유전 알고리즘에서의 문제 독립적 유전자 재배열
(Problem-Independent Gene Reordering for Genetic Algorithms)

권 영 근 † 김 용 혁 † 문 병 로 ††
(Yung-Keun Kwon) (Yong-Hyuk Kim) (Byung-Ro Moon)

요 약 위치기반 인코딩을 사용하는 유전 알고리즘에서 정적 유전자 재배열이란 상관성이 높은 유전자들이 서로 인접하도록 배치하는 것을 말한다. 그것은 유전 알고리즘이 효과적으로 고품질의 스키마들을 생성하고 보존하는 데 도움을 준다. 본 논문에서는 선형의 위치기반 인코딩을 위한 정적 재배치 방법을 제안한다. 본 논문에서 제안하는 방법은 특정 문제에 한정된 정보를 사용하지 않는다는 점에서 기존의 방법들과 차이가 있다. 그것은 모든 유전자들 사이의 상관성을 계산하여 가중치가 있는 완전 그래프를 만든다. 그리고 그 그래프에서 상대적으로 가중치가 높은 간선들만 골라 냄으로써 가중치가 없는 희소 그래프로 변환한다. 끝으로 그래프 탐색을 통해 유전자 재배열을 찾는다. 여러 문제에 관한 광범위한 실험을 통해 본 논문에서 제안한 방법은 재배열을 하지 않은 유전 알고리즘에 비해 현저한 성능 향상을 보여 주었다.

키워드 : 유전 알고리즘, 정적 재배치, 위치기반 인코딩

Abstract In genetic algorithms with locus-based encoding, static gene reordering is to locate the highly related genes closely together. It helps the genetic algorithms to create and preserve the schema of high-quality effectively. In this paper, we propose a static reordering framework for linear locus-based encoding. It differs from existing reorderings in that it is independent of problem-specific knowledge. It makes a complete graph where weights represent the interelationship between each pair of genes. And, it transforms the graph into a unweighted sparse graph by choosing the edges having relatively high weight. It finds a gene reordering by graph search method. Through the wide experiments about several problems, the method proposed in this paper shows significant performance improvement as compared with the genetic algorithm that does not rearrange genes.

Key words : genetic algorithm, static reordering, locus-based encoding

1. 서 론

Holland는 스키마 정리를 통해 결정 길이가 짧고 차수가 낮은 좋은 품질의 스키마들이 전통적인 유전 알고리즘(Genetic Algorithm: GA) 구조에서 생존 확률이 높다는 것을 보였다. 그러한 특징을 보이는 좋은 품질의 스키마들을 빌딩 블록이라고 한다. 빌딩 블록은 상호간 강한 상관성을 가지면서 적합도에 크게 공헌하는 유전자들의 집합이다. 유전 알고리즘의 성능은 빌딩 블록의 생존 환경과 재생산성에 의해 크게 영향을 받는다.

어떤 유전자 그룹의 교배 연산에 의한 생존 확률은 염색체내 유전자들의 배치에 의해 크게 좌우된다. 스키마의 특정 기호들의 위치가 넓게 분산되면 결정 길이가 길어지므로 교배를 거치면서 생존할 확률이 낮아진다. 따라서 유전자들의 자리를 정하는 전략은 유전 알고리즘의 성능에 크게 영향을 미친다. 예를 들면 역치는 동적으로 유전자들의 위치를 바꾸기 위해 만든 방법 중의 하나이다[1]. 그리고, 동적으로 유전자들의 위치를 활용하기 위한 방법들을 결합 학습이라고 부른다[2]. 메시 유전 알고리즘(Messy GA)은 잠재적으로 동적 유전자 재배치를 수행하는 사례이다[3].

위치기반 인코딩의 유전 알고리즘 성능은 유전자 위치를 정적으로 재배치함으로써 향상될 수 있다는 사실이 연구되어 왔다. 유전 알고리즘에서 정적 재배치 기법은 참고문헌[4,5]에서 처음 소개되었다. 그 기본적인 아이디어는 염색체 표현에서 유전자들의 위치를 유전 알고리즘이 좋은 스키마를 효과적으로 잘 보존할 수 있도

† 비 회 원 : 서울대학교 컴퓨터공학부
kwon@soar.snu.ac.kr
yhdfly@soar.snu.ac.kr
†† 정 회 원 : 서울대학교 컴퓨터공학부 교수
moon@soar.snu.ac.kr
논문접수 : 2004년 8월 19일
심사완료 : 2005년 8월 19일

록 재배치하는 것이다. 좋은 재배치는 원래의 배치 때보다 좋은 품질의 스키마를 더욱 잘 생성할 수 있어야 한다. 위치기반 인코딩에서 유전자 위치를 정적으로 재배치함으로써 유전 알고리즘의 성능을 향상시킬수 있다는 사실이 많은 연구들을 통해 확인되었다[4-7].

그러나, 기존의 재배치 방법들은 그 응용 문제에 한정된 정보에 의존한다[4-6]. 그러므로, 새로운 문제들 각각에 대해 새로운 휴리스틱을 통해 재배치해야 한다. 본 논문에서는 특정 문제에 한정적인 정보를 사용하지 않는 정적 재배치 방법을 제안한다. 그 방법은 염색체 표현으로서 위치기반 인코딩을 하는 경우에 적용될 수 있다. 본 논문에서는 세 가지 대표적인 NP-hard 조합 최적화 문제들(그래프 분할, 선형 배열, 순회 판매원 문제)에 대해 실험을 수행하였다. 그 결과, 재배치를 하지 않는 경우에 비해 현저한 성능 향상을 보여 주었다. 한편, 이 논문에서 재배열(rearrangement), 재배치(reordering), 전처리(preprocessing) 등은 같은 의미로 사용된다.

본 논문의 구성은 다음과 같다. 2절에서는 세 가지 테스트 문제에 대해서 요약하고, 3절에서는 이 논문에서 이용한 유전 알고리즘 구조를 설명한다. 4절에서는 문제 독립적인 유전자 재배치에 대해 논한다. 5절에서는 실험 결과를 설명하고 끝으로 6절에서 결론을 맺는다.

2. 테스트에 사용된 문제

2.1 그래프 2-분할 문제

$G=(V,E)$를 무가중치 무방향 그래프라고 하자. 이 때 V는 n개의 노드들의 집합이고 E는 e개의 간선들의 집합이다. 그래프 G의 균형 잡힌 2-분할 $\{C_1,C_2\}$은 $C_1,C_2 \subset V, C_1 \cup C_2 = V, C_1 \cap C_2 = \phi, ||C_1|-|C_2|| \leq 1$을 만족하는 분할을 말한다. 분할 $\{C_1,C_2\}$의 컷사이즈는 끝점이 서로 다른 부분 집합에 속해 있는 간선들의 총 수를 말하며 $|\{(v,w) \in E: v \in C_1, w \in C_2\}|$으로 정의된다. 그래프 분할 문제는 최소 컷사이즈를 갖는 분할을 찾는 문제이다. 그 문제는 과거에 광범위하게 연구되었고[5,8-10], NP-hard 문제로 알려져 있다[11].

2.2 선형 배열

무가중치 무방향 그래프 $G=(V,E)$가 주어졌을 때 $\sum_{(u,v)\in E} |\sigma(u)-\sigma(v)|$으로 정의되는 스팬 합을 최소화하는 순열 $\sigma: V \rightarrow V$을 찾는 문제를 선형 배열 문제라고 한다. 선형 배열 문제를 풀기 위해 많은 연구들이 있어 왔으며[12-14], 이 문제도 NP-hard 문제로 알려져 있다[11].

2.3 순회 판매원 문제

$G=(V,E)$를 각 간선에 가중치가 있는 완전 그래프라고 하자. G의 해밀토니안 사이클은 그래프의 모든 점을 한 번씩만 지나는 사이클을 말한다. 순회 판매원 문제(Traveling Salesman Problem: TSP)는 최소 가중치를 갖는 해밀토니안 사이클을 찾는 문제로서, 잘 알려진 NP-hard 문제이다[11]. TSP는 광범위한 응용과 복잡성 때문에 과거에 폭넓게 연구되었다. 유전 알고리즘 역시 TSP에 많이 적용되어 좋은 성능을 보여 주었다[15-18].

3. 혼합형 유전 알고리즘

혼합형 유전 알고리즘은 지역 최적화 휴리스틱이 결합된 유전 알고리즘이다. 그림 1은 이 논문에서 사용된 혼합형 안정상태 유전 알고리즘의 일반적인 구조를 보여 준다. GA의 각 부분에 대한 자세한 설명은 다음과 같다.

```
안정상태 유전 알고리즘 {
    염색체의 유전자 배열을 정한다. // 정적 재배열
    해집단을 초기화한다.
    while( 종료 조건이 만족되지 않는다면 ) {
        해집단에서 부모1, 부모2를 선택한다.
        부모1, 부모2를 교배하여 자식해를 만든다.
        자식해를 지역 최적화한다.
        해집단의 한 개체를 자식해로 대치시킨다.
    }
    해집단 중 가장 좋은 해를 반환한다.
}
```

그림 1 이 논문에서 사용된 혼합형 유전 알고리즘의 구조

- 위치기반 인코딩: 해집단에서 각각의 해는 염색체로 표현된다. 먼저, 그래프 분할 문제에 대해서는 이진 인코딩이 사용되었다. 각 유전자는 0 또는 1의 값을 갖는데 대응되는 노드가 속한 부분 집합을 나타낸다. 선형 배열 문제에서는 순차 순열 인코딩을 사용하는데, 각 유전자는 해당 점의 선형 배열상의 위치를 표현한다. 반면, 순회판매원 문제에서는 순환 순열 인코딩을 사용하는데, 어떤 노드 v에 대응하는 유전자는 그 해밀토니안 사이클에서 v 뒤에 오는 다른 점을 표현한다. 이와 같이, 문제에 따라 인코딩이 다르기는 하지만 유전자들의 위치가 명시적인 의미를 갖는다는 점에서 공통점을 갖는데 그래서 이러한 인코딩을 위치기반 인코딩이라 부른다. 다음 4절에서 설명할 유전자 재배열 휴리스틱은 오직 위치기반 인코딩에만 적용될 수 있으므로 위치기반 인코딩을 사용하는 것은 반드시 필요하다.
- **선택과 교배 연산자:** 두 부모를 선택하기 위해 선택 연산자로서 해집단에서 가장 좋은 해가 가장 나쁜 해보다 뽑힐 확률이 4배 크도록 조정하여 부모해를 뽑는 적합도 비례 룰렛휠 선택 연산자가 사용된다. 한편, 교배 연산자는 부모해를 부분적으로 결합함으로써 새 자식해를 생성한다. 그래프 분할 문제에서는 5점

교배 연산자를 사용하였다. 교배 연산 뒤에 자식해는 균형 조건($||C_1|-|C_2|| \leq 1$)을 만족시키지 못할 수 있다. 이러한 적합하지 않은 해를 복구하기 위해 염색체에서 임의의 지점을 선택하여 그 곳부터 오른쪽으로 훑어가면서 균형을 위해 필요한 1을 0으로 혹은 0을 1로 바꾼다. 순열 인코딩에서는 부분 대응 교배 연산자를 이용한다[19]. 선형 배열 문제에서는 자식해의 유전자 값이 중복되지 않으므로 어떤 복구도 필요하지 않다. 그러나 순회 판매원 문제에서는 상호 단절된 부분 사이클이 두 개 이상 존재할 수 있으므로 해밀토니안 사이클이 되지 않을 수 있다. 이러한 문제점을 해결하기 위해 참고문헌[6]에서 사용된 복구 알고리즘을 사용하였다.

- **지역 최적화:** 유전 알고리즘은 지역 최적점 부근에서 미세 조정을 잘 하지 못하기 때문에 혼합형 유전 알고리즘이 복잡한 문제에서 만족할 만한 성능을 얻기 위한 자연스런 방법으로 간주되고 있다. 이 연구에서는 가장 기본적인 지역 최적화 알고리즘 중의 하나인 2-Opt를 사용하는데 그것은 참고문헌[20]에서 순회 판매원 문제에 적용된 휴리스틱의 개념을 바탕으로 한다. 2-Opt는 두 개의 유전자 값을 반복적으로 교환함으로써 더 좋은 품질의 이웃해를 찾는다. 본 논문의 유전 알고리즘에서는 교배 연산에 의해 생성된 자식해에 2-Opt를 적용한다.
- **대치 연산자와 종료 조건:** 자식해를 생성한 후 유전 알고리즘은 두 부모해 중 품질이 더 좋지 못한 해와 대치시킨다. 이 대치 연산자는 일반적으로 해집단의 다양성을 잘 보존할 수 있는 특징을 갖는데 그 이유는 부모해와 자식해가 상당히 비슷한 유전자형의 해들이기 때문이다. 한편, 본 논문의 유전 알고리즘은 일정 세대가 지난 후 종료한다.

4. 유전자 재배치

스키마는 염색체 안에 포함하고 있는 패턴의 일종이다. 어떤 스키마는 교배 연산을 통해 생존하기도 하고 소멸하기도 한다. 유전 알고리즘은 길이가 짧은 스키마에서 길이가 긴 스키마로 커 가는 과정을 통해 설명될 수 있으므로 품질이 좋은 스키마를 보존하는 일은 매우 중요하다. 일점(one-point) 교배의 경우 길이가 짧은 스키마는 생존하기 쉽지만 다점 교배를 사용하는 경우엔 그 생존 능력이 떨어지게 된다. 스키마의 생존은 그 길이에 물론 영향을 받지만 결정된 기호의 분포에도 영향을 받는다. 예를 들어 다음의 차수(결정된 기호의 개수)가 6인 스키마 H_1과 H_2가 있다고 하자.

H_1 =***#***#***#***#***#***#***,

H_2 =***###**********###***

(여기에서 #은 결정된 기호, *는 결정되지 않은 기호)

두 스키마는 길이(가장 왼쪽의 결정된 기호와 가장 오른쪽의 결정된 기호 사이의 길이)가 20으로 같지만 H_1에서는 결정된 기호가 골고루 분포되어 있는 반면, H_2에서는 클러스터로 뭉쳐있다. 1점 교배를 사용하면 두 스키마의 생존 확률은 6/26으로 같다. 반면, 2점 교배를 사용하는 경우 스키마 H_1의 생존 확률은 $(\binom{6}{2}+5\binom{4}{2})/\binom{26}{2}=45/325$이지만 H_2의 생존 확률은 $(\binom{6}{2}+\binom{16}{2})/\binom{26}{2}=135/325$가 된다. 즉, H_2의 생존 확률이 훨씬 높게 된다. 이 예는 유전자들의 염색체 표현상에서 지리적인 위치가 얼마나 중요한지를 보여주고 있다. 만약 두 유전자가 강한 관계를 가지고 있다면 그들을 지리적으로 가깝게 배치하는 것이 탐색에 있어서 훨씬 이롭게 된다. 유전자 재배치란 이렇듯 강한 관계에 있는 유전자를 가깝게 배치함으로써 품질이 좋은 스키마를 다점 교배 연산에 의해 소멸시키지 않고 잘 보존케 한다.

유전자 재배치가 물론 장점만을 가진다고 볼 수는 없다. 강한 관계를 가지는 유전자를 가깝게 배치하여 스키마가 파괴되지 않고 잘 보존되도록 하는 것은 유전 알고리즘의 탐색에 있어서 설익은 수렴을 야기할 수 있다. 하지만 이러한 문제점은 변이 연산이나 대치 연산, 선택 연산 등에서 해 집단의 다양성을 유지하도록 하는 과정을 통해 충분히 극복할 수 있다.

4.1 문제 독립적 유전자 재배치

3절에서 언급하였듯이, 본 논문에서는 위치기반 인코딩을 사용하고 유전자들을 재배치한다. 그림 2는 이 논문에서 제안하는 스키마 전처리의 과정을 유전자의 개수가 5개인 예를 통해 설명한다. 그것은 특정 문제에 한정된 정보에 의존하지 않고 문제에 독립적으로 작동한다.

4.1.1 [단계 1] 고품질 해들의 생성

먼저, M개의 상대적으로 품질이 좋은 해들을 생성한다. 상대적으로 품질이 좋은 해란 직관적으로는 문제의 해 공간에서 임의로 생성된 해 보다 나은 품질을 가지고 있다고 확신이 되는 해를 말한다. 상대적으로 품질이 낫다는 것은 곧 그 해가 좋은 품질의 스키마를 포함하고 있을 가능성이 많다는 것을 의미한다. 품질이 상대적으로 좋은 이러한 해들을 통해 그들의 공통된 특성으로부터 좋은 스키마가 잘 보존되는 유전자 재배치를 할 수 있게 된다. 상대적으로 품질이 좋은 해를 찾는 데에는 다양한 방식이 가능하다. 예를 들면, 임의로 생성한 많은 해 중에서 좋은 품질을 가지는 해를 선택할 수도 있고 유전 알고리즘과 같은 탐색 알고리즘을 수행하여 그 결과 얻어진 품질이 좋은 해를 선택할 수도 있다. 본

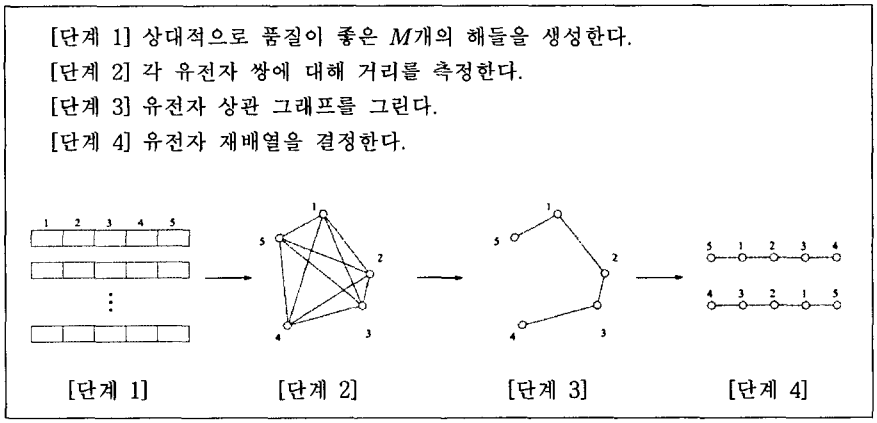


그림 2 문제 독립적 스키마 전처리의 과정

논문에서는 문제의 인코딩에 따라 일반적으로 알려져 있는 근접 탐색 방식인 2-Opt 휴리스틱을 이용하여 100개의 해들을 생성하였다. 이 2-Opt 방식은 지역 탐색 방법의 일종이지만 다른 지역 탐색 방법이 해당 문제에 많이 의존하여 수행되는 데에 비해 2-Opt는 문제의 인코딩만 주어지면 구현이 가능한 문제에 독립적인 가장 일반적인 형태의 탐색 방법이다.

4.1.2 [단계 2] 각 유전자 쌍에 대한 거리 측정

생성된 해들로부터 각 인코딩 종류에 따라 모든 유전자 쌍 사이의 거리를 계산한다. 그림 3은 두 유전자 g_i과 g_j 사이의 거리 $D(g_i,g_j)$를 구하는 방법을 설명한다. 그림에서 $f_l(g_i)$는 l 번째 해의 유전자 g_i의 값을 나타낸다. 그리고 함수 $I()$는 "indicator" 함수를 뜻하는 것으로 인자로 "참"을 받으면 1을, "거짓"을 받으면 0을 리턴하는 함수이다. 즉, $I(true)=1$, $I(false)=0$이다.

이진 인코딩

$$D(g_i,g_j)=\frac{1}{M}\sum_{l=1}^{M} I(f_l(g_i)\neq f_l(g_j))$$

순차 순열 인코딩

$$D(g_i,g_j)=\frac{1}{M}\sum_{l=1}^{M}|f_l(g_i)-f_l(g_j)|$$

순환 순열 인코딩

$$D(g_i,g_j)=\frac{1}{M}\sum_{l=1}^{M} argmin_k(g_i=f_l^k(g_j) or\ g_j=f_l^k(g_i))$$

그림 3 위치기반 인코딩에서 유전자간 거리 척도

거리는 여러 가지로 정의할 수 있고 문제에 따라 달리 정의할 수 있다. 문제가 주어질 때 어떤 거리를 사용해야 하느냐라는 점은 중요한데 그것은 문제의 해가 어떤 인코딩 방식을 사용하느냐에 크게 의존한다. 예를 들어 이진수, K-진수, 자연수, 실수 등 사용하는 인코딩의 유형에 따라 사용할 수 있는 거리 척도가 달라진다. 본 논문에서는 인코딩 방식을 분류하여 이진수 인코딩과 순열 인코딩 문제를 다루는데 이들은 유전 알고리즘 분야에서 가장 많이 이용되는 인코딩이다. 그러나 다른 인코딩 방식에도 문제에 따라 크게 좌우되는 것 없이 거리 측도를 일반적으로 정의하는 것이 가능하다. 예를 들어 실수 인코딩이라면 가장 일반적인 거리인 유클리디안 거리(l_2)를 사용하면 된다.

이진수 인코딩에서 거리 척도로 가장 많이 쓰이는 방식은 해밍 거리이다. 즉, 두 비트가 다른 경우에 거리가 1씩 증가하는 방식을 주로 사용한다. 유전 알고리즘 분야에서는 주로 이진수 인코딩에서 해밍 거리를 사용하고 있다. 본 논문에서도 이진수 인코딩을 하는 문제에 대해 해당 문제에 의존적인 거리를 사용하는 것이 아니라 일반적으로 널리 사용되고 있는 해밍 거리를 적용하였다. 한편, 순열 인코딩을 사용하는 문제에 대해서도 최대한 일반적인 거리를 사용하려고 하였다. 논문의 서론에서도 언급하였지만 유전자 재배치는 위치기반 인코딩을 사용하는 문제로 제한된다. 위치기반 인코딩은 각 유전자가 가지는 고정된 위치 정보가 있어서 i번째 유전자는 그 위치의 유전자가 가지는 정보를 포함하게 된다. 그래서 각 해마다 가지는 유전자들의 정보의 차의 합으로서 거리를 정의하여 해밍 거리로부터 자연스럽게 확장할 수 있다. 이와 같이, 본 논문에서는 거리를 문제의 세부적인 사항에 의존하지 않고 문제에 사용된 인코딩 방식에만 의존하여 정의하였다.

3절에서 설명하였듯이 그래프 분할 문제를 위해서 이진 인코딩이 사용된다. 순차 순열 인코딩과 순환 순열 인코딩이 선형 배열 문제와 순회 판매원 문제를 위해 각각 사용된다. 그래서 유전자들에 대응하는 노드들과 $1-D(g_i,g_j)$를 가중치로 갖는 간선으로 이루어진 완전

표 1 그래프 분할 문제에 대한 실험 결과 (50회 시행)

그래프	기본 배열		BFS 재배열		DFS 재배열		Max-Adj 재배열	
	평균	표준편차	평균	표준편차	평균	표준편차	평균	표준편차
G500.2.5	52.48	1.59	52.38	1.60	51.50	1.54	51.94	1.57
G500.05	220.96	2.16	220.58	2.22	221.12	2.26	220.84	1.66
G500.10	630.32	2.58	629.88	2.35	630.34	2.44	629.64	2.11
G500.20	1751.48	4.43	1749.12	4.13	1750.14	4.22	1748.84	3.38
G1000.2.5	101.08	2.95	99.96	2.64	101.64	2.77	100.02	2.47
G1000.05	455.28	3.14	454.72	3.31	456.20	4.04	454.90	2.28
G1000.10	1374.04	5.20	1374.62	4.15	1372.88	4.83	1372.18	4.84
U500.05	9.16	3.01	5.88	1.98	4.66	1.66	4.72	1.67
U500.10	33.86	9.13	28.06	4.51	26.74	2.09	26.38	1.92
U500.20	178.52	2.14	178.12	0.85	178.18	0.87	178.36	1.44
U500.40	412.00	0.00	412.00	0.00	412.00	0.00	412.00	0.00
U1000.05	25.32	5.99	15.00	4.08	16.24	5.02	12.80	4.16
U1000.10	75.20	15.56	56.82	8.98	44.98	6.56	48.44	5.08
U1000.20	242.04	18.76	231.62	13.17	228.16	10.64	230.32	11.57
U1000.40	737.00	0.00	737.00	0.00	737.00	0.00	737.00	0.00
cat.352	3.28	1.14	1.04	0.28	1.28	0.70	1.76	0.98
cat.702	8.00	1.86	8.60	1.34	4.36	1.48	5.60	1.29
cat.1052	12.20	2.97	15.84	3.23	9.76	2.28	8.40	1.99
rcat.134	1.00	0.00	1.00	0.00	1.08	0.40	1.00	0.00
rcat.554	2.68	0.74	1.80	0.99	1.60	0.93	1.52	0.89
rcat.994	4.08	1.23	2.40	0.93	2.28	0.97	2.68	0.74

그래프를 구하게 된다.

4.1.3 [단계 3] 유전자 상관 그래프 그리기

[단계 2]에서 구한 가중치가 있는 완전 그래프를 유전자 상관 그래프(gene-interaction graph)라 불리는 가중치 없는 희소 그래프로 변환한다. 본 논문에서는 [단계 2]에서 구한 그래프의 가중치들이 정규 분포를 갖는다고 가정한다. [단계 1]에서 상대적으로 품질이 좋은 해들을 골라냈고 이들 해를 사용하여 [단계 2]에서 계산되는 거리로부터 유전자 간의 거리를 계산해 내었다. 상대적으로 품질이 좋은 해를 골라서 품질이 좋은 스키마를 많이 포함할 가능성으로부터 계산된 유전자 간의 거리는 어느 정도 의미를 가지게 된다. 하지만 이 수치는 일반적인 수치가 아니라 어느 정도 노이즈를 포함하는 정보가 되며, 본 논문에서는 각각의 거리는 정규 분포에 해당하는 노이즈를 갖는 것으로 가정하였다. W를 간선의 가중치라 할 때, $P(W \geq w) = 0.05$를 만족하는 w보다 큰 가중치를 갖는 간선들을 골라낸다. 이로부터 가중치 없는 유전자 상관 그래프를 얻게 된다.

4.1.4 [단계 4] 유전자 배열 찾기

유전자 상관 그래프로부터 유전자 재배열을 수행한다. 유전자들의 집합 $g_1, g_2, \ldots, g_n$이 주어졌을 때 유전자 재배열 $g_{\sigma(1)}, g_{\sigma(2)}, \ldots, g_{\sigma(n)}$은 하나의 순열인 일대일 대응 $\sigma : 1,2,\ldots,n \rightarrow 1,2,\ldots,n$에 의해 표현된다. 만약 $\sigma(j) = i$이라면, 유전자 v_i는 유전자 재배열에서 j번째 유전자이다. 일반적으로 유전자 재배열의 목적은 유전자 상관 그래프에서 클러스터링 구조를 보존하는 것이다. 본 논문에서는 세 가지 그래프 탐색 기법을 사용하였는데, 너비 우선 탐색(Breadth First Search: BFS), 깊이 우선 탐색(Depth First Search: DFS), 최대 인접 탐색(Max-Adjacency)[21]을 사용하였다. 너비 우선 탐색과 깊이 우선 탐색은 입력 그래프의 임의의 점에서부터 너비 우선 혹은 깊이 우선으로 각각 탐색한다. 그러한 탐색에 의해 방문하는 순서대로 노드들을 재배열한다. 한편 최대 인접 탐색은 임의의 점에서 출발하여 지금껏 방문하지 않은 노드들 중 방문한 노드들과 가장 많은 간선들로 연결된 노드를 반복적으로 찾아 재배열한다.

5. 실험 결과

5.1 그래프 분할

그래프 분할 문제에서 총 21개의 그래프에 대해 테스트하였다. 그 그래프들은 랜덤 그래프(Gn.d), 랜덤기하 그래프(Un.d), 캐터필러 그래프(cat.n과 rcat.n) 등 세 가지 종류로 나눌 수 있으며 지금까지 많은 연구에 이용되어 왔다[5,9,22,23]. 표 1은 50회 시행에 따른 컷사이즈의 평균과 표준 편차를 나타낸다. 그 표에서, "BFS", "DFS", "Max-Adj"는 유전자 전처리 방법을 가리킨다. 다시 한 번 강조하면 이러한 전처리는 참고문헌 [4]나 [5]에서의 기존 정적 재배열 방법과 달리 해당 문제에 독립적인 유전자 상관 그래프을 가지고 수행된다는 점이다. "기본 배열"에서 유전자들을 임의의 순서로 배열한다. 전처리를 제외하고 유전 알고리즘은 같

은 프레임 워크를 갖는다. 표 2는 기본 배열과 각각의 재배열 사이의 t-검정 결과를 나타낸다. 전처리를 거친 유전 알고리즘은 임의로 배열된 유전자로 수행된 유전 알고리즘에 비해 눈에 띄게 좋은 성능을 보여 주었다. 특히 전처리는 랜덤기하 그래프와 캐터필러 그래프에 대해서 큰 성능 향상을 보여 주었다. 또한 2-Opt 보다 더 강력한 지역 최적화 휴리스틱을 사용한다면 더욱 향상된 해를 구할 수도 있음을 지적하고자 한다. 하지만, 여기에서는 주관심이 제안된 유전자 재배열 방법의 효과를 알아보는 것이므로 지역최적화 방법을 2-Opt로 고정하였다.

재배열에 관한 시각적 통찰을 위해 문제 인스턴스와 전처리된 배열을 그림 4에서 그려 보았다. 그림 4(a)는 원래의 그래프를 보여 준다. 그림 4(b)와 그림 4(c)는 각 유전자 배열에서 연속하는 점들 사이를 선분으로 표현함으로써 나타난 것이다. 당연하게, 임의로 배열된 유전자들은 대부분의 유전자 쌍들 사이의 어떤 관계도 반영하지 못한다. BFS 재배열은 연관성이 큰 유전자들을 염색체에서 가깝게 배치시킴을 볼 수 있다.

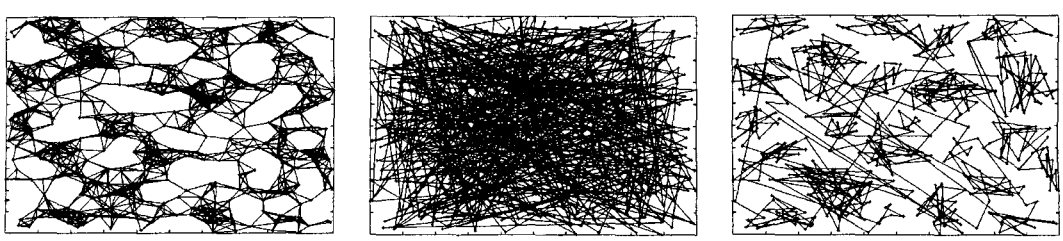
5.2 선형 배열

선형 배열 문제에서는 표 1에서 사용된 그래프 중 희소 기하 그래프에 대해 테스트하였다. 표 3은 50회 시행에 따른 스팬 합의 평균과 표준 편차를 나타내며, 표 4는 기본 배열과 각각의 재배열에 대한 t-검정 결과를 나타낸다. 세 가지 전처리 방법은 모든 인스턴스에서 "기본 배열"보다 성능이 좋았다. 특히 평균적으로 "BFS"가 가장 좋은 성능을 보여 주었다. 그림 5에서 선형 배열 문제에 관해 재배치 이후의 유전자들의 배열 예를 보여 준다. 상관성이 높은 유전자들이 염색체에서 가까이 위치함을 역시 관찰할 수 있다.

표 2 그래프 분할 문제에서 기본 배열 결과와의 t-검정(양측) 결과

그래프	BFS 재배열		DFS 재배열		Max-Adj 재배열	
	t-값	p-값	t-값	p-값	t-값	p-값
G500.2.5	0.31289	0.75503	3.12477	**0.00234**	1.70659	0.09107
G500.05	0.86750	0.38779	-0.36254	0.71773	0.31191	0.75577
G500.10	0.89049	0.37539	-0.03982	0.96832	0.07617	0.15233
G500.20	2.75658	**0.00697**	1.54856	0.12471	3.35038	**0.00115**
G1000.2.5	1.99833	**0.04845**	-0.97805	0.33046	1.94636	0.05448
G1000.05	0.86869	0.38714	-1.27178	0.20646	0.69304	0.48993
G1000.10	-0.61633	0.53911	1.15501	0.25089	1.85122	0.06715
U500.05	6.43766	**4.50E-09**	9.24920	**5.16E-15**	9.11912	**9.88E-15**
U500.10	4.02837	**0.00011**	5.37711	**5.14E-07**	5.67137	**1.43E-07**
U500.20	1.22844	0.22222	1.03999	0.30091	0.43863	0.66190
U500.40	N/A	N/A	N/A	N/A	N/A	N/A
U1000.05	10.06663	**8.68E-17**	8.21104	**8.98E-13**	12.1328	**3.14E-21**
U1000.10	7.23234	**1.06E-10**	12.6532	**2.5E-22**	11.5584	**5.24E-20**
U1000.20	3.21514	**0.00177**	4.55167	**1.53E-05**	3.76071	**0.00029**
U1000.40	N/A	N/A	N/A	N/A	N/A	N/A
cat.352	13.44549	**5.61E-24**	10.54332	**8.04E-18**	7.13468	**1.69E-10**
cat.702	-1.84895	0.06748	10.81527	**2.08E-18**	7.48332	**3.16E-11**
cat.1052	-5.86183	**6.17E-08**	4.60741	**1.23E-05**	7.51760	**2.67E-11**
rcat.134	N/A	N/A	-1.42887	0.15622	N/A	N/A
rcat.554	5.03365	**2.19E-06**	6.44110	**4.43E-09**	7.10207	**1.98E-10**
rcat.994	7.73149	**9.44E-12**	8.14141	**1.27E-12**	6.91033	**4.92E-10**

- N/A : 두 비교 집단의 분산이 모두 0이므로 검정 불가
- 볼드체는 유의 수준 5% 이내



(a) 인스턴스 (b) 기본 배열 (c) BFS 재배열

그림 4 그래프 분할에서의 재배치 결과 예 (인스턴스 U500.10)

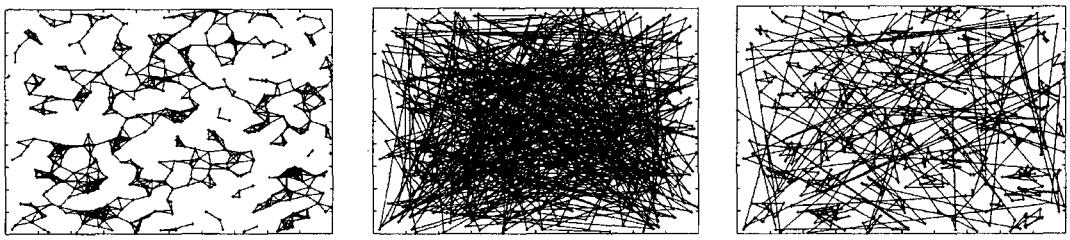
표 3 선형 배열 문제에 대한 실험 결과 (50회 시행)

그래프	기본 배열		BFS 재배열		DFS 재배열		Max-Adj 재배열	
	평균	표준편차	평균	표준편차	평균	표준편차	평균	표준편차
U500.05	162852.40	6.777E+03	158473.18	5.517E+03	159004.50	6.271E+03	158562.42	6.184E+03
U500.10	317022.28	1.027E+04	306477.68	1.280E+04	308955.18	1.152E+04	307840.62	1.184E+04
U1000.05	658743.66	1.473E+04	643714.92	1.815E+04	647381.98	2.078E+04	643003.50	1.698E+04
U1000.10	1355660.12	2.648E+04	1333089.58	3.701E+04	1339380.60	3.193E+04	1335310.82	3.189E+04

표 4 선형 배열 문제에서 기본 배열 결과와의 t-검정(양측) 결과

그래프	BFS 재배열		DFS 재배열		Max-Adj 재배열	
	t-값	p-값	t-값	p-값	t-값	p-값
U500.05	3.54345	**0.00061**	2.94691	**0.00401**	3.30644	**0.00132**
U500.10	4.54188	**1.59E-05**	3.69636	**0.00036**	4.14204	**7.30E-05**
U1000.05	4.54583	**1.56E-05**	3.15353	**0.00214**	4.95134	**3.07E-06**
U1000.10	3.50699	**0.00069**	2.77493	**0.00661**	3.47139	**0.00077**

- 볼드체는 유의 수준 5% 이내



(a) 인스턴스 (b) 기본 배열 (c) BFS 재배열

그림 5 선형 배열에서의 재배치 결과 예 (인스턴스 U500.05)

5.3 순회 판매원 문제

표 5는 TSPLIB [24]의 네 개의 인스턴스에 관한 50회 시행에 따른 해밀토니안 사이클의 가중치의 평균과 표준 편차를 나타내며, 표 6은 t-검정 결과를 나타낸다. 그 결과는 앞서 두 가지 실험 결과와 일치한다. "BFS", "DFS", "Max-Adj"에 의해 전처리된 유전 알고리즘은 "기본 배열"에 비해 좋은 해를 발견할 더 많은 기회를 갖는다. 그림 6은 인스턴스 att532에 대한 유전자 배열에 관한 결과 예를 보여준다. 이 그림에서 가까이 인접한 도시들이 BFS 재배열에 의해 염색체에서 가까이 배치되는 경향을 확인할 수 있다. 한편, 실제 구해진 TSP 투어를 시각화함으로써 재배열의 효과를 관찰할 수 있다. 그림 7은 그림 6의 재배열을 가지고 수행된 유전 알고리즘이 찾아낸 투어들을 나타낸다. 유전 알고리즘은 유전자들의 배열에 따라 상당히 다른 투어를 찾아냄을 확인할 수 있다.

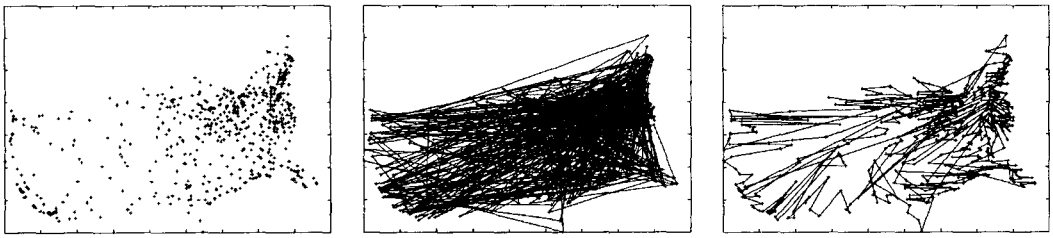
표 5 순회 판매원 문제에 대한 실험 결과 (50회 시행)

그래프	기본 배열		BFS 재배열		DFS 재배열		Max-Adj 재배열	
	평균	표준편차	평균	표준편차	평균	표준편차	평균	표준편차
lin318	43649.28	288.62	42588.30	268.45	42752.66	353.23	42533.30	267.43
pcb442	53405.06	366.64	52114.08	526.36	51969.56	451.30	51654.20	207.40
att532	29295.76	169.11	28375.08	217.97	28525.68	226.04	28382.66	210.12
rat783	9509.70	55.08	9304.24	94.05	9242.20	95.76	9182.46	94.19

표 6 순회 판매원 문제에서 기본 배열 결과와의 t-검정(양측) 결과

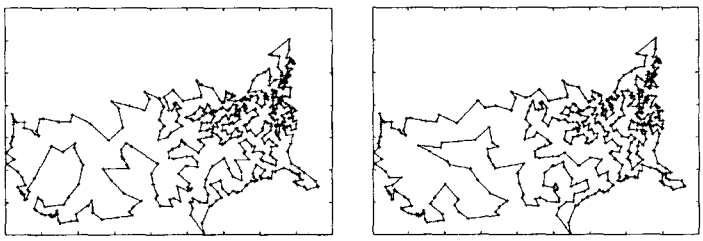
그래프	BFS 재배열		DFS 재배열		Max-Adj 재배열	
	t-값	p-값	t-값	p-값	t-값	p-값
lin318	19.03316	**1.09E-34**	13.89915	**6.57E-25**	20.05509	**1.83E-36**
pcb442	14.23094	**1.39E-25**	17.45691	**7.73E-32**	29.39083	**2.12E-50**
att532	23.59841	**3.29E-42**	19.28908	**3.88E-35**	23.93822	**9.96E-43**
rat783	13.3292	**9.76E-24**	17.12258	**3.24E-31**	21.20609	**2.13E-38**

- 볼드체는 유의 수준 5% 이내



(a) 인스턴스 (b) 기본 배열 (c) BFS 재배열

그림 6 순회 판매원 문제에서의 재배치 결과 예 (인스턴스 att532)



(a) 그림 6(b)에 의한 TSP 해 (b) 그림 6(c)에 의한 TSP 해

그림 7 다른 유전자 배치에 의한 TSP 해 (인스턴스 att532)

5.4 분산 척도

지금까지의 실험 결과로 주어진 문제와 독립적인 유전자 재배치를 통해 유전 알고리즘의 성능을 개선할 수 있음을 볼 수 있었다. 앞 절에서의 실험 결과는 유전자 재배치가 유전 알고리즘의 성능을 평균적으로 상당히 개선할 수 있음을 보여 주었으나 그 결과를 보면 문제 인스턴스에 따라 크게 성능을 개선할 수 있는 경우가 있는가 하면 유전자 재배치를 해도 성능을 그다지 개선할 수 없는 경우가 있음을 볼 수 있었다. 실제로 알고리즘을 수행하지 않고 유전 알고리즘의 성능을 어느 정도나 개선할 수 있는지를 미리 측정할 수 있는 척도는 매우 중요하다고 할 수 있다. 이 절에서는 직접 유전 알고리즘을 수행하기에 앞서 유전자 재배치를 통해 대략 어느 정도의 성능 개선이 있을 수 있는지를 가늠하는 척도를 제안하고 실제 그 척도와 성능 개선 정도 사이의 관계를 비교해 본다. 4절에서 각 유전자 쌍의 거리를 구하였었는데, 성능 개선에 관한 척도로서 모든 유전자 쌍의 거리 값들[1]의 표준 편차값을 사용한다. 그 척도를 분산 척도라 부르기로 한다. 이 분산 척도로 유전자 재배치를 통한 대략의 성능 개선 정도를 예측할 수 있다. 한 예로 그림 8은 그래프 분할 문제에 대해 21개 인스턴스들의 분산 척도와 성능 개선 정도 사이의 관계를

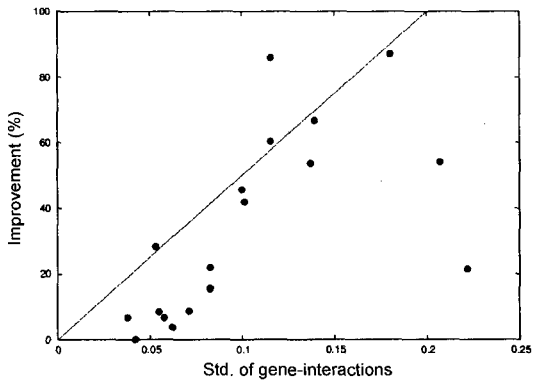


그림 8 분산 척도와 성능 개선률 사이의 관계

보여준다. 여기에서 성능 개선 정도를 다음과 같이 정의한다.

$$\text{성능개선정도}(\%) = 100 \times \frac{\text{일반 유전 알고리즘 성능} - \text{유전자 재배치 유전 알고리즘 성능}}{\text{일반 유전 알고리즘 성능} - \text{최적해}}$$

그림 8은 분산 척도가 성능 개선 정도와 매우 잘 일치함을 보여준다. 이 결과로부터 분산 척도는 문제 인스턴스의 특성과 대략의 유전 알고리즘의 성능 개선에 대한 가이드라인을 보여 주는 장점들을 가지고 있고 유용하게 쓰일 수 있다.

6. 결 론

본 논문에서 위치기반 인코딩에서 유전자들을 정적으로 재배치하는 프레임워크를 제안하였다. 그것은 재배치

1) 일반적으로 유전자 쌍의 거리는 문제의 크기 n이 커짐에 따라 커지게 된다. 즉, 문제의 크기에 의존하게 된다. 이 문제점을 극복하기 위해 문제의 크기 n에 따라 결정되는 유전자 쌍의 거리에 대한 기대값을 원래의 값에서 나눈 변환값을 사용한다.

없는 유전 알고리즘에 비해 일관된 성능 향상을 보여주었다. 기존의 연구에서처럼, 각 문제에 관해 해당 문제의 한정된 정보를 잘 활용함으로써 더 나은 재배열을 할 수 있을지 모른다. 본 논문에서 제안한 방법의 가장 중요한 특징은 재배치 과정 중에 해당 문제에 한정된 어떤 정보도 필요하지 않다는 점이다. 유전 알고리즘에 대해 새로운 문제가 주어질 때 새로운 전처리 휴리스틱을 개발할 필요가 없으며 유일하게 필요한 사항은 그 문제에 적합한 위치기반 인코딩이 무엇인지를 결정하는 것이다. 인코딩이 정해지면 그 인코딩에 적합한 일반적인 거리를 통해 유전자 상관성을 측정할 수 있다. 물론, 그것이 그 문제에 가장 적당한 거리 척도인지 아니면 어떤 다른 거리 척도가 더 적합한지를 파악하는 문제가 어렵지만 이는 중요한 앞으로의 과제가 될 수 있을 것이다. 그러나 다행히도, 현재 유전 알고리즘 분야에서 중요하게 다루어지는 대부분의 문제가 몇 가지 대표적인 인코딩 중의 하나로 분류될 수 있으므로 본 논문에서 제안한 방법이 쉽게 적용될 수 있을 것이다. 또한 유전자 상관성에 관한 유용한 연구가 이미 많이 진행되어 왔으며 이러한 결과를 잘 활용할 여지도 있다[25-30].

이 연구에서는 오직 선형 인코딩만을 고려하였다. 비록 그것이 전통적이며 가장 많이 이용되는 인코딩이기는 하지만, 다차원 인코딩 역시 유전 알고리즘 분야에서 각광받고 있다[31,32]. 제안된 재배치 프레임워크는 그것이 오직 선형 인코딩에만 적용될 수 있다는 단점을 가지고 있다. 그 재배치 방법을 다차원 인코딩에 확장하는 것은 시도할 만한 가치가 있는 주제일 것이다.

참 고 문 헌

[1] J. Bagley. The Behavior of Adaptive Systems Which Employ Genetic and Correlation Algorithms. PhD thesis, University of Michigan, Ann Arbor, MI, 1967.

[2] G. R. Harik and D. E. Goldberg. Learning linkage. In Foundations of Genetic Algorithms 4, pages 247-262, 1996.

[3] D. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and rst results. Complex System, Vol. 3, pages 493-530, 1989.

[4] T. N. Bui and B. R. Moon. Hyperplane synthesis for genetic algorithms. In Fifth International Conference on Genetic Algorithms, pages 102-109, July 1993.

[5] T. N. Bui and B. R. Moon. Genetic algorithm and graph partitioning. IEEE Trans. on Computers, Vol. 45, No. 7, pages 841-855, 1996.

[6] T. N. Bui and B. R. Moon. A new genetic approach for the traveling salesman problem. In IEEE Conference on Evolutionary Computation, pages 7-12, June 1994.

[7] B. R. Moon and C. K. Kim. A two-dimensional embedding of graphs for genetic algorithms. In International Conference on Genetic Algorithms, pages 204-211, 1997.

[8] P. Merz and B. Freisleben. Memetic algorithms and the tness landscape of the graph bi-partitioning problem. In Proceedings of the 5th International Conference on Parallel Problem Solving From Nature, 1998. Lecture Notes in Computer Science, Vol. 1498, pages 765-774, Springer-Verlag.

[9] R. Battiti and A. Bertossi. Greedy, prohibition, and reactive heuristics for graph partitioning. IEEE Trans. on Computers, Vol. 48, No. 4, pages 361-385, 1999.

[10] Y. H. Kim and B. R. Moon. A hybrid genetic search for graph partitioning based on lock gain. In Genetic and Evolutionary Computation Conference, pages 167-174, 2000.

[11] M. Garey and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco, 1979.

[12] D. Adolphson and T. Hu. Optimal linear ordering. SIAM J. Appl. Math., Vol. 25, No. 3, pages 403-423, 1973.

[13] C. Cheng. Linear placement algorithms and applications to VLSI design. Networks, Vol. 17, pages 439-464, 1987.

[14] P. Merz and B. Freisleben. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In Proceedings of the Congress on Evolutionary Computation, Vol. 3, pages 2063-2070. IEEE Press, 1999.

[15] P. Merz and B. Freisleben. Genetic local search for the TSP: New results. In IEEE Conference on Evolutionary Computation, pages 159-164, 1997.

[16] Y. Nagata and S. Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. In 7th International Conference on Genetic Algorithms, pages 450-457, 1997.

[17] S. Jung and B. R. Moon. Toward minimal restriction of genetic encoding and crossovers for the 2D Euclidean TSP. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 6, pages 557-565, 2002.

[18] D. I. Seo and B. R. Moon. Voronoi quantized crossover for traveling salesman problem. In Genetic and Evolutionary Computation Conference, pages 544-552, 2002.

[19] D. Goldberg and R. Lingle. Alleles, loci, and the traveling salesman problem. In First International Conference on Genetic Algorithms and Their Applications, pages 154-159, 1985.

[20] C. H. Papadimitriou and K. Steiglitz. Combinatorial Optimization: Algorithms and Complexity, Dover

Publications, Inc., Mineola, New York, 1998.
[21] C. Alpert and A. B. Kahng. A general framework for vertex orderings, with applications to netlist clustering. In IEEE/ACM International Conference on Computer-Aided Design, pages 63-67, 1994.
[22] D. S. Johnson, C. Aragon, L. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation, Part 1, graph partitioning. Operations Research, Vol. 37, pages 865-892, 1989.
[23] T. N. Bui and B. R. Moon. A genetic algorithm for a special class of the quadratic assignment problem. The Quadratic Assignment and Related problems, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 16, pages 99-116, 1994.
[24] http://www.iwr.uniheidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html
[25] Y. Davidor. Epistasis variance: A viewpoint on ga-hardness. In Foundations of Genetic Algorithms 3, pages 23-35. Morgan Kaufmann, 1991.
[26] C. Reeves and C. Wright. An experimental design perspective on genetic algorithms. In Foundations of Genetic Algorithms 3, pages 7-22. Morgan Kaufmann, 1995.
[27] C. Reeves and C. .Wright. Epistasis in genetic algorithms: An experimental design perspective. In Proceedings of the Sixth International Conference on Genetic Algorithms, pages 217-224. Morgan Kaufmann, 1995.
[28] C. Fonlupt, D. Robilliard, and Philippe Preux. A bit-wise epistasis measure for binary search spaces. Lecture Notes in Computer Science, Vol. 1498, pages 47-56, 1998.
[29] M. Munetomo and D. Goldberg. Identifying linkage by nonlinearity check, 1998.
[30] M. Pelikan, D. Goldberg, and F.o Lobo. A survey of optimization by building and using probabilistic model. Technical Report 99018, IlliGAL, September 1999.
[31] T. N. Bui and B. R. Moon. On multi-dimensional encoding/crossover. In Sixth International Conference on genetic Algorithms, pages 49-56, 1995.
[32] A. B. Kahng and B. R. Moon. Toward more powerful recombinations. In Sixth International Conference on genetic Algorithms, pages 96-103, 1995.

권 영 근
1999년 서울대학교 전산과학전공 학사 2001년 서울대학교 전기컴퓨터공학부 석사. 현재 서울대학교 전기컴퓨터공학부 박사과정. 관심분야는 최적화 이론, 시스템근사화, 금융 공학

김 용 혁
정보과학회논문지 : 소프트웨어 및 응용 제 32 권 제 5 호 참조

문 병 로
정보과학회논문지 : 소프트웨어 및 응용 제 32 권 제 5 호 참조