정이 모집단의 크기만큼 반복된다. 이 방법은 파라미터인 토너먼트 크기 k를 조절하여 선별압력(selection presure)을 조정할 수 있다. k값이 크면 선별압력은 증가하여 좋은 해가 많이 선택되지만 해의 다양성은 줄어든다(Goldberg, 1989).

3.4 유전연산자

자연계에서는 한 세대의 모집단에 속한 개체들이 서로 결합하여 자손을 생산함으로써 다음 세대의 개체들을 생산하고, 이러한 과정 속에서 돌연변이가 발생하여 부모 개체와 형질이 다른 개체들이 생겨나게 된다. 이러한 일련의 과정을 유전알고리즘에서는 교차와 돌연변이의 연산자를 이용하여 구현하고 있다. 교차는 서로 다른 두 개체의 유전 인자들을 교차하여 자손에게 유전함으로써 새로운 개체를 생산하는 연산자이다. 교차를 통해 자손들은 부모 개체들이 여러 세대교체 속에서 살아남을 수 있었던 우성인 형질(정보)을 계승받는다. 돌연변이는 부모로부터 유전된 개체의 형질을 변화시킴으로써 부모와 다른 형질을 갖는 개체를 생산하는 연산자이다. 유전연산자는 표현과 문제의 특성에 따라 여러 방법들이 제안되었다. 본 연구에서는 ORDP의 특성을 이용한 발견적 유전연산자를 새롭게 제안하여 사용한다.

(1) 교차

ORDP는 적은 자원으로써 높은 신뢰도를 갖는 단계가 좋은 정보이다. 따라서, 각 단계가 갖는 정보를 효율적으로 추출할 수 있는 발견적 교차방법을 사용한다. 이 방법은 각 단계의 자원 소요량에 따른 신뢰도의 상대적 기여도를 계산하여 기여도가 큰 단계가 자손에 유전되도록 한다. 기여도는 단계의 평균 신뢰도와 각 단계의 신뢰도의 차, 단계의 평균 자원 소요량과 각 단계의 자원 소요량의 차의 비로써 결정된다. 결국 이러한 교차 방법은 적은 자원으로 시스템 신뢰도를 향상시킬 수 있는 단계가 다음 세대에 생존할 수 있도록 유도하고자 하는 것이다. 이를 통해, 알고리즘의 성능을 향상시킬 수 있다.

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} h_{lij} \cdot x_{ij} \leq 1, \quad l = 1,2,\cdots,L \quad (2')$$

한편, 제약에 대한 측정단위가 모두 다른 경우에는 이를 합리적으로 비교하는데 한계를 갖는다. 따라서, 제2장에서 보인 식 (2)의 양변을 우변상수로 나누어 우변상수가 모두 1이 되도록 정규화(normalization)하는 식 (2′)를 사용하는 것이 합리적이다. 여기서, h_{lij}는 g_{lij}의 정규화된 값, 즉 $h_{lij} = g_{lij} / b_l$을 뜻한다. 이 값은 교차에서 신뢰도에 대한 기여도 및 자원 소요량을 계산하는 과정에 사용된다.

본 연구에서 제안한 교차는 다음의 과정을 통해 이루어진다. 아래 절차에 의해 교차가 수행되면 하나의 자손 o1이 생산된다. 또 다른 자손 o2는 부모개체 p1과 p2의 역할을 바꾸어 생산한다.

step 1: 부모개체 p1, p2를 선택한다.

step 2: p1에 대해 각 단계의 기여도 d_i를 다음과 같이 계산한다.

$$d_i = \frac{R_i - \bar{R}}{v_i - \bar{v}}$$

여기서, $\bar{R} = R^{1/N}$,

$$\bar{v} = \sum_{i=1}^{N}\sum_{j=1}^{n_i}\left\{x_{ij}\left(\sum_{l=1}^{L} h_{lij}\right)\right\}/N,$$

$$v_i = \sum_{j=1}^{n_i} x_{ij}\left(\sum_{l=1}^{L} h_{lij}\right).$$

그리고, R은 제약을 고려하지 않고 계산된 개체(p1)의 시스템 신뢰도이다.

step 3: 범위 [1, $N/2$]의 임의 정수, N_q를 생성한다. d_i가 높은 N_q번째까지 해당하는 단계를 p1으로 부터 받고, 나머지 단계는 p2로부터 받아 자손 o1을 생산한다.

앞의 3.1절에서 소개한 <표 1>의 자료를 바탕으로 하여 교차과정을 예를 들어 설명하면 다음과 같다. 그리고 <그림 4>는 교차에 의해 새로운 자손 개체가 생산되는 과정을 보인 것이다.

step 1: 교차 대상이 되는 두 부모 개체가 <그림 4>에서 보인 p1, p2와 같이 선택되었다고 하자.

step 2: p1에 대해 각각의 값을 계산하면 다음과 같다.

R=0.996, $\bar{R}$ =0.998, $\bar{v}$ =1.053,
R_1=0.9997, R_2=0.9966, R_3=0.9994,
v_1=1.140, v_2=0.900, v_3=1.120,
d_1=0.013, d_2=0.013, d_3=0.012.

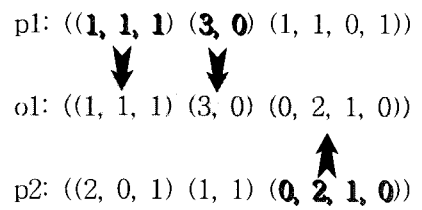
step 3: 만약, N_q=2라 하자. d_i가 큰 2개의 단계(단계 1, 단계 2)를 p1에서 받고, 나머지 단계 3은 p2에서 상속받아 자손 o1을 생산한다.

step 4: 부모의 역할을 바꿔 p2에 대해 step 2와 step 3을 반복한다.

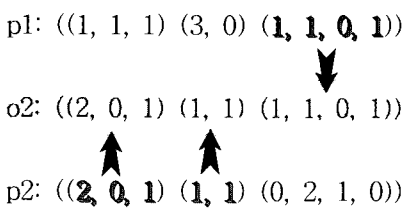
step 2: p2에 대해 각각의 값을 계산하면 다음과 같다.

R=0.980, $\bar{R}$ =0.993, $\bar{v}$ =1.007,
R_1=0.9998, R_2=0.9805, R_3=0.9996,
v_1=1.100, v_2=0.700, v_3=1.220,
d_1=0.070, d_2=0.042, d_3=0.030.

step 3: N_q=2이므로, d_i가 큰 2개의 단계(단계 1, 단계 2)를 p2에서 받고, 나머지 단계 3은 p1에서 상속받아 자손 o2를 생산한다.



(a) 자손개체 o1의 생산



(b) 자손개체 o2의 생산

<그림 4> 교차에 의해 생산된 자손개체들

(2) 돌연변이

돌연변이는 개체의 인자들을 변화시켜 다양한 해의 탐색을 가능하게 하는 연산자이다. 따라서 ORDP에 사용될 수 있는 돌연변이 역시 임의의 인자를 선택하여 해당 인자값을 변화시키는 방법을 사용할 수 있다. 이것은 어느 단계에 추가될 수 있는 부품의 수를 변화시킨다는 의미가 된다.

본 연구에서는 돌연변이 역시 인자(부품)의 신뢰도와 자원 정보를 이용하여 인자의 기여도(e_j)를 계산하고 기여도에 따라 인자값을 변화시키는 발견적 돌연변이 방법을 사용하였다. 이것은 개체가 적은 자원으로 신뢰도를 개선할 가능성이 높은 인자값을 변화시키는 것이다. 돌연변이율은 유전인자 단위가 아닌 단계 단위로 주어진다. 그 절차는 다음과 같다.

step 1: 모집단에서 돌연변이율에 의해 단계 i^*가 선택되었다고 하자. 이때 i^*를 포함한 개체의 가능여부를 판별한다.

step 2: step 1에서 선택된 개체가 가능개체이면 step 3으로, 그렇지 않으면 (즉, 비가능개체이면) step 4로 간다.

step 3: 단계 i^*에 대응되는 인자들에 대하여 다음과 같이 e_j를 계산한다.

$$e_j = \frac{r_{i^*j}}{\sum_{l=1}^{L}\{h_{li^*j}/(1-G_l)\}}$$

가장 큰 e_j를 갖는 인자를 선택하여 그 인자값을 u_{i*}의 범위안에서 임의로 높여준다. 만약, $\sum_{j=1}^{n_{i*}} x_{i^*j} = u_{i^*}$ 이면 가장 큰 e_j를 갖는 인자값을 +1, 가장 작은 e_j를 갖는 인자값은 -1을 해준다.

step 4: 단계 i^*에 대응되는 인자들에 대하여 다음과 같이 e_j를 계산한다.

$$e_j = \frac{r_{i^*j}}{\sum_{l=1}^{L} h_{li^*j}}$$

인자값이 1보다 크고, 가장 작은 e_j를 갖는 인자의 값을 u_{i*}의 범위안에서 임의로 줄여준다.

돌연변이 과정의 예는 다음과 같다. <그림 5>는 각각 가능개체와 비가능개체에 대한 돌연변이 과정에 의해 새로운 자손이 생산된 모습을 보여준다.

가능개체에 대한 돌연변이의 예

step 1: 단계 i^* = 3으로 선택되었다고 하자.

step 2: 만약, 선택된 개체가 ((2, 0, 1) (1, 0) (0, 0, 1, 1))이라면 가능개체이므로, step 3을 수행한다.

step 3: e_1=0.90/1.718=0.524,
e_2=0.91/1.881=0.484,
e_3=0.95/1.989=0.478,
e_4=0.93/1.660=0.560.
e_4의 값이 가장 높으므로 단계 3의 네번째 인자값을 u_3의 범위 내에서 임의로 높여준다.
(예: x_{34} = 1 → x'_{34} = 2)

비가능개체에 대한 돌연변이의 예

step 1: 단계 i^* = 2로 선택되었다고 하자.

step 2: 만약, 선택된 개체가 ((0, 2, 0) (2, 2) (1, 0, 1, 1))이라면 비가능개체이므로, step 4를 수행한다.

step 4: e_1=0.85/0.300=2.833,
e_2=0.87/0.400=2.175.

e_2의 값이 가장 낮으므로 단계 2의 두번째 인자값을 u_2의 범위 내에서 임의로 줄여준다.
(예: $x_{22} = 2 \rightarrow x'_{22} = 1$)

p: ((2, 0, 1) (1, 0) (0, 0, 1, **1**))
o: ((2, 0, 1) (1, 0) (0, 0, 1, **2**))

(a) 가능개체의 돌연변이

p: ((0, 2, 0) (2, **2**) (1, 0, 1, 1))
o: ((0, 2, 0) (2, **1**) (1, 0, 1, 1))

(b) 비가능개체의 돌연변이

<그림 5> 돌연변이에 의해 생산된 자손개체들

4. 실험 및 결과분석

4.1 실험문제 및 실험설계

본 연구에서 제안한 ORDP를 위한 유전알고리즘의 성능을 분석하기 위하여 실험을 실시하였다. 실험문제로는 Nakagawa and Miyazaki(1981)가 제시한 14개 단계를 갖는 문제에 가용한 자원의 허용한도를 다양하게 하여 만든 33개 문제를 사용하였다. 이 문제들은 시스템 비용과 무게의 제약을 갖는 문제로써 이들 문제의 수리모형은 <그림 2>에서 식 (2)만 식 (6)과 (7)로 변환시키고, 식 (1)과 식 (3)을 그대로 사용하면 된다.

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} c_{ij}x_{ij} \leq C \qquad (6)$$

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} w_{ij}x_{ij} \leq W \qquad (7)$$

여기서, c_{ij}, w_{ij}는 단계 i에 추가되는 j번째 부품의 비용, 무게를 각각 나타내고, C와 W는 시스템에 허용되는 비용과 무게의 상한값이다. c_{ij}, w_{ij}, 그리고 목적함수에서 요구되는 r_{ij}, 즉 단계 i에 추가되는 j번째 부품의 신뢰도에 대한 정규화되지 않은 구체적인 값은 <표 2>에 주어져 있다. 이 표는 Fyffe *et al.*(1968)에 의해 처음 소개되었으며, 신뢰도 최적화와 관련된 여러 연구문헌에서 실험문제로 사용하였다(Nakagawa and Miyazaki, 1981; Gen and Cheng, 1997). 한편, 33개의 실험문제에서 시스템의 비용제약(C)은 130으로 고정되고, 무게제약(W)만을 달리하여 설정되었다. 무게제약은 문제 1부터 문제번호가 증가될 때마다 191부터 1씩 차감하여 문제 33의 무게제약은 159로 설정하였다.

유전알고리즘으로 다루고자 하는 문제를 해결하기 위해서는 앞에서 언급한 유전요소들과 함께 여러 유전 파라미터들을 결정해야 한다. 여기에는 모집단 크기(population size), 교차율(crossover rate), 돌연변이율(mutation rate), 종료조건(stopping criteria)

<표 2> 실험 문제의 데이터

단계 번호	부품의 종류(n_i)											
	1			2			3			4		
	r_{i1}	c_{i1}	w_{i1}	r_{i2}	c_{i2}	w_{i2}	r_{i3}	c_{i3}	w_{i3}	r_{i4}	c_{i4}	w_{i4}
1	0.90	1	3	0.93	1	4	0.91	2	2	0.95	2	5
2	0.95	2	8	0.94	1	10	0.93	1	9	-	-	-
3	0.85	2	7	0.90	3	5	0.87	1	6	0.92	4	4
4	0.83	3	5	0.87	4	6	0.85	5	4	-	-	-
5	0.94	2	4	0.93	2	3	0.95	3	5	-	-	-
6	0.99	3	5	0.98	3	4	0.97	2	5	0.96	2	4
7	0.91	4	7	0.92	4	8	0.94	5	9	-	-	-
8	0.81	3	4	0.90	5	7	0.91	6	6	-	-	-
9	0.97	2	8	0.99	3	9	0.96	4	7	0.91	3	8
10	0.83	4	6	0.84	4	5	0.90	5	6	-	-	-
11	0.94	3	5	0.95	4	6	0.96	5	6	-	-	-
12	0.79	2	4	0.82	3	5	0.85	4	6	0.90	5	7
13	0.98	2	5	0.99	3	5	0.97	2	6	-	-	-
14	0.90	4	6	0.92	4	7	0.95	5	6	0.99	6	9

등이 있다. 유전파라미터는 넓은 해공간의 탐색과 좋은 해의 효율적인 이용이 조화될 수 있도록 해결하려는 문제의 특성에 따라 적절히 결정되어야 한다. 유전 파라미터들의 적정값을 결정하는 방법에 대해 많은 연구들이 있지만(Grefenstette, 1986), 유전 파라미터의 결정은 아직 학문이라기 보다는 기술로써 남아있다(Michalewicz, 1999). 본 연구에 적용된 유전 파라미터 중에서 모집단의 크기와 종료조건은 계산시간과 탐색효율을 고려하여 적절히 설정하였고, 교차율, 돌연변이율은 예비실험을 통해 비교적 우수한 성능을 보인 값으로 결정하였다. 모집단 크기는 100, 토너먼트 크기(k)는 2, 교차율은 0.35, 돌연변이율은 0.04, 그리고 종료조건은 200세대로 하였다. 제안된 유전알고리즘은 C++ 프로그래밍 언어로 구현되었으며, 400MHz의 펜티엄 CPU를 탑재한 IBM-PC에서 수행되었다.

4.2 실험결과 및 분석

제안한 알고리즘의 성능은 Nakagawa and Miyazaki(1981)의 결과와 비교한다. 그들의 연구에서는 본 연구에서와 동일한 실험문제를 사용하여, 그들이 개발한 알고리즘(여기서는 N&M 알고리즘이라고 표현하기로 함)의 성능결과를 제시하였다. 또한, N&M 알고리즘은 ORDP의 기존 연구에서 많이 접근하였던 라그랑즈 승수(lagrange multiplier)를 이용한 동적계획법에 의한 해법보다 우수함을 보였다. 따라서 여기에서는 이 실험 결과와 본 연구에서 새롭게 제안한 유전알고리즘에 의한 실험 결과를 비교하고자 한다. N&M 알고리즘은 여러 제약들을 하나의 제약으로 결합하여 얻을 수 있는 대리(surrogate) 제약을 사용한다는 개념을 갖는 알고리즘이다. N&M 알고리즘의 구체적인 절차는 Nakagawa and Miyazaki (1981)의 연구를 참조할 수 있다.

실험결과는 <표 3>에 제시되어 있다. <표 3>의 1열, 2열은 각각 문제번호와 무게제약(W)을 보인 것이다. 3, 4, 5열은 N&M 알고리즘에 의해 찾아진 해와 그에 대한 자원 소요량을 각각 나타낸다. 6, 7열은 본 연구에서 제안한 유전알고리즘에 의해 구한 해로써 6열은 10회 반복실험에서 얻어진 평균값을, 7열은 반복실험중 최선해를 각각 나타낸다. 8, 9열은 최선해에 대한 자원 소요량을 보여준다. 4열에서 '*'로 표시된 문제는 N&M 알고리즘이 제약을 어긴 해를 찾은 경우이다. 7열의 '+' 는 제안된

<표 3> 알고리즘의 성능 분석 결과

문제 번호	무게 제약	N&M 알고리즘			제안한 유전알고리즘			
		신뢰도	비용	무게	신뢰도 (평균)	신뢰도 (최대)	비용	무게
1	191	0.9864	130	191	0.9857	0.9864	129	191
2	190	0.9854	132*	189	0.9852	0.9856	129	190
3	189	0.9850	131*	188	0.9844	0.9852	129	189
4	188	0.9847	129	188	0.9841	0.9848	129	188
5	187	0.9840	133*	186	0.9834	0.9838	127	187
6	186	0.9831	129	186	0.9829	0.9837	127	186
7	185	0.9829	129	185	0.9820	0.9828+	127	185
8	184	0.9822	126	184	0.9818	0.9826	128	184
9	183	0.9815	130	182	0.9813	0.9818	129	183
10	182	0.9815	130	182	0.9806	0.9810+	128	182
11	181	0.9800	128	181	0.9799	0.9803	123	181
12	180	0.9796	126	180	0.9789	0.9798	123	180
13	179	0.9792	127	179	0.9783	0.9790	124	179
14	178	0.9772	123	177	0.9774	0.9779	123	178
15	177	0.9772	123	177	0.9770	0.9772	123	177
16	176	0.9764	125	176	0.9752	0.9760+	123	176
17	175	0.9744	121	174	0.9745	0.9752	122	175
18	174	0.9744	121	174	0.9733	0.9744	121	174
19	173	0.9723	122	173	0.9723	0.9735	120	173
20	172	0.9720	123	172	0.9718	0.9724	121	172
21	171	0.9700	119	170	0.9711	0.9714	119	171
22	170	0.9700	119	170	0.9699	0.9707	119	170
23	169	0.9675	121	169	0.9684	0.9691	119	169
24	168	0.9666	120	168	0.9671	0.9680	118	168
25	167	0.9656	117	167	0.9656	0.9657	117	167
26	166	0.9646	116	166	0.9647	0.9648	118	166
27	165	0.9621	118	165	0.9629	0.9636	116	165
28	164	0.9609	116	164	0.9617	0.9623	114	164
29	163	0.9602	114	163	0.9600	0.9602	114	163
30	162	0.9589	112	162	0.9587	0.9590	113	162
31	161	0.9565	111	161	0.9565	0.9578	111	161
32	160	0.9546	110	159	0.9554	0.9556	111	160
33	159	0.9546	110	159	0.9543	0.9546	110	159

유전알고리즘이 찾은 최선해 중 N&M 알고리즘에 의해 해결된 제약을 어기지 않은 문제들과 비교하여 낮은 값을 갖는 경우이다.

최선해의 비교에서 3개의 문제(7, 10, 16번)을 제외한 모든 문제에서 제안한 알고리즘의 최선해가 N&M 알고리즘보다 같거나 우수함을 보였다. 특히, N&M 알고리즘이 제약을 어긴 문제에 대해서도 제안한 알고리즘은 10초 미만의 비교적 짧은 시간에 제약을 만족하는 수용 가능한 해를 찾았다.

또한, 제안된 알고리즘은 N&M 알고리즘의 해와 비교하여 보다 적은 자원으로 더 높은 신뢰도를 갖는 최선해를 찾는 경우가 많았다. 평균값의 비교에 있어서 제안한 알고리즘은 33개의 문제 중 2/3정도가 N&M 알고리즘보다 낮은 값을 보였고, 나머지 다른 문제에서는 같거나 높은 값을 갖는 것으로 나타났다.

이러한 실험결과를 보인 주된 이유로는 본 연구에서 제안한 유전알고리즘이 문제의 특성을 반영한 발견적 유전연산자를 사용한 때문으로 판단된다. 즉, 교차는 각 단계의 신뢰도가 시스템의 전체 신뢰도에 어느 정도 영향을 미치는지 계산하여 기여도가 큰 단계가 자손에 상속되도록 설계되었다. 또한, 돌연변이에서는 기여도가 가장 낮은 단계의 인자구성을 변환시켜 신뢰도를 개선할 가능성을 높일 수 있다.

5. 요약 및 결론

본 연구에서는 시스템의 신뢰도를 개선하기 위한 방안으로써 중복 설계시 발생하는 시스템 신뢰도 최적화 문제(ORDP)를 다루었다. 이 문제는 시스템 신뢰도와 가용 자원간의 상호조정(trade-off)을 위한 최적 중복수를 결정하는 문제로써 시스템 규모가 커질수록 해 공간이 지수적으로 증가하는 NP-hard 부류에 속하는 문제이다.

본 연구는 ORDP를 효율적으로 해결하기 위하여 유전알고리즘의 적용방안을 제시하였다. 이를 위하여 해의 적절한 탐색에 용이한 개체의 표현, 초기 모집단 생성방법, 비가능해를 효율적으로 다룰 수 있는 적응도 평가 방법을 제안하였다. 또한, 유전알고리즘을 이용하여 ORDP를 해결한 기존 연구들과 다르게, 해의 탐색성능을 향상시키기 위해 개체가 갖는 정보를 이용하는 발견적 유전연산자를 새롭게 개발하였다. 제안한 알고리즘의 성능을 분석하기 위해 비용제약과 무게제약을 갖는 문제로써 자원의 범위를 달리하여 만든 33개의 문제에 대해 실험을 실시하여, 그 결과는 기존의 알고리즘과 비교되었다. 실험 결과, 제안된 알고리즘은 실험 문제 모두에 대해 효율적임을 알 수 있었다. 또한 기존의 방법과 비교에서도 제안한 알고리즘이 보다 우수하였다.

본 연구 결과는 더욱 대형화, 복잡화되는 시스템의 설계시 신뢰도와 가용한 자원간의 상관관계를 분석하는데 적용 가능할 것으로 보인다. 그리고 다양한 자원의 제약 외에 기술적인 제약이 추가되는 경우에 있어서도 유연하게 적용 가능하여 비교적 짧은 시간에 시스템의 설계와 관계된 경제성 분석의 정책수립과 의사결정 자료로써 활용될 수 있을 것으로 기대된다.

참고문헌

[1] 김여근, 윤복식, 이상복 (1997), 메타휴

리스틱, 영지문화사.

[2] Coit, D.W. and Smith, A.E. (1996), "Penalty guided genetic search for reliability design optimization," Computers and Industrial Engineering, Vol. 30, No. 4, pp. 895-904.

[3] Fyffe, D., Hines, W., and Lee, N. (1968), "System reliability allocation and a computational algorithm," IEEE Transactions on Reliability, Vol. R-17, p. 68.

[4] Gen, M. and Cheng, R. (1997), *Genetic Algorithms and Engineering Design*, John Wiley & Sons, New York.

[5] Goldberg, D.E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

[6] Grefenstette, J.J. (1986), "Optimization of control parameters for genetic algorithms," IEEE transactions on system, man, and cybernetics, Vol. 16, pp. 122-128.

[7] Holland, J.H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

[8] Kuo, W. and Rajendra, V. (2000), "An annotated overview of system-reliability optimization," IEEE Transactions on Reliability, Vol. 49, No. 2, pp. 176-187.

[9] Lee, C.Y., Yun, Y.S., and Gen, M. (2002), "Reliability optimization design for complex systems by hybrid GA with fuzzy logic control and local search," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E85, No. 4, pp.880-891.

[10] Michalewicz, Z. (1999), *Genetic algorithms + Data Structures = Evolution Programs*, 3rd., Springer, Berlin.

[11] Nakagawa, Y. and Miyazaki, S. (1981), "Surrogate constraints algorithm for reliability optimization problems with two constranints," IEEE Transactions on Reliability, Vol. R-30, No. 2, pp.175-180.

[12] Olsen, A.L. (1994), "Penalty function and the knapsack problem," IEEE International Conference on Evolutionary Computation, pp. 554-558.