

2상 양자 진화 알고리즘

한국현

한국과학기술원 전자전산학과

Abstract- This paper proposes a novel evolutionary algorithm, called a two-phase quantum-inspired evolutionary algorithm (TPQEA), which can improve the performance of QEA (quantum-inspired evolutionary algorithm). From recent research on combinatorial optimization, the results of QEA with population size 1 were better than those of CGA(conventional GA) with population size 100. More specifically, the convergence speed of QEA was 70 times faster than that of CGA. Although the performance of QEA is excellent, this paper shows that its performance can be higher. Through the analysis of changing the initial Q-bit value and the introduction of a new termination criterion and a H_ϵ gate, a two-phase QEA (TPQEA) is proposed. To demonstrate the effectiveness and applicability of TPQEA, several experiments are carried out on combinatorial and numerical optimizations. The results show that TPQEA outperforms QEA in convergence speed, robustness and the power of a global/local search.

I. 서론

진화 알고리즘(evolutionary algorithms)은 기본적으로 확률적 검색을 기반으로 하는 최적화 기법이다. 특히, 자연의 생물학적 진화 원리에 기반을 두고 있으며, 최적화 문제에 대해 특정 휴리스틱이 없이도 적용이 가능하다는 범용성을 가장 큰 장점으로 갖는다. 이와 같은 진화 알고리즘은 이론적 배경이 부족하다는 사실에도 불구하고, 실험적 결과들은 기대 이상을 보여주고 있으며, 실제로 많은 분야에서 최적화 도구로 사용되어지고 있다. 특히, 수식적 연산 방법으로는 최적 해를 찾기 어려운 문제들에 주로 적용되고 있다. 많은 사람들이 최적화 도구로 사용하고 있는 진화 알고리즘으로는 크게 유전자 알고리즘(GA: genetic algorithm)[1, 2, 3], 진화 프로그래밍(EP: evolutionary programming)[4], 진화 전략(ES: evolution strategies)[5, 6] 등이 있다. 해의 표현법이나

진화 연산자 등에 있어서 약간의 차이는 있지만, 세 가지 알고리즘의 기본 원리와 개념 등은 매우 유사하다.

진화 알고리즘의 탐색에 있어서 가장 중요한 요소 중의 하나는 전역 탐색(exploration)과 지역 탐색(exploitation)의 균형이다. 전역 탐색만을 수행할 경우에는 무작위 검색과 같은 결과를 나타내며, 지역 탐색만을 수행할 경우에는 국부 최적 해를 찾는 결과를 초래하게 된다. 진화 알고리즘의 구현 측면에서 보면, 개체 집단의 다양성과 다음 세대의 높은 선택 강도가 각각 전역 탐색과 지역 탐색에 해당된다. 선택 강도의 증가는 개체 집단의 다양성을 감소시키고, 선택 강도의 감소는 반대로 개체 집단의 다양성을 증가시킨다[7]. 따라서 검색 초기에는 개체 집단의 다양성을 강조하고, 검색이 진행될수록 점차로 선택 강도를 높여 지역 탐색을 강조해야 한다. 하지만 이와 같은 선택 강도의 조절은 진화 알고리즘의 사용자에게 가장 큰 어려움으로 남게 된다. 이와 같은 문제점을 해결하기 위해 새로운 진화 알고리즘인 양자 진화 알고리즘(이하 QEA: quantum-inspired evolutionary algorithm)이 최근에 제안되었다[8]. QEA는 알고리즘의 구조적 특성상 전역 탐색과 지역 탐색의 균형이 용이하게 이루어진다. QEA는 진화 연산과 유사성을 보이는 양자 전산(quantum computing)의 개념을 도입함으로써 기존의 진화 알고리즘보다 우수한 성능을 보일 수 있는 알고리즘으로 제안되었다. QEA는 양자 비트, 상태의 중첩 등과 같은 양자 전산의 기본 원리를 이용한다. QEA는 양자의 확률적 표현법을 이용함으로써 개체들의 내력을 모두 저장할 수 있으며, 초기 전역 탐색에서 최적해로 수렴함에 따라 점차로 지역 탐색을 수행하는 기능을 갖고 있다. 또한 알고리즘의 종료 조건을 결정할 수 있는 명확한 지표 값을 설정할 수 있고, 많은 개체를 동시에 표현할 수 있는 특징에 의해 적은 개체만을 이용하여 우수한 해를 짧은 시간 내에 얻을 수 있다는 장점을 갖는다.

양자 전산이란 양자 컴퓨터와 양자 알고리즘을 포함하는 연구 분야이다. 양자 컴퓨터는 80년대 초에 기본 개념이 제안되었고[9, 10], 현재 양자 컴퓨터로 여겨지는 구조의 구체적인 청사진은 80년대 후반에 제시되었다[11, 12]. 양자 전산에 대한 연구는 90년대 초반부터 발표되기 시작한 양자 알고리즘에 의해 관심과 지원이 증폭되었다. 현재까지 잘 알려진 양자 알고리즘으로는 Deutsch-Jozsa 알고리즘[13], Simon 알고리즘[14], Shor의 소인수 분해 알고리즘[15, 16], Grover의 데이터베이스 검색 알고리즘[17, 18] 등이 있다. 특히, 양자 전산은 Shor의 소인수 분해 알고리즘의 등장으로 현재 널리 사용되고 있는 암호 체계가 붕괴될 수 있다는 사실이 알려지면서 선진국

들의 폭발적 관심을 받게 되었다. 이와 같이 양자 전산의 우수한 성능이 입증되면서 다른 분야에서도 양자 전산의 개념을 도입하려는 움직임이 시작되었고, 진화 연산 분야에서도 양자 전산과의 접목된 연구가 시작되었다.

진화 연산과 양자 전산의 접목된 연구는 크게 두 가지 분야로 나뉘어진다. 첫 번째는 유전자 프로그래밍(genetic programming)과 같은 자동 프로그래밍 기법을 이용하여 새로운 양자 알고리즘을 개발하는 연구 분야이고[19, 20, 21], 두 번째는 양자 전산 개념을 도입하여 새로운 진화 알고리즘을 개발하는 연구 분야이다[22, 23, 24, 8]. QEA는 후자의 연구 분야에 속하게 된다.

진화 알고리즘에 양자 전산 개념을 적용한 연구는 다른 연구 분야와는 다르게 거의 시작 단계에 있다고 볼 수 있다. 이 연구 분야의 시초는 “quantum-inspired computing”이라는 이름으로 비물리학자들에게 양자 전산이 소개되면서부터라고 볼 수 있다[25]. 양자 전산이 소개된 이후 처음으로 양자 개념을 도입한 유전자 알고리즘이 발표되었다[22]. 하지만, 유전자 알고리즘의 교배(crossover) 연산자에 양자의 간섭 개념을 적용한 비교적 간단한 알고리즘이었다. 그 후, 진화 알고리즘에 양자 전산의 가장 중요한 요소 중의 하나인 상태의 중첩(superposition) 개념이 적용된 연구가 처음으로 발표되었고[23], 중첩된 개체는 병렬 진화 알고리즘 구조를 효율적으로 활용할 수 있다는 연구 결과가 발표되었다[24]. 이와 같은 연구 결과[23, 24]를 바탕으로 기존의 진화 알고리즘보다 우수한 성능을 보이는 QEA가 발표되었다[8]. 조합 최적화의 대표적인 문제인 주머니 문제(kanpsack problem)의 경우, 100개의 개체를 사용한 유전자 알고리즘의 결과보다 1개의 개체를 이용한 QEA의 결과가 더 우수했고, 이때 연산 시간은 QEA가 대략 유전자 알고리즘의 70분의 1 정도 밖에 소요되지 않았다[8]. 특히, 실제 응용 문제인 여러 개의 디스크에 데이터를 효율적으로 저장하는 디스크 할당 문제에 QEA가 적용된 연구가 진행되었다[26]. 이 연구에서는 현존하는 휴리스틱 알고리즘보다 유전자 알고리즘의 결과가 더 우수했고, 유전자 알고리즘의 결과보다는 QEA의 결과가 우수했다는 점을 보이고 있다. 또한, QEA가 더 좋은 해를 찾아주었음에도 불구하고, 수렴 속도 면에서는 유전자 알고리즘보다 3배에서 최대 11배까지 빠르다는 결과를 보여주었다.

본 논문에서는 기존의 진화 알고리즘에 비해 최근 우수한 성능을 보이고 있는 QEA 알고리즘의 성능을 보다 더 향상시킬 수 있는 “2상 양자 진화 알고리즘”(이하 TPQEA: two-phase quantum-inspired evolutionary algorithm)을 제안하고자 한다. 그림 1은 TPQEA와 QEA가 양자 전산 개념을

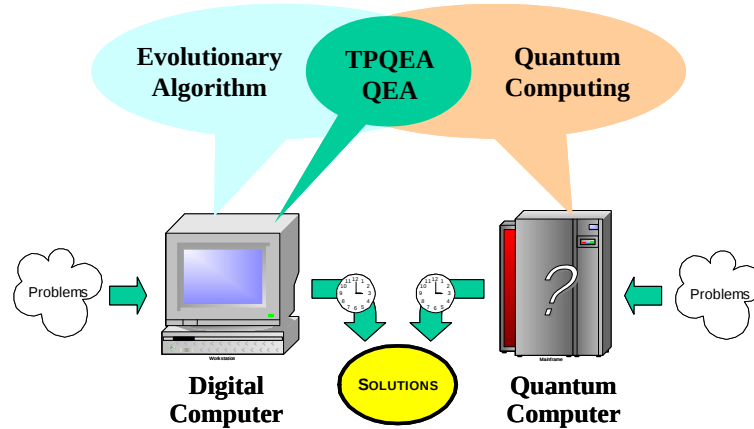


그림 1: TPQEA와 QEA의 개념도.

도입한 새로운 진화 알고리즘임을 나타낸다.

본 논문의 구성은 다음과 같다. II장에서는 양자 진화 알고리즘(QEA)에 대해 설명하고, 성능을 검증한다. III장에서는 본 논문에서 제안하고자 하는 2상 양자 진화 알고리즘(TPQEA)에 대해 설명하고, 그 특성을 분석한다. IV장에서는 조합 최적화와 수치 최적화 문제에 대한 실험을 수행하고 결과를 분석한다. 마지막으로 V장에서 결론을 맺는다.

II. 양자 진화 알고리즘 (QEA: quantum-inspired evolutionary algorithm)

본 장에서는 TPQEA 알고리즘을 제안하기에 앞서, 조합 최적화 문제를 위해 제안된 QEA 알고리즘[8]을 체계적으로 정리하고, 성능을 검증한다. QEA는 새로운 Q-비트(Q-bit) 표현법과 Q-게이트(Q-gate) 연산자, 그리고 관측 과정 등으로 설계된다.

1. 표현법 (Representation)

진화 알고리즘에서 해를 코딩하기 위한 표현법으로는 여러 가지 방법이 있다. 기존의 표현법은 크게 이진(binary) 표현법, 숫자(numeric) 표현법, 기호(symbolic) 표현법으로 나누어 생각할 수 있다[27]. QEA는 새로운 표현법인 Q-비트 표현법을 정의하여 사용한다. Q-비트는 확률적인 표현법으로 양자 전산의 큐비트(qubit) 개념에 기반을 두고 있다.

Q-비트는 QEA에서 정보를 표현하는 최소 단위로서 두 복소수 (α, β) 로 정의하며, 다음과 같

이 표현한다:

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

이때 $|\alpha|^2 + |\beta|^2 = 1$ 의 조건을 항상 만족하며, $|\alpha|^2$ 은 Q-비트가 0의 상태일 확률을 나타내고, $|\beta|^2$ 은 Q-비트가 1의 상태일 확률을 나타낸다. Q-비트는 (α, β) 값에 따라 1의 상태나 0의 상태 또는 1과 0의 중첩된 상태를 표현할 수 있다. 참고로, Q-비트는 복소수로 정의되어 있으나, 현재까지 발표된 연구에서는 실수만을 사용하고 있다.

Q-비트 m 개가 연속되어 만들어진 개체를 Q-비트 개체(Q-bit individual)로 정의하며, 다음과 같이 표현된다:

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{array} \right]. \quad (1)$$

이때 $|\alpha_i|^2 + |\beta_i|^2 = 1, i = 1, 2, \dots, m$, 조건을 만족한다.

Q-비트 표현법은 여러 상태의 중첩을 표현할 수 있는 장점을 갖는다. 예를 들어, 다음과 같은 3-Q-비트 시스템이 있다고 가정하자:

$$\left[\begin{array}{c|c|c} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} & \frac{\sqrt{3}}{2} \end{array} \right]. \quad (2)$$

이 시스템의 상태는 다음과 같이 표현할 수 있다:

$$\begin{aligned} & \frac{1}{4}|000\rangle + \frac{\sqrt{3}}{4}|001\rangle - \frac{1}{4}|010\rangle - \frac{\sqrt{3}}{4}|011\rangle \\ & + \frac{1}{4}|100\rangle + \frac{\sqrt{3}}{4}|101\rangle - \frac{1}{4}|110\rangle - \frac{\sqrt{3}}{4}|111\rangle. \end{aligned}$$

위의 결과는 8 가지의 상태 $|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle$ 가 각각 $\frac{1}{16}, \frac{3}{16}, \frac{1}{16}, \frac{3}{16}, \frac{1}{16}, \frac{3}{16}, \frac{1}{16}, \frac{3}{16}$ 의 확률로 중첩된 상태를 나타낸다. 즉, (2)와 같은 3-Q-비트 시스템은 동시에 8 가지의 상태를 나타낼 수 있다는 것을 의미한다.

Q-비트 표현법을 갖는 진화 알고리즘은 위와 같이 여러 상태의 중첩을 표현할 수 있기 때문에 기존의 접근 방법보다 더 좋은 다양성(diversity)을 제공할 수 있다. Q-비트 표현법에서는 (2)와 같은 단 하나의 개체가 8 가지 상태를 표현하기에 충분하지만, 기존의 표현법에서는 적어도 8 가지의 개체 (000), (001), (010), (011), (100), (101), (110), (111)를 필요로 하게 된다. 또한 Q-비트 표현법은 표현법 자체에서 좋은 수렴성(convergence)을 보이게 된다. $|\alpha_i|^2$ 또는 $|\beta_i|^2$ 의 값이 1 또는

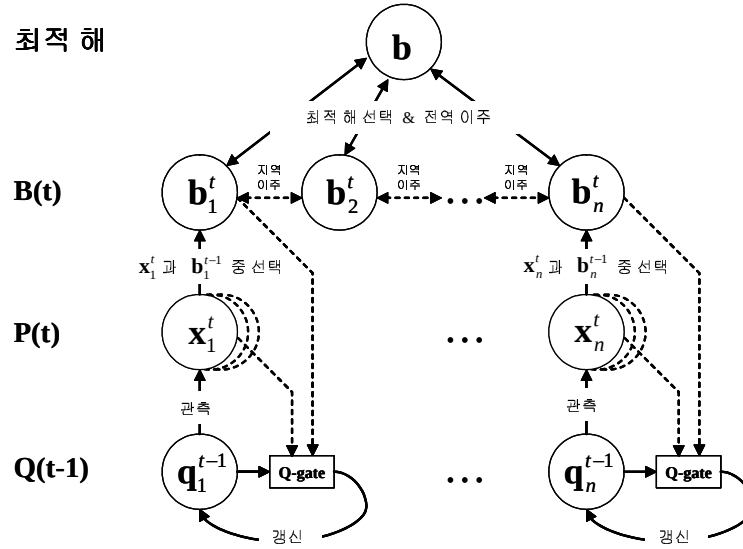


그림 2: QEA에서의 변수들 간의 관계.

0으로 점근함에 따라 Q-비트 개체는 하나의 상태로 수렴해 가게 되며, 점차로 다양성(diversity)은 사라지게 된다. 즉, Q-비트 표현법은 진화 알고리즘에서의 탐사(exploration)와 개척(exploitation) 두 가지 특성의 균형을 자연스럽게 이룰 수 있다.

2. QEA 알고리즘의 기본 구조

QEA는 기존의 진화 알고리즘과 유사한 확률 기반의 최적화 알고리즘이다. 하지만, QEA에서 사용되는 변수들을 보면 기존의 진화 알고리즘과는 큰 차이가 있음을 확인할 수 있다. QEA에서 해를 표현하기 위한 개체군은 Q-비트 개체들의 집합으로 표현하며, 다음과 같이 표기한다:

$$Q(t) = \{q_1^t, q_2^t, \dots, q_n^t\}. \quad (3)$$

이때 t 는 현재 세대(generation), n 은 총 개체수(population size)를 나타낸다. 또한 q_j^t 는 하나의 Q-비트 개체를 의미하고, 다음과 같이 표현될 수 있다:

$$q_j^t = \left[\begin{array}{c|c|c|c} \alpha_{j1}^t & \alpha_{j2}^t & \dots & \alpha_{jm}^t \\ \beta_{j1}^t & \beta_{j2}^t & \dots & \beta_{jm}^t \end{array} \right]. \quad (4)$$

이때 m 은 Q-비트 개체를 이루는 Q-비트의 개수이고, $j = 1, 2, \dots, n$ 이다. Q-비트 개체는 확률적 표현법으로서 기존의 진화 알고리즘의 결정적인(deterministic) 표현법과는 큰 차이가 있다. 이는 해(solution)와 개체간에 일대일 대응이 되어야 한다는 고정관념에서 벗어나, 해와 개체의 분리를 이룬 것이다. Q-비트 개체는 세대가 지남에 따라 해(solution)의 내력 정보를 포함하게 된다.

Procedure QEA**begin** $t \leftarrow 0$ i) $Q(t)$ 초기화ii) $Q(t)$ 상태 관측에 의한 $P(t)$ 생성iii) $P(t)$ 평가iv) $P(t)$ 중 최적 해를 $B(t)$ 에 저장v) **while** (not 종료 조건) **do****begin** $t \leftarrow t + 1$ vi) $Q(t-1)$ 상태 관측에 의한 $P(t)$ 생성vii) $P(t)$ 평가viii) $Q(t)$ 갱신 (Q-게이트 연산자 이용)ix) $B(t-1)$ 과 $P(t)$ 중 개체별 최적 해를 $B(t)$ 에 저장x) $B(t)$ 중 최적 해를 $\mathbf{b}$ 에 저장xi) **if** (전역 이주 조건)**then** 최상위 최적 해 $\mathbf{b}$ 를 모든 개체별 최적 해 집합 $B(t)$ 로 이주 (전역 이주)xii) **else if** (지역 이주 조건)**then** 지역 그룹 내에서의 개체별 최적 해 간의 이주 (지역 이주)**end****end**

그림 3: QEA 알고리즘.

그림 2는 QEA에서 사용되는 모든 변수들 간의 관계를 나타낸다. Q-비트 개체 $Q(t)$ 는 확률적으로 상태를 표현하기 때문에 개체를 직접 평가하는 것이 불가능하다. 개체의 평가를 위해 Q-비트 개체 $\mathbf{q}_j^t$ 를 관측하여 이진(binary) 개체 $\mathbf{x}_j^t$ 를 얻고, 관측된 이진 개체를 평가함으로써 Q-비트 개체의 평가를 대신한다. 이때 1회 이상의 관측도 가능하다. 이진 개체들의 집합은 $P(t)$ 로 표현하며, 각 개체별 최적 해의 집합을 $B(t)$ 로 표현한다. 또한 개체별 최적 해의 집합 중에서 가장 최적의 해를 $\mathbf{b}$ 로 표기한다.

QEA 알고리즘의 기본 구조는 그림 3의 수도 코드로 표현되고, 이때 QEA 알고리즘에 대한 설명은 다음과 같다.

i) ‘ $Q(t)$ 초기화’ 과정에서는 모든 Q-비트 개체 $\mathbf{q}_j^0$ 의 각 Q-비트 값 (α_i^0, β_i^0) 을 모두 $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ 로 설정한다. 이것은 각 Q-비트 개체 $\mathbf{q}_j^0$ 가 모든 가능한 해를 같은 확률로 선형 중첩된 상태로 포함한다는 것을 의미한다.

ii) 이 과정에서는 확률적인 상태를 갖는 각 Q-비트 개체를 관측함으로써 이진(binary) 개체를

얻는다. 이것은 확률적인 Q-비트 개체를 직접 평가하는 것이 불가능하기 때문에, 관측된 이진 개체로 대신 평가하기 위함이다. 관측 과정은 각 비트별로 이루어지며, 0과 1 사이의 난수를 발생시켜 $|\beta_i|^2$ 과 비교하여 해당 비트가 0 또는 1인지를 결정하게 된다.

iii) ‘ $P(t)$ 평가’에서는 관측된 각 이진 해(solution)들이 적합도 함수를 통해 평가된다.

iv) 초기의 개체별 최적 해 집합 $B(0)$ 에는 개체별로 가장 우수한 이진 해가 저장된다.

v) 종료 조건을 만족할 때까지 **while** 루프를 반복하게 된다. 이때 QEA의 종료 조건은 다음과 같이 최적 해 $\mathbf{b}$ 의 평균 확률로 정의될 수 있다:

$$Prob(\mathbf{b}) = \frac{1}{n} \sum_{j=1}^n \left(\prod_{i=1}^m p_{ji} \right). \quad (5)$$

이때

$$p_{ji} = \begin{cases} |\alpha_{ji}|^2, & \text{if } b_i = 0 \\ |\beta_{ji}|^2, & \text{if } b_i = 1 \end{cases}$$

이고, 최적 해 $\mathbf{b} = [b_1 \ b_2 \ \cdots \ b_m]^T$, $(\alpha_{ji}, \beta_{ji})$ 는 수식 (4)와 같이 j 번째 Q-비트 개체의 i 번째 Q-비트를 나타낸다. 즉, 최적 해의 평균 확률을 이용한 종료 조건은 다음과 같이 나타낼 수 있다:

$$Prob(\mathbf{b}) > \gamma. \quad (6)$$

이때 γ 는 조건 $0 < \gamma < 1$ 을 만족한다.

vi, vii) $Q(t-1)$ 의 각 Q-비트 개체 $\mathbf{q}_j^{t-1}$ 의 상태를 관측하여 이진 해 집합 $P(t)$ 를 얻고, 각 이진 해 $\mathbf{x}_j^t$ 를 평가한다. 이때 Q-비트 개체 $\mathbf{q}_j^{t-1}$ 를 1회 이상 관측할 수 있으며, 만약 N_{ob} 회 관측하였을 경우에는 관측된 이진 해의 표기를 $\mathbf{x}_{jl}^t, l = 1, 2, \dots, N_{ob}$,로 바꾸어야 한다.

viii) ‘ $Q(t)$ 갱신’과정에서는 QEA의 변환 연산자인 Q-게이트(Q-gate)를 이용하여 Q-비트 개체가 보다 좋은 해의 방향으로 수렴해 나가도록 각 Q-비트들을 변환시킨다. QEA의 기본 Q-게이트로는 회전(rotation) 게이트를 사용하지만, NOT, Hadamard, Controlled-NOT 게이트 등도 이용이 가능하다. 기본 Q-게이트인 회전 게이트는 다음과 같다:

$$U(\Delta\theta_i) = \begin{bmatrix} \cos(\Delta\theta_i) & -\sin(\Delta\theta_i) \\ \sin(\Delta\theta_i) & \cos(\Delta\theta_i) \end{bmatrix}. \quad (7)$$

이때 $\Delta\theta_i, i = 1, 2, \dots, m$,는 부호에 따라 각 Q-비트를 0 또는 1의 상태로 이동시키는 회전각을 나타낸다. $\Delta\theta_i$ 는 문제에 따라 결정되는 사용자 설정 변수이고, 각 비트별로 개체별 최적 해 $\mathbf{b}_j^t$ 의 i 번

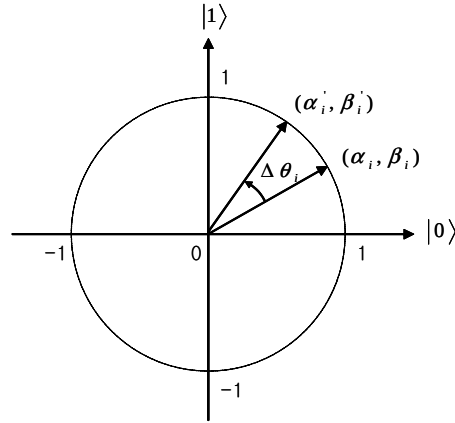


그림 4: 기본 Q-게이트: 회전 게이트(Rotation gate).

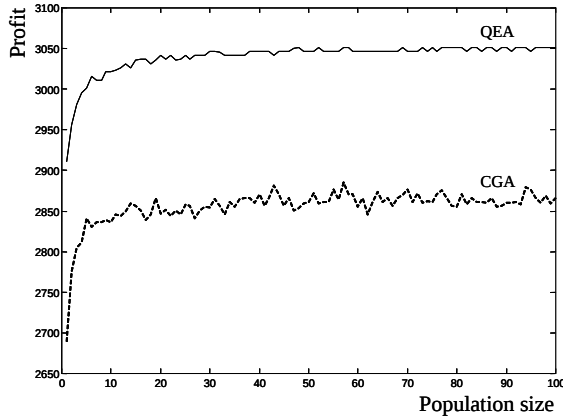
째 비트 b_i 와 이전 해 $\mathbf{x}_j^t$ 의 i 번째 비트 x_i , 그리고 $\mathbf{b}_j^t$ 와 $\mathbf{x}_j^t$ 의 적합도 크기에 따라 $\Delta\theta_i$ 의 부호가 결정된다. 또한 회전각의 크기 $|\Delta\theta_i|$ 는 Q-비트 개체의 수렴 속도를 결정하게 된다. 하지만, 너무 큰 값을 설정할 경우에는 발산 또는 조기 수렴(premature convergence)을 보일수 있다. 적합도 함수가 $f(\cdot)$ 일 경우, 기본적으로 $\Delta\theta_i$ 는 $f(\mathbf{x}_j) \leq f(\mathbf{b}_j)$ 의 조건을 만족할 경우에만 ($x_i = 0, b_i = 1$ 일 경우) 양수 p 또는 ($x_i = 1, b_i = 0$ 일 경우) 음수 n 을 할당한다. 나머지 경우에는 모두 0으로 설정한다.

ix, x) 이전 세대의 개체별 최적 해 $\mathbf{b}_j^{t-1}$ 와 이번 세대의 이전 해 $\mathbf{x}_j^t$ 중에서 좋은 해를 이번 세대의 개체별 최적 해 $\mathbf{b}_j^t$ 로 저장한다. 그리고 이번 세대의 개체별 최적 해 집합 $B(t)$ 중에서 가장 최적의 해를 최상위 최적 해 $\mathbf{b}$ 에 저장한다.

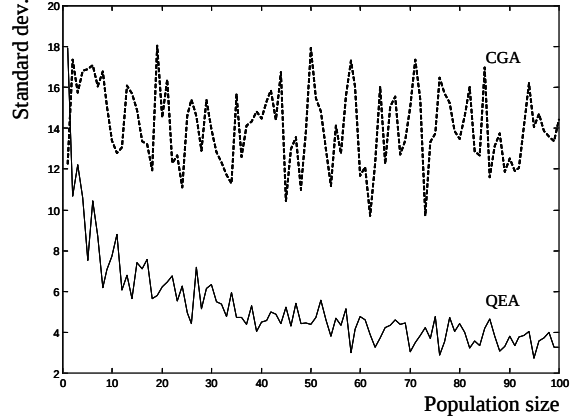
xi) 전역 이주(global migration) 조건을 만족할 경우, 최상위 최적 해 $\mathbf{b}$ 를 모든 개체별 최적 해 $B(t)$ 로 복사한다. 이때 이주(migration)는 하나의 이전 개체가 다른 위치로 복사되는 과정으로 정의된다. 전역 이주는 세대의 변화에 따라 주기적으로 설정한다. 예를 들어 전역 이주 주기 T_g 를 100으로 설정하였다면, 100 세대마다 전역 이주가 발생하게 된다.

xii) 지역 이주(local migration) 조건을 만족할 경우, 지역 그룹 내에서의 개체별 최적 해 간의 이주가 이루어진다. 지역 그룹 내에서 가장 우수한 해가 다른 개체별 최적 해 위치로 복사된다. 지역 이주 주기는 T_l 로 표기하고, 주로 1로 설정한다. 이때 지역 그룹(local-group)은 지역 이주에 의해 상호 영향을 받는 부분 개체군(subpopulation)으로 정의될 수 있다.

3. 주머니 문제(knapsack problem)를 통한 QEA의 성능 검증



(a) 평균 최상 이득



(b) 최상 이득의 표준 편차

그림 5: 총 개체수(population size) 변화에 따른 QEA와 CGA의 실험 결과 (평균 주머니 용량을 갖는 500개 아이탬의 주머니 문제): QEA의 전역 이주 주기 $T_g = 100$, 지역 그룹 크기는 총 개체수 n 에 따라 $\max(\text{integer}(\frac{n}{5}), 1)$ 로 설정하였다. CGA의 파라미터는 [8]에서와 동일하게 설정하였다. (a) 30번 실험에 대한 평균 최상 이득, (b) 30번 실험 결과의 표준편차를 나타낸다.

QEA의 성능을 검증하기 위해 평균 주머니 용량(average knapsack capacity)을 갖는 500개 아이탬의 주머니 문제에 대한 실험을 수행하였다. 주머니 문제에 대한 자세한 설명은 부록 A에서 다룬다. 그림 5는 결과의 비교를 위해 기존의 유전자 알고리즘(CGA: conventional GA)의 실험 결과를 함께 도시하였다. CGA는 [8]에서와 같이 선형 페널티(linear penalty) 방법과 우성 위주 보상(greedy repair)방법을 동시에 고려하였다. (a)의 결과에서 보면, 100개체를 갖는 CGA의 결과보다 1개체를 갖는 QEA의 결과가 더 우수함을 확인할 수 있다. 이때 수렴 속도를 측정한 결과, QEA가 CGA에 비해 약 70배가 더 빨랐다. (b)의 결과에서는 총 개체수가 증가함에 따라 QEA의 표준편차가 점차로 줄어들고 있는 것을 확인할 수 있다. 이것은 총 개체수가 증가할수록 QEA 동작의 강인성(robustness)이 증가한다는 것을 의미한다. 반면에 CGA의 결과는 총 개체수가 증가하여도 최상 해의 표준편차가 줄어들지 않았다. 본 실험 결과는 QEA가 CGA에 비해 해(solution)의 탐색 능력이나 수렴 속도, 그리고 강인성 측면에서 모두 우수함을 입증한다.

III. 2상 양자 진화 알고리즘 (TPQEA: two-phase QEA)

본 장에서는 II장에서 기술한 QEA 알고리즘의 성능을 보다 더 향상시킬 수 있는 알고리즘 구조를 제안한다: 1) QEA의 종료 조건은 II장의 수식 (6)으로 나타낼 수 있다[8]. 이와 같이 정의된

표 I: 종료 조건 (6)의 γ 값 변화에 따른 영향 (500개의 아이টে를 갖는 주머니 문제에 대한 실험 결과): 실험에 사용된 QEA의 개체군 크기(population size)는 12, 전역 이주 주기 $T_g = 100$, 지역 그룹 크기는 3으로 설정하였고, 총 30번의 실험을 수행하였다. $b.$, $m.$, $w.$ 는 각각 *best*, *mean*, *worst*를 의미하고, σ 와 t 는 각각 30번 실험 결과의 표준 편차와 알고리즘의 평균 종료 세대수이다.

γ of (6)		0.001	0.01	0.1	0.8	0.9
500	b.	3031.2	3036.2	3031.3	3031.3	3036.3
	m.	3008.0	3014.3	3019.0	3018.0	3020.3
	w.	2980.7	2991.1	3006.3	3006.3	3001.3
	σ	12.571	9.798	6.548	6.993	7.895
	t	905	1045	1240	1896	3071

종료 조건의 문제점을 살펴보고, 새로운 종료 조건을 제안한다; 2) QEA의 Q-비트 개체를 초기화 하는 과정은 모든 가능한 해를 같은 확률로 포함시키기 위해 각 Q-비트의 값을 $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ 로 설정 하였다. 여기서, Q-비트의 초기 값의 변화가 QEA의 성능에 큰 영향을 미칠 수 있음을 확인하고, TPQEA의 가능성을 제시한다; 3) 또한, Q-비트 개체가 국부 최적 해(local optima)로부터 효율적으로 빠져나올 수 있도록 하는 새로운 연산자인 H_e 게이트를 제안한다; 4) 마지막으로, Q-비트 초기 값의 설정과 새로운 종료 조건 및 H_e 게이트 등을 기반으로 TPQEA를 제안한다.

1. 종료 조건 (Termination criterion)

진화 알고리즘에서의 종료 조건은 상당히 중요한 요소 중의 하나이다. 진화 알고리즘이 우수하다고 하더라도 명확한 종료 조건이 주어지지 않는다면 좋은 결과를 얻기 힘들다. 기존 진화 알고리즘의 사용자들은 대부분 종료 조건으로 최대 세대수를 이용한다. 이것은 기존의 진화 알고리즘에 있어서 명확한 종료 조건을 찾는 일이 쉽지 않다는 것을 의미한다. 반면에 QEA의 경우에는 확률적 개체를 이용하기 때문에, II장의 수식 (6)과 같은 명확한 종료 조건의 설정이 가능하다. 즉, 최상위 최적 해 b 의 Q-비트 개체별 평균 확률값이 γ 보다 클 경우 알고리즘을 종료하게 된다.

표 I은 종료 조건 (6)의 γ 값 변화에 따른 영향을 살펴보기 위한 500개 아이টে를 갖는 주머니 문제(knapsack problem)의 실험 결과이다. 이때 주머니의 용량 제한은 평균 주머니 용량(average knapsack capacity) 제한을 사용하였다. γ 값을 0.001, 0.01, 0.1, 0.8, 0.9로 설정하여 실험을 수행하였다. γ 값이 0.01과 0.9일 경우의 결과를 보면, 종료된 평균 세대수는 거의 3배 가까이 차이가 나지만, 얻어진 최적 해는 거의 차이가 없음을 알 수 있다. 이와 같은 결과는 사용자의 입장에서 적당한 종료 조건을 부여하기 위한 γ 값의 설정에 어려움을 느낄 수 있다는 것을 의미한다. 이것은

표 II: 종료 조건 (8)의 γ 값 변화에 따른 영향 (500개의 아이টে를 갖는 주머니 문제에 대한 실험 결과): 실험에 사용된 다른 파라미터들의 설정은 표 I과 같다. $b.$, $m.$, $w.$ 는 각각 *best*, *mean*, *worst*를 의미하고, σ 와 t 는 각각 30번 실험 결과의 표준 편차와 알고리즘의 평균 종료 세대수이다.

γ of (8)		0.9	0.95	0.99
500	b.	2979.5	3016.2	3031.3
	m.	2949.5	2993.9	3020.1
	w.	2905.9	2960.7	3001.3
	σ	20.372	14.295	7.681
	t	484	722	1164

$Prob(\mathbf{b})$ 값이 각 Q-비트의 확률에 대해 너무 민감하기 때문이다.

각 Q-비트의 확률에 대한 민감도를 낮추어 사용자 입장에서 조건 γ 값을 명확하게 설정할 수 있는 새로운 종료 조건의 설계를 위해 다음과 같이 Q-비트 개체의 수렴도를 정의한다.

정의 1. 평균 비트 수렴도 C_b 는 Q-비트 개체의 수렴 정도를 나타내는 기준으로서 정의되며, 수식으로는 다음과 같이 정의된다:

$$C_b(\mathbf{q}) = \frac{1}{m} \sum_{i=1}^m |1 - 2|\alpha_i|^2|$$

또는

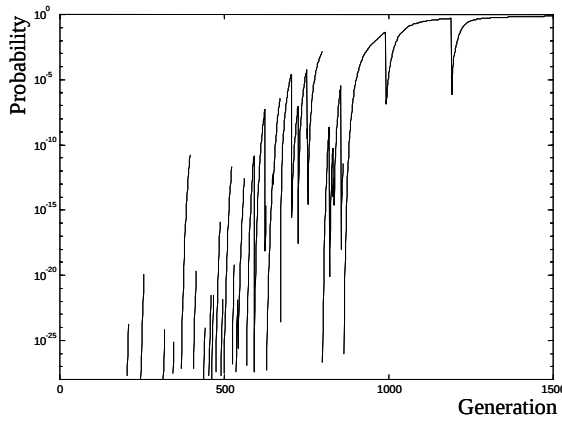
$$C_b(\mathbf{q}) = \frac{1}{m} \sum_{i=1}^m |1 - 2|\beta_i|^2|.$$

이때 $\mathbf{q}$ 는 Q-비트 개체이고, (α_i, β_i) 는 $\mathbf{q}$ 의 i 번째 Q-비트이다.

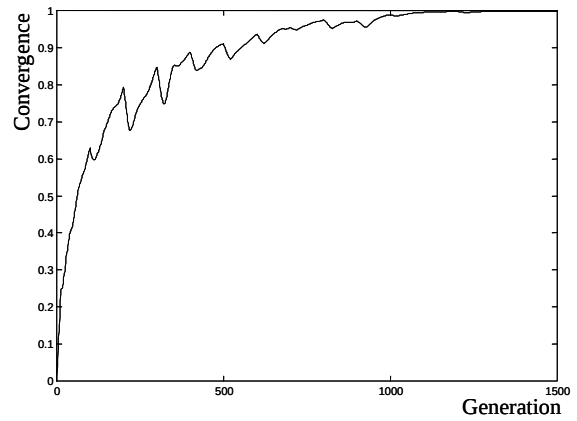
정의된 평균 비트 수렴도를 이용하여 다음과 같이 새로운 종료 조건을 설계할 수 있다:

$$\left(\frac{1}{n} \sum_{j=1}^n C_b(\mathbf{q}_j) \right) > \gamma. \quad (8)$$

이때 n 은 총 개체수이고, $C_b(\mathbf{q}_j)$ 는 j 번째 Q-비트 개체의 평균 비트 수렴도를 나타낸다. 새로운 종료 조건 (8)은 각 Q-비트가 0 또는 1의 상태로 어느 정도 수렴하였는지에 대한 명확한 의미를 제공한다. 예를 들어 평균 비트 수렴도 $C_b(\mathbf{q})$ 값이 0.99인 경우는 모든 Q-비트들이 0 또는 1의 상태로 평균 99% 수렴했다는 것을 의미한다. 즉, 사용자들은 평균 비트 수렴도의 명확한 의미를 바탕으로 보다 쉽게 종료 조건 γ 를 설정할 수 있게 된다. 표 II는 종료 조건 (8)의 γ 값 변화에 따른 영향을 살펴보기 위한 500개 아이টে를 갖는 주머니 문제(knapsack problem)의 실험 결과이다. 표 I과는 다르게 γ 의 변화에 따른 결과의 변화가 잘 나타나고 있음을 확인할 수 있다.



(a) 최적 해 $\mathbf{b}$ 의 확률: $\log_{10}(Prob(\mathbf{b}))$



(b) 평균 비트 수렴도: C_b

그림 6: (a) 기존의 종료 조건 (6)과 (b) 새로운 종료 조건 (8)의 비교. 세대(generation)에 따른 두 척도(measure)의 변화를 나타낸다. (a)의 경우에는 척도 값의 큰 변화 때문에 y 축은 $\log_{10}$ 스케일로 나타내었다.

두 종료 조건에 대한 차이는 그림 6에서 확연히 나타난다. 그림 6의 (a)와 (b)는 각각 기존의 종료 조건 (6)과 새로운 종료 조건 (8)의 척도(measure) 간의 차이점을 나타낸다. (a)는 $\log_{10}$ 스케일로 나타내었음에도 불구하고 세대의 변화에 따라 확률 값이 큰 폭으로 변하고 있음을 확인할 수 있다. 반면에, 새로운 종료 조건의 평균 비트 수렴도는 세대의 변화에 따른 Q-비트 개체의 수렴 여부를 잘 나타내고 있음을 확인할 수 있다.

2. Q-비트 초기 값의 영향

QEA의 초기화 과정인 ‘Q(t) 초기화’ 단계에서는 II장에서 언급한 바와 같이 모든 Q-비트 개체의 모든 Q-비트들을 $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ 로 초기화 시켰다[8]. 이것은 하나의 Q-비트 개체가 탐색 공간(search space) 내의 모든 가능한 해(solution)들을 각각 같은 확률로 포함하도록 초기화한 것이다. 즉, 초기에는 탐색 공간에 대한 정보가 전혀 없다고 가정하고, 탐색 공간의 모든 영역에 대해 무작위 탐색(random search)을 적용한 것이다. 하지만, 해를 찾고자 하는 탐색 공간에 대한 약간의 정보를 가지고 있다고 가정한다면, 이러한 사전 정보를 Q-비트의 초기 값에 반영할 수 있음을 쉽게 예측할 수 있다.

예를 들어, 500개 아이템을 갖는 주머니 문제를 생각해 보자. 이때 평균 주머니 용량(average knapsack capacity)이 아닌, 한정적 주머니 용량(restrictive knapsack capacity) $C = 2v$, $v = 10$,의 조

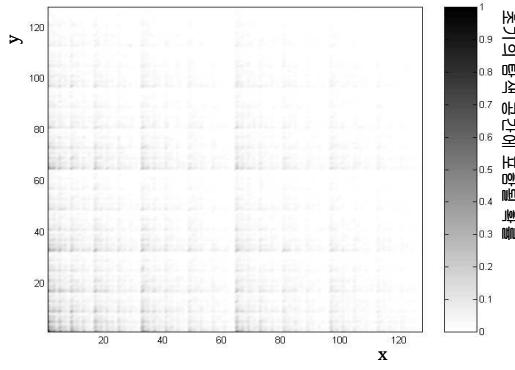
표 III: Q-비트의 초기 값 변화에 따른 영향 (한정적 주머니 용량, $C = 20$,을 갖는 500개 아이템의 주머니 문제에 대한 실험 결과): 실험에 사용된 QEA의 개체군 크기(population size)는 15, 전역 이주 주기 $T_g = 100$, 지역 그룹 크기는 3으로 설정하였고, 총 30번의 실험을 수행하였다. 종료 조건 (8)을 사용하였고, 이때 $\gamma = 0.99$ 로 설정하였다. *b.*, *m.*, *w.*는 각각 *best*, *mean*, *worst*를 의미하고, σ 와 t 는 각각 30번 실험 결과의 표준 편차와 알고리즘의 평균 종료 세대수이다.

(α_i^2, β_i^2)		(0.99, 0.01)	(0.5, 0.5)	(0.01, 0.99)
500	b.	94.993	94.856	94.892
	m.	86.973	81.978	80.969
	w.	79.942	69.983	69.996
	σ	3.774	5.846	5.662
	t	697	4682	8231

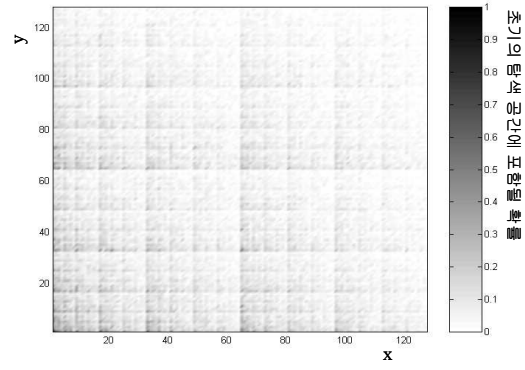
건을 갖는 주머니 문제를 고려한다. 아이টে을 담을 수 있는 주머니의 용량이 상당히 작기 때문에, 최적 해는 단지 수 개의 아이টে만을 포함할 것이라는 사실을 알 수 있다. 즉, 전체 탐색 공간 중에서 주머니의 용량 제한을 초과하는 탐색 공간이 거의 대부분일 것이다. 이때 우리는 이미 “최적 해는 단지 수 개의 아이টে만을 포함한다”는 사전 정보를 가지고 있는 것이고, 이 정보로부터 ($|\beta_i|^2$ 은 1의 상태로 관측될 확률이므로) Q-비트의 초기 값을 $\left(\sqrt{1 - \beta_i^2}, \beta_i\right) \Big|_{\beta_i \rightarrow 0}$ 와 같이 설정할 수 있을 것이다.

표 III은 한정적 주머니 용량($C = 20$)을 갖는 500개 아이টে의 주머니 문제에 대한 실험 결과를 보여준다. 표 III에서는 초기의 Q-비트 값이 QEA의 성능에 얼마나 큰 영향을 미치는가를 보여주고 있다. 초기 Q-비트 값 $(\alpha_i^2, \beta_i^2) = (0.99, 0.01)$ 인 경우가 다른 두 경우에 비해 더 좋은 결과를 보이고 있음에도 불구하고, 알고리즘의 평균 종료 시간은 가장 짧은 것을 확인할 수 있다. 수렴 속도로 환산하면, (0.99, 0.01)의 결과가 (0.5, 0.5)와 (0.01, 0.99)의 결과에 비해 각각 6.7배, 11.8배 더 빠르게 수렴하였다. 또한, 30번의 실험을 통해 얻은 최적 해들의 표준 편차도 (0.99, 0.01)의 경우가 가장 작은 결과를 보이고 있다. 이것은 알고리즘이 해를 찾아주는데 있어서의 강인성(robustness)이 더 우수하다는 것을 의미한다.

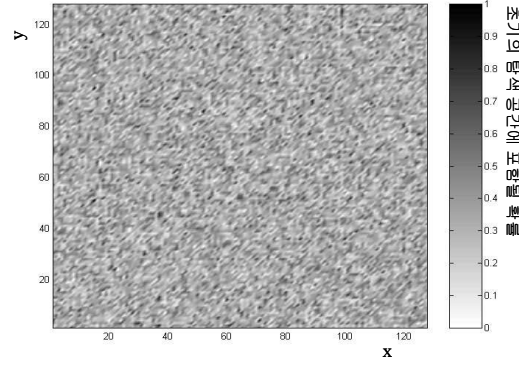
위의 실험 결과는 앞서 사전 정보를 통해 예측한 결과와 정확히 일치한다. 이와 같은 결과는 초기의 탐색 공간과 전역 최적 해와의 관계로 설명이 가능하다. 초기의 탐색 공간을 전역 최적 해와 가까운 영역으로 설정한다면 짧은 시간 내에 보다 더 우수한 해를 얻을 수 있을 것이다. 반면에, 초기의 탐색 공간을 전역 최적 해로부터 멀리 떨어진 영역으로 설정하였다면, 수렴 시간도 늘어날 뿐만 아니라 종료 시점에서 얻어진 해가 전역 최적 해와 유사할 확률도 낮아지게 된다.



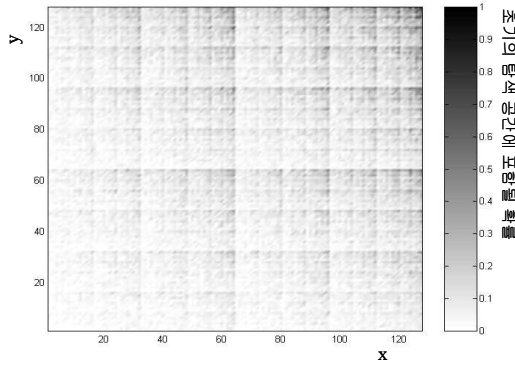
(a) $(\alpha_i^2, \beta_i^2) = (0.65, 0.35)$



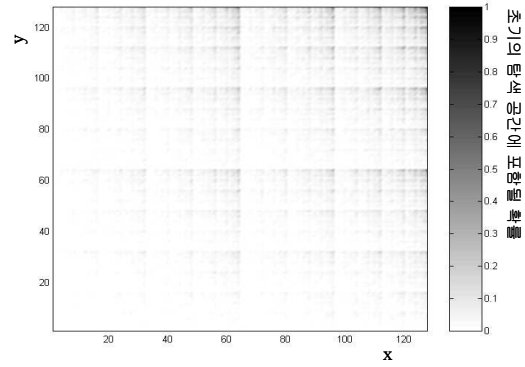
(b) $(\alpha_i^2, \beta_i^2) = (0.6, 0.4)$



(c) $(\alpha_i^2, \beta_i^2) = (0.5, 0.5)$



(d) $(\alpha_i^2, \beta_i^2) = (0.4, 0.6)$



(e) $(\alpha_i^2, \beta_i^2) = (0.35, 0.65)$

그림 7: 초기의 Q-비트 (α_i^2, β_i^2) 값의 설정에 따른 초기 탐색 공간(search space)의 차이: 전체 탐색 공간의 크기는 x 와 y 가 각각 2^7 으로 총 2^{14} 이다. (x, y) 평면 상에서 해당 위치의 밝기가 어두울수록 초기의 탐색 공간에 포함될 확률이 높다는 것을 의미한다. 초기 탐색 공간의 확률은 각 Q-비트 개체를 10^5 번 관측한 결과를 이용하여 계산하였다.

그림 7의 (a)-(e)는 초기의 Q-비트 (α_i^2, β_i^2) 값의 설정에 따른 초기 탐색 공간의 차이를 나타낸다. α 와 β 값을 동일하게 설정한 (c)의 경우에는, 모든 탐색 공간에 대해 거의 균일한 확률 분포를 보이고 있다. 이것은 초기 탐색이 전 영역에 대해 무작위 검색으로 시작한다는 것과 일치하는 결과이다. 하지만, α 와 β 값을 동일하게 설정하지 않은 (a), (b), (d), (e)의 결과에서는 의미있는 확률 분포를 보인다. α 가 β 보다 더 크게 설정될수록 (0000000, 0000000)의 방향으로 확률 분포가 쏠리게 되고, 반대로 α 가 β 보다 더 작게 설정될수록 (1111111, 1111111)의 방향으로 확률 분포가 쏠리게 된다는 사실을 확인할 수 있다.

3. H_ϵ 게이트

QEA의 변형 연산자는 Q-게이트로 정의되며, 기본 Q-게이트로는 회전 게이트를 사용한다. 하지만, Q-비트가 0 또는 1의 상태로 이미 완전히 수렴한 경우를 생각해 보자. 특정 상태로 완전히 수렴한 Q-비트는 다음 세대에서 아무리 관측을 하여도 수렴한 상태만을 결과로 보이게 된다. 즉, 완전히 수렴된 상태는 관측을 통해 빠져나올 여지가 전혀 없게 된다. 이것은 Q-비트 개체가 국부 최적 해에 빠졌을 경우에 쉽게 빠져나오지 못할 것임을 예측할 수 있다. 물론 전역 또는 지역 이주를 통해 수동적으로 탈출할 수 있는 여지가 있지만, 능동적인 탈출은 거의 불가능할 것이다. 이와 같은 문제점을 해결하기 위해서 다음과 같은 새로운 H_ϵ 게이트를 제안한다.

정의 2. H_ϵ 게이트는 회전 게이트의 변형된 형태를 갖는 Q-게이트로서, 다음과 같이 정의된다:

$$[\alpha'_i \beta'_i]^T = H_\epsilon(\alpha_i, \beta_i, \Delta\theta_i). \quad (9)$$

여기서, $[\alpha''_i \beta''_i]^T = R(\Delta\theta_i)[\alpha_i \beta_i]^T$ 에 대해:

i) 만약 $|\alpha''_i|^2 \leq \epsilon$ 이고 $|\beta''_i|^2 \geq 1 - \epsilon$ 이면,

$$[\alpha'_i \beta'_i]^T = [\sqrt{\epsilon} \sqrt{1 - \epsilon}]^T;$$

ii) 만약 $|\alpha''_i|^2 \geq 1 - \epsilon$ 이고 $|\beta''_i|^2 \leq \epsilon$ 이면,

$$[\alpha'_i \beta'_i]^T = [\sqrt{1 - \epsilon} \sqrt{\epsilon}]^T;$$

iii) 나머지 경우에는

$$[\alpha'_i \beta'_i]^T = [\alpha''_i \beta''_i]^T$$

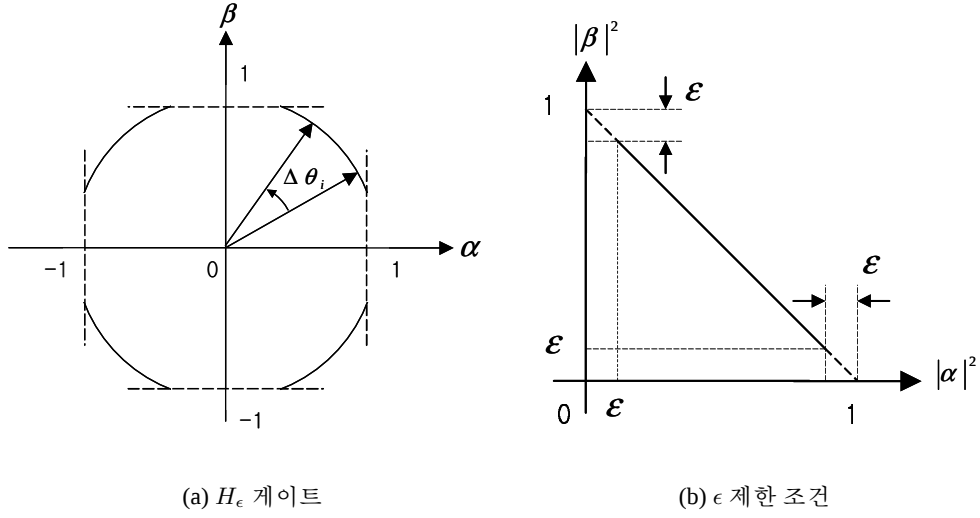


그림 8: H_ϵ 게이트: 회전 게이트의 개선된 형태인 새로운 Q-게이트.

로 정의된다. 이때, $0 < \epsilon \ll 1$ 을 만족하고, $R(\Delta\theta_i)$ 는 회전 게이트, $\Delta\theta_i, i = 1, 2, \dots, m$,는 각 Q-비트의 회전 각을 말한다.

그림 8은 H_ϵ 게이트를 나타낸다. 그림에서도 알 수 있듯이, $\lim_{\epsilon \rightarrow 0} H_\epsilon(\cdot)$ 은 회전 게이트와 동일하게 된다. 회전 게이트는 각 Q-비트의 확률 $|\alpha|^2$ 또는 $|\beta|^2$ 을 0 또는 1로 수렴시키지만, H_ϵ 게이트는 ϵ 또는 $(1 - \epsilon)$ 으로 수렴시킨다. H_ϵ 게이트는 0 또는 1의 상태로 완전히 수렴하는 것을 방지하기 때문에, Q-비트 개체가 국부 최적 해에 빠졌을 경우에도 관측 과정을 통해 능동적으로 탈출하는 것이 가능해진다. 즉, H_ϵ 게이트는 국부 최적 해가 많이 존재하는 문제일수록 우수한 성능을 보일 수 있다.

만약 H_ϵ 게이트를 Q-게이트로 사용한다면, 알고리즘의 종료 조건 (8)은 다음과 같이 ϵ 이 고려된 형태로 변경되어야 한다:

$$\left(\frac{1}{n} \sum_{j=1}^n C_b(\mathbf{q}_j) \right) > (1 - 2\epsilon)\gamma. \quad (10)$$

4. 2상 양자 진화 알고리즘 (TPQEA)

III.2절에서는 Q-비트 초기 값의 설정이 QEA의 성능을 향상시킬 수 있다는 사실을 확인하였다. 이것은 그림 7에서와 같이 Q-비트의 초기 값이 초기의 탐색 공간과 직접적으로 연결되어 있기 때문이었다. 만약 여기서, 전역 최적 해를 포함하는 탐색 공간과 연결된 초기 Q-비트 값을 찾아

이용할 수 있다고 한다면, Q-비트 개체는 짧은 시간 내에 효율적으로 전역 최적 해의 방향으로 수렴해 나갈 것이다. 이와 같은 개념을 QEA 알고리즘에 추가하여, 2상 양자 진화 알고리즘(TPQEA)을 제안하고자 한다.

TPQEA는 그림 9와 같이 크게 두 과정으로 정의된다. 첫 번째 과정은 ‘1단계 QEA’(first-phase QEA)로 우수한 성능을 보이는 초기 Q-비트 값을 찾는다. 두 번째 과정은 ‘2단계 QEA’(second-phase QEA)로서 1단계 QEA에서 찾은 초기 Q-비트 값을 이용하여 모든 Q-비트 개체들을 초기화하고 최적 해를 찾는다.

Procedure TPQEA

begin

- i) First-phase QEA
- ii) Second-phase QEA

end

그림 9: 2상 양자 진화 알고리즘(TPQEA).

Procedure first-phase QEA (1단계)

begin

- $t \leftarrow 0$
 - i) $Q(t)$ 지역 그룹별 초기화
 $Q(t)$ 상태 관측에 의한 $P(t)$ 생성
 $P(t)$ 평가
 $P(t)$ 중 최적 해를 $B(t)$ 에 저장
 - ii) **while** (not 종료 조건) **do**
begin
 $t \leftarrow t + 1$
 $Q(t - 1)$ 상태 관측에 의한 $P(t)$ 생성
 $P(t)$ 평가
 $Q(t)$ 갱신 (Q-게이트 연산자 이용)
 $B(t - 1)$ 과 $P(t)$ 중 개체별 최적 해를 $B(t)$ 에 저장
if (지역 이주 조건)
then 지역 그룹 내에서의 개체별 최적 해 간의 이주 (지역 이주)
end
 - iii) 가장 우수한 결과를 보인 지역 그룹의 초기 Q-비트 값을 $[\alpha^b \beta^b]^T$ 에 저장
- end**
-

그림 10: TPQEA의 1단계 QEA.

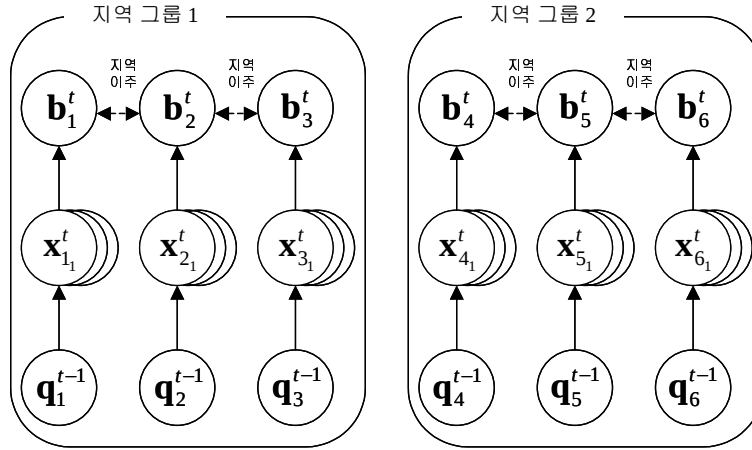


그림 11: 1단계 QEA 구조의 예: 개체군 크기(population size)는 6, 지역 그룹의 크기(local-group size)는 3, Q-비트 개체의 관측 회수는 3으로 설정하였을 경우의 구조이다.

그림 10은 TPQEA의 1단계 QEA를 나타낸다. 1단계 QEA는 기본 QEA와 거의 유사하지만, $Q(t)$ 의 초기화 과정이 다르다는 점에서 차이가 있다.

i) 초기화 과정에서는 지역 그룹별로 서로 다른 Q-비트 값을 설정하여, 각 지역 그룹이 서로 다른 탐색 공간을 탐색하도록 유도한다. g 번째 지역 그룹의 초기 Q-비트 값은 다음과 같이 설정한다:

$$\begin{bmatrix} \alpha_g \\ \beta_g \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{(1-2\delta)}{N_g-1}g + \delta} \\ \sqrt{1 - \frac{(1-2\delta)}{N_g-1}g + \delta} \end{bmatrix}. \quad (11)$$

여기서, N_g 는 지역 그룹의 개수이고, $0 < \delta \ll 1$ 의 조건을 만족하는 δ 는 0 또는 1의 상태에 대한 최소 확률을 나타낸다. 즉, 식 (11)은 $[\delta, 1 - \delta]$ 의 구간을 N_g 개수로 등분하여 그룹별 Q-비트의 확률로 할당한다.

1단계 QEA에서는 각 지역 그룹의 균질성(homogeneity)을 보장하기 위해 최상위 최적 해 $\mathbf{b}$ 와 전역 이주 과정을 사용하지 않는다. 좋은 Q-비트 초기 값을 찾기 위해서 지역 그룹별로 독립적인 최적화를 수행한다. 따라서, 지역 그룹간에는 정보 교환이 없으며, 지역 그룹 내에서만 지역 이주를 수행하게 된다. 예를 들어, 총 개체군 크기가 6이고, 지역 그룹의 크기가 3, Q-비트 개체의 관측 회수가 3일 경우의 1단계 QEA 구조는 그림 11과 같다.

ii) 1단계 QEA의 종료 조건으로는 H_e 게이트의 사용 여부에 따라 조건식 (8) 또는 (10)을 사용할 수 있다. 하지만, 1단계에서 2단계로의 빠른 전환이 필요한 문제에 대해서는 다음과 같은 종료

Procedure second-phase QEA (2단계)**begin** $t \leftarrow 0$

- i) $[\alpha^b \beta^b]^T$ 를 이용한 $Q(t)$ 초기화
 $Q(t)$ 상태 관측에 의한 $P(t)$ 생성
 $P(t)$ 평가
 $P(t)$ 중 최적 해를 $B(t)$ 에 저장

while (not 종료 조건) do**begin** $t \leftarrow t + 1$ $Q(t-1)$ 상태 관측에 의한 $P(t)$ 생성 $P(t)$ 평가 $Q(t)$ 갱신 (Q-게이트 연산자 이용) $B(t-1)$ 과 $P(t)$ 중 개체별 최적 해를 $B(t)$ 에 저장 $B(t)$ 중 최적 해를 $\mathbf{b}$ 에 저장**if** (전역 이주 조건)**then** 최상위 최적 해 $\mathbf{b}$ 를 모든 개체별 최적 해 집합 $B(t)$ 로 이주 (전역 이주)**else if** (지역 이주 조건)**then** 지역 그룹 내에서의 개체별 최적 해 간의 이주 (지역 이주)**end****end**

그림 12: TPQEA의 2단계 QEA.

조건식을 사용할 수 있다:

$$\left(\max_{j=1}^n C_b(\mathbf{q}_j) \right) > \gamma. \quad (12)$$

또는, H_ϵ 게이트를 사용할 경우에는 ϵ 의 영향을 고려하여

$$\left(\max_{j=1}^n C_b(\mathbf{q}_j) \right) > (1 - 2\epsilon)\gamma \quad (13)$$

의 종료 조건을 사용할 수 있다.

iii) 알고리즘의 마지막에는 종료 시점에서 가장 우수한 결과를 보인 지역 그룹의 초기 Q-비트 값을 $[\alpha^b \beta^b]^T$ 에 저장하는 과정이 추가된다.

그림 12는 TPQEA의 2단계 QEA 알고리즘을 나타낸다. 기본 QEA 알고리즘과 거의 동일하며, ' $t \leftarrow 0$ ' 과정이 제거되고, $Q(t)$ 의 초기화 과정이 변경된 점만이 다르다. 기본 QEA의 초기화 과정에서는 모든 Q-비트를 $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ 로 초기화하였지만, 2단계 QEA에서는 모든 Q-비트를 1단계 QEA에서 찾은 $[\alpha^b \beta^b]^T$ 값으로 초기화한다.

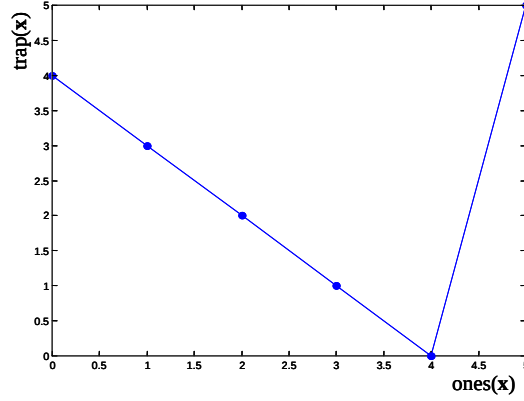


그림 13: 5-비트 트랩 함수.

표 IV: 연속된 5-비트 트랩 문제($N_{trap} = 20$)의 실험 결과: 실험에 사용된 개체군 크기(population size)는 15, 전역 이주 주기 $T_g = 100$, 지역 그룹 크기는 3으로 설정하였고, 총 30번의 실험을 수행하였다. 종료 조건 (10)을 사용하였고, 이때 $\gamma = 0.99$ 로 설정하였다. TPQEA의 1단계를 위한 γ 와 δ 는 각각 0.9와 0.05로 설정하였다. 또한, TPQEA와 QEA를 위한 H_ϵ 게이트의 ϵ 은 0.01로 설정하였다. $b.$, $m.$, $w.$ 는 각각 *best*, *mean*, *worst*를 의미하고, σ 와 t 는 각각 30번 실험 결과의 표준 편차와 알고리즘의 평균 종료 세대수이다. TPQEA의 t 는 두 단계의 세대수를 모두 합친 결과이다.

		QEA	TPQEA
f_{trap}	b.	90	100
	m.	85.033	100
	w.	81	100
	σ	1.722	0
	t	218	84

5. TPQEA의 성능 검증: 5-비트 트랩(5-bit trap) 문제에서의 적용

TPQEA의 성능을 검증하기 위해 다음과 같은 연속된 5-비트 트랩 문제를 고려한다:

$$f_{trap}(\mathbf{x}) = \sum_{i=0}^{N_{trap}-1} trap(x_{5i+1}, x_{5i+2}, x_{5i+3}, x_{5i+4}, x_{5i+5}).$$

여기서, N_{trap} 은 트랩의 개수이고, 하나의 5-비트 트랩 함수는 다음과 같다:

$$trap(\mathbf{x}) = \begin{cases} 4 - ones(\mathbf{x}) & , \text{ if } ones(\mathbf{x}) \leq 4 \\ 5 & , \text{ if } ones(\mathbf{x}) = 5. \end{cases}$$

함수 f_{trap} 의 최대 값을 갖는 해 (1, 1, 1, 1, 1)을 찾기 위해서는, 그림 13에서 보는 바와 같이 트랩 (0, 0, 0, 0, 0)에 빠져서는 안된다. 여러 개의 5-비트 트랩이 연속된 문제는 기존의 진화 알고리즘들이 대부분 전역 최적 해를 구하지 못하는 문제로 잘 알려져 있다.

표 IV는 연속된 5-비트 트랩 문제의 실험 결과를 보여준다. 5-비트 트랩의 개수 $N_{trap} = 20$ 으

로 설정하였고, 이때 전역 최적 해는 모든 100비트의 값이 1일때 $f_{trap} = 100$ 이다. 실험 결과로부터 TPQEA의 성능이 QEA에 비해 상당히 우수하다는 것을 확인할 수 있다. 30번 실험에 대해 QEA의 경우에는 평균 5개의 5-비트 트랩만을 빠져나온 반면, TPQEA의 경우에는 30번 실험 모두 20개의 5-비트 트랩을 모두 빠져나와 전역 최적해를 구하였다. 특히, TPQEA는 2단계로 구성되어 있음에도 불구하고, 평균 종료 세대수에 있어서도 QEA보다 훨씬 작은 값을 보이고 있음을 확인할 수 있다.

IV. 실험 및 결과

본 장에서는 TPQEA의 성능을 검증하기 위해 여러 최적화 문제에 대한 실험을 수행한다. 실험에서는 조합 최적화 문제와 수치 최적화 문제를 모두 다룬다.

1. 조합 최적화 (Combinatorial optimization)

조합 최적화 실험으로는 III장에서 언급한 한정적 주머니 용량($C = 2v, v = 10$)을 갖는 주머니 문제를 고려한다. 주머니 문제에 대한 자세한 설명은 부록 A에서 다룬다. 100개, 250개, 500개의 아이템을 갖는 세 가지 문제를 고려하였고, TPQEA의 성능 비교를 위해 QEA에 대한 실험을 함께 수행하였다. 주머니 문제에 대해서는 기존의 유전자 알고리즘(CGA: conventional GA)보다 QEA의 성능이 월등히 우수하다는 결과가 이미 검증되었으므로[8], 여기서는 QEA와의 비교만을 다루었다. II.3절의 결과에서도 100개 개체를 이용한 CGA의 결과보다 1개 개체를 사용한 QEA의 결과가 더 우수하였고, 이때 수렴 속도는 QEA가 약 70배 정도 빨랐다.

실험에서 사용한 TPQEA와 QEA의 파라미터 설정은 표 V와 같다. TPQEA와 QEA의 파라미터는 모두 동일하게 설정하였고, TPQEA의 1단계 QEA를 위한 파라미터 값만 추가되었다. TPQEA와 QEA의 Q-게이트 회전각 $\Delta\theta_i$ 의 값 $p (= |n|)$ 는 0.01π 로 설정하였다.

표 VI은 100, 250, 500개 아이템의 주머니 문제에 대한 TPQEA와 QEA의 실험 결과를 나타낸다. 최종적으로 찾은 최적 해의 이득 값에 있어서 세 가지 문제에 대해 모두 TPQEA가 우수한 결과를 보여주고 있음을 확인할 수 있다. TPQEA의 이득 값이 더 우수함에도 불구하고, 수렴 시간은 QEA보다 더 짧았다. 수렴 속도로 환산하면 TPQEA의 수렴 속도가 100, 250, 500 아이템 문제에 대해서 QEA의 수렴 속도보다 각각 1.8, 2.9, 4.0배 더 빠르다는 것을 알 수 있다. 또한, 30번의

표 V: 주머니 문제의 실험을 위한 TPQEA와 QEA의 파라미터 설정: ‘-’ 표시는 해당 파라미터는 사용되지 않음을 의미한다.

	QEA	TPQEA	
		1단계	2단계
총 개체수(population size)	15	15	
Q-비트 개체의 관측 회수	1	1	
지역 그룹 크기(local-group size)	3	3	
전역 이주 주기: T_g	100	-	100
종료 조건: γ	0.99	0.99	0.99
종료 조건식	(8)	(12)	(8)
(11)의 최소 확률: δ	-	0.01	-

표 VI: Q-비트의 초기 값 변화에 따른 영향 (한정적 주머니 용량, $C = 20$,을 갖는 주머니 문제에 대한 실험 결과): 실험에 사용된 파라미터는 표 V와 같고, 총 30번의 실험을 수행하였다. $b.$, $m.$, $w.$ 는 각각 *best*, *mean*, *worst*를 의미하고, σ 와 t 는 각각 30번 실험 결과의 표준 편차와 알고리즘의 평균 종료 세대수이다.

		QEA	TPQEA			QEA	TPQEA			QEA	TPQEA
100	b.	69.998	69.999	250	b.	94.998	94.997	500	b.	89.998	94.968
	m.	67.819	68.467		m.	87.484	90.122		m.	81.788	85.309
	w.	59.995	64.969		w.	74.991	84.674		w.	69.983	74.983
	σ	3.774	2.279		σ	4.604	3.551		σ	5.082	4.998
	t	1463	804		t	2680	929		t	4624	1165

실험을 통해 얻은 최적 해들의 표준 편차 결과에 있어서도 TPQEA가 더 우수한 결과를 보이고 있다. 이것은 TPQEA 알고리즘의 강인성(robustness)이 더 우수하다는 것을 의미한다.

그림 14는 TPQEA의 우수한 성능을 확연히 나타내고 있다. 실험으로 얻은 최적 해의 세대에 따른 평균 이득 그래프 (a), (c), (e)의 TPQEA 결과를 보면, 1단계에서 2단계로 전환되는 시점을 확인할 수 있고, 단계의 전환이 이루어진 이후에는 TPQEA와 QEA의 결과가 큰 차이로 벌어지는 것을 확인할 수 있다. 단, 100개 아이터를 갖는 문제에서는 비교적 작은 차이를 보이고 있다. 이것은 100개의 아이터를 갖는 문제는 비교적 간단하여, TPQEA와 QEA 모두 짧은 시간 내에 최적 해를 찾을 수 있었기 때문이다. 특히, 모든 개체들의 평균 이득 그래프 (b), (d), (f)는 TPQEA와 QEA의 성능의 차이를 확연히 보여준다. TPQEA의 경우에는 단계 전환이 발생한 이후 모든 개체들이 빠른 속도로 최적 해를 향해 수렴해가는 것을 볼 수 있는 반면, QEA의 경우에는 개체들의 수렴 속도가 느린 것을 확인할 수 있다. 500개 아이터의 문제 결과에서는 1,500 세대에서 TPQEA는 모든

개체들이 최적 해로 거의 수렴한 상태이지만, QEA의 개체들은 변화가 거의 없음을 볼 수 있다.

표 VI의 결과와 그림 14의 결과로부터 TPQEA가 QEA에 비해 최종적으로 얻어지는 최적 해의 수준, 수렴 속도, 동작의 강인성 등 모든 면에서 더 우수한 성능을 보인다는 사실을 확인할 수 있다.

2. 수치 최적화 (Numerical optimization)

QEA는 기본적으로 조합 최적화를 다루기 위한 알고리즘으로 제안되었고, 수치 최적화에 적용된 연구는 아직 발표되지 않았다. 일반적으로 수치 최적화에서는 실수 표현법을 이용하는 EP나 ES가 이진 표현법을 사용하는 GA보다 우수한 성능을 보인다고 알려져 있다. QEA도 이진 표현법을 사용하기 때문에 수치 최적화에 있어서 유리한 조건은 아니다. 그럼에도 불구하고, 본 절에서는 5가지의 수치 최적화 문제를 통해 QEA와 TPQEA의 수치 최적화 적용 가능성을 확인하고, 성능을 비교한다.

실험에서 사용된 5가지 수치 최적화 문제는 다음과 같다.

Sphere function:

$$f(\mathbf{x}) = \sum_{i=1}^N x_i^2 \quad (14)$$

탐색 공간은 $-100.0 \leq x_i \leq 100.0$, $N = 30$ 이다. $\mathbf{x} = (0, 0, \dots, 0)$ 에서 전역 최소값 0.0을 갖는다.

Ackley function:

$$f(\mathbf{x}) = -20 \exp \left(-0.2 \sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2} \right) - \exp \left(\frac{1}{N} \sum_{i=1}^N \cos(2\pi x_i) \right) + 20 + e \quad (15)$$

탐색 공간은 $-32.0 \leq x_i \leq 32.0$, $N = 30$ 이다. $\mathbf{x} = (0, 0, \dots, 0)$ 에서 전역 최소값 0.0을 갖는다.

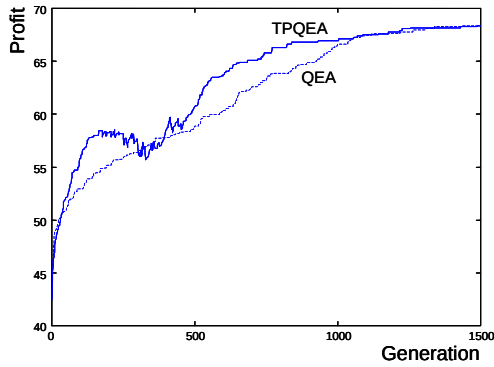
Rastrigin function:

$$f(\mathbf{x}) = 10N \sum_{i=1}^N (x_i^2 - 10 \cos(2\pi x_i)) \quad (16)$$

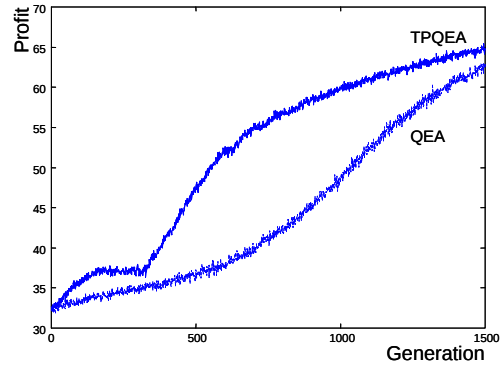
탐색 공간은 $-5.12 \leq x_i \leq 5.12$, $N = 30$ 이다. $\mathbf{x} = (0, 0, \dots, 0)$ 에서 전역 최소값 0.0을 갖는다.

Schwefel function:

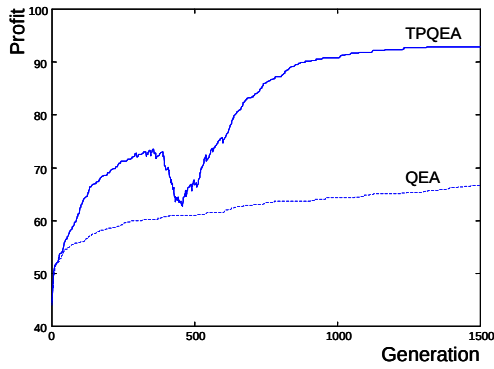
$$f(\mathbf{x}) = 418.9829N - \sum_{i=1}^N x_i \sin(\sqrt{|x_i|}) \quad (17)$$



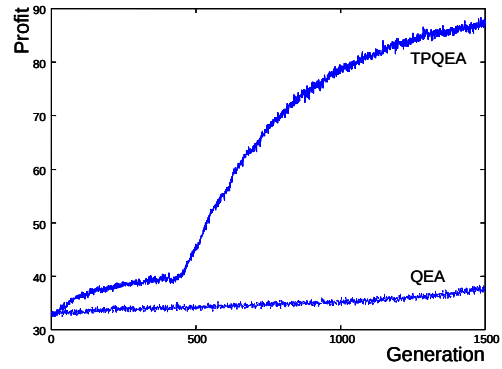
(a) 평균 최상 이득 (100 아이템)



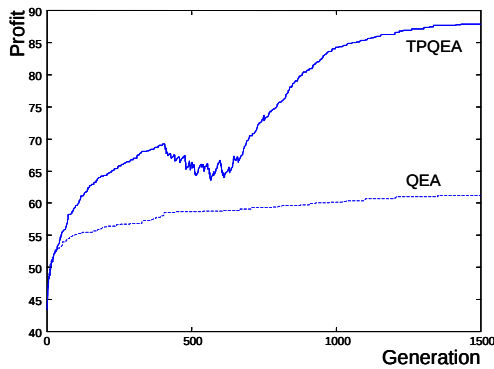
(b) 모든 개체들의 평균 이득 (100 아이템)



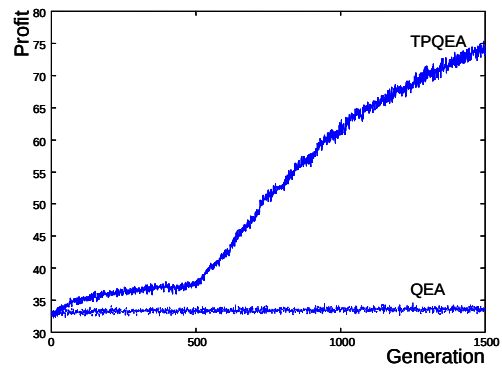
(c) 평균 최상 이득 (250 아이템)



(d) 모든 개체들의 평균 이득 (250 아이템)



(e) 평균 최상 이득 (500 아이템)



(f) 모든 개체들의 평균 이득 (500 아이템)

그림 14: 주머니 문제 실험 결과에 대한 TPQEA와 QEA의 성능 비교: 실험에 사용된 파라미터의 설정은 표 V와 같다. 단, 최종 종료 조건과는 무관하게 1,500 세대 까지의 결과를 도시하였다. 수직축은 주머니에 담긴 아이템들의 이득 값을 나타내고, 수평축은 세대수를 의미한다. 모든 실험은 30번 수행되었다. (a), (c), (e)는 30번 실험에 대한 평균 최상 이득(mean best profits)을 나타내고, (b), (d), (f)는 30번 실험에 대한 모든 개체들의 평균 이득(mean average profits)을 나타낸다.

표 VII: 수치 최적화 실험을 위한 TPQEA와 QEA의 파라미터 설정: ‘-’ 표시는 해당 파라미터는 사용되지 않음을 의미한다.

	QEA	TPQEA	
		1단계	2단계
총 개체수(population size)	15	15	
Q-비트 개체의 관측 회수	3	3	
지역 그룹 크기(local-group size)	3	3	
H_ϵ 게이트의 ϵ	0.01	0.01	
전역 이주 주기: T_g	100	-	100
종료 조건	MAX_GEN	$\gamma = 0.9$ of (10)	MAX_GEN
(11)의 최소 확률: δ	-	0.05	-

탐색 공간은 $-500.0 \leq x_i \leq 500.0$, $N = 30$ 이다. $\mathbf{x} = (-420.9687, -420.9687, \dots, -420.9687)$ 에서 전역 최소값 0.0을 갖는다.

Rosenbrock function:

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \quad (18)$$

탐색 공간은 $-30.0 \leq x_i \leq 30.0$, $N = 30$ 이다. $\mathbf{x} = (1, 1, \dots, 1)$ 에서 전역 최소값 0.0을 갖는다.

위의 5가지 함수는 모두 30개의 변수를 갖는 상당히 복잡한 함수들이다. 대부분의 함수들은 많은 국부 최소점을 갖고 있다. 각 함수에 대한 자세한 내용은 부록 B에서 기술한다.

실험에서 사용한 TPQEA와 QEA의 파라미터 설정은 표 VII과 같다. 문제 (14)-(18)에 대해 이진 코딩을 위한 스트링의 길이는 각 변수의 최소 해상도가 10^{-4} 이하가 되도록 변수당 22, 20, 17, 23, 20 비트로 각각 설정하였다. 수치 최적화에서 좋은 성능을 보이는 EP와의 비교를 위해, 동일한 문제를 다루고 있는 참고 논문[29]으로부터 두 가지 EP 방법의 실험 결과를 인용하였다. 첫번째는 기본 EP(SEP: standard EP)를 이용한 결과이고, 두번째는 fine-grained 병렬 진화 알고리즘(PEP)을 이용한 결과이다. 두 방법 모두 64개의 개체를 사용하였다. 특히, PEP는 실제 슈퍼컴퓨터 CrayT3E의 64개 프로세서를 이용하여 실험한 결과이다. TPQEA, QEA, SEP, PEP간의 공정한 비교를 위해, 인용한 논문에서 설정한 종료 조건 MAX_GEN을 그대로 이용하였다.

표 VIII은 TPQEA, QEA, PEP, SEP에 대한 5가지 수치 최적화 함수 (14)-(18)의 실험 결과이다. 비교적 단순한 f_{Sphere} 와 f_{Ackley} 함수의 경우, PEP의 결과가 약간 좋았지만, 4 가지 방법의 결과가 거의 유사하다. 하지만, 복잡한 문제인 $f_{Rastrigin}$, $f_{Schwefel}$, $f_{Rosenbrock}$ 함수의 경우, 개체수가 적

		EP		QEAs	
		SEP (64)	PEP (64)	QEA (15, 3)	TPQEA (15, 3)
f_{Sphere}	b.	0.000002	0.000000	0.000403	0.000453
	m.	0.003784	0.000510	0.001097	0.001351
	w.	0.075371	0.143013	0.002159	0.002807
	σ	-	-	0.000505	0.000705
	MAX_GEN	2000	2000	2000	2000
f_{Ackley}	b.	0.0090	0.0003	0.0142	0.0123
	m.	0.5366	0.0602	0.8226	0.4993
	w.	1.3404	2.0134	2.5325	2.3184
	σ	-	-	0.9210	0.7939
	MAX_GEN	1500	1500	1500	1500
$f_{Rastrigin}$	b.	19.90	16.92	17.22	9.08×10^{-6}
	m.	28.29	28.92	27.40	4.93
	w.	44.84	50.93	41.73	21.43
	σ	-	-	7.348	5.173
	MAX_GEN	5000	5000	5000	5000
$f_{Schwefel}$	b.	226.60	434.37	35.68	1.09
	m.	868.56	667.59	204.88	63.40
	w.	1304.00	1318.17	452.22	154.27
	σ	-	-	97.200	50.718
	MAX_GEN	9000	9000	9000	9000
$f_{Rosenbrock}$	b.	16.3	12.2	27.9	27.1
	m.	346.2	147.30	110.4	85.3
	w.	1393.7	9520.3	550.1	405.1
	σ	-	-	152.6	93.0
	MAX_GEN	20000	20000	20000	20000

표 VIII: 수치 최적화 함수 (14)-(18)의 실험 결과: EP의 결과는 [29]로부터 인용되었다. 특히, PEP는 실제 슈퍼컴퓨터 CrayT3E의 64개 프로세서를 이용한 실험을 통해 얻어진 결과이다. 각 실험은 20번씩 수행되었고, 괄호 안의 숫자는 사용된 총 개체군 크기(population size)를 의미한다. TPQEA와 QEA의 (15, 3)은 총 개체군 크기(population size) 15, 관측 회수 3회를 의미한다. $b.$, $m.$, $w.$ 는 각각 *best*, *mean*, *worst*를 의미하고, σ 와 MAX_GEN은 각각 20번 실험 결과의 표준 편차와 종료 세대수이다. ‘-’은 인용한 결과에서 주어지지 않은 값이다.

음에도 불구하고 TPQEA와 QEA의 결과가 SEP와 PEP의 결과보다 우수한 것을 확인할 수 있다. 특히, $f_{Schwefel}$ 과 $f_{Rosenbrock}$ 함수의 경우, ‘b.’와 ‘w.’의 결과는 TPQEA와 QEA가 PEP 및 SEP 보다 강인(robust)한 성능을 보인다는 사실을 보여준다.

TPQEA와 QEA의 결과만을 비교해 보면, 모든 면에 있어서 TPQEA가 QEA보다 우수한 성능을 보이고 있음을 확인할 수 있다. 특히, 표준 편차 결과는 TPQEA가 QEA보다 강인성이 우수함을 말해준다. 이것은 TPQEA의 1단계 QEA가 의미 있는 초기 Q-비트 값을 찾아 2단계 QEA로 제공할 수 있음을 의미한다.

수치 최적화 실험 결과는 TPQEA 및 QEA가 조합 최적화는 물론 수치 최적화 문제에도 적용이 가능하다는 사실을 입증해 주고 있다.

V. 결론

본 논문에서는 최근 우수한 성능을 보이고 있는 양자 진화 알고리즘(QEA)을 체계적으로 정리하고, QEA의 성능을 보다 향상시킬 수 있는 알고리즘 구조를 제안한다. 새로운 척도(measure)인 평균 비트 수렴도의 정의를 통해 명확한 종료 조건을 제시하고, QEA의 새로운 연산자인 H_{ϵ} 게이트를 정의하여 많은 국부 해를 갖는 문제에 대해서도 효율적인 탐색이 가능하도록 제안한다. 또한, 초기 Q-비트 값의 설정이 초기 탐색 공간과 밀접한 관계가 있음을 밝히고, 이를 이용하여 QEA의 성능을 향상시킬 수 있는 새로운 구조인 2상 양자 진화 알고리즘(TPQEA)을 제안한다. 연속된 5-비트 트랩 문제의 실험 결과는 TPQEA가 QEA에 비해 전역 탐색 능력이나 동작의 강인성, 수렴 속도면에 있어서 월등히 우수함을 보여주었다.

조합 최적화 문제인 주머니 문제의 실험 결과에서 TPQEA는 모든 면에서 QEA보다 우수한 성능을 보이고 있음을 확인하였다. 특히, 수렴 속도면에 있어서는 약 1.8배에서 최대 4.0배까지 빠르다는 사실을 확인하였다. 또한, 수치 최적화 문제에 대한 실험 결과를 통해 TPQEA 및 QEA가 조합 최적화 문제 뿐만 아니라 수치 최적화 문제에도 적용이 가능하며, 이진 스트링을 사용함에도 불구하고 우수한 성능을 보인다는 사실을 입증하였다. 특히, 슈퍼컴퓨터 CrayT3E의 64개 프로세서를 사용한 병렬 진화 알고리즘(PEP)의 실험 결과보다 TPQEA와 QEA가 더 우수한 결과를 보여 주었고, 모든 면에 있어서 TPQEA가 QEA보다 우수한 성능을 보여 주었다.

최근의 QEA에 대한 연구 결과[8]와 본 논문의 결과를 통해, 실제로 사용되고 있는 기존의 진화 알고리즘을 TPQEA로 대체하였을 경우, 수 배에서 수 십배까지 성능 향상을 가져올 수 있음을 알 수 있다. 특히, 수렴 속도면에서는 많은 성능 향상을 이룰 수 있을 것이다. 현재 진화 알고리즘이 많이 사용되고 있는 반도체 회로 설계의 최적화 문제, 각종 CAD 툴의 자동 부품 배치 및 자동 라우팅 기법 등을 포함하는 자동 최적화 문제, 각종 공작 기계의 작업 순서 최적화 문제 등 다양한 최적화 분야에 적용이 가능하다. 즉, 진화 알고리즘을 이용하여 설계된 최적화 툴의 코어는 TPQEA로 대체가 가능하며, 이때 현저한 성능 향상을 기대할 수 있다.

부록 A. 주머니 문제 (Knapsack problem)

주머니 문제는 조합 최적화(combinatorial optimization)의 대표적인 문제로 NP-hard 부류에 속한다[28]. 주머니 문제는 다음과 같이 기술할 수 있다: 주어진 많은 아이템 중에서 주머니의 무게 용량을 초과하지 않는 범위 내에서 가장 이득 값이 크도록 일부 아이템을 선택하는 문제이다. 이때, 아이템들은 각각 무게와 이득 값이 다르다.

수식으로 정리하면, m 개의 아이템과 하나의 주머니에 대해 이득 $f(\mathbf{x})$:

$$f(\mathbf{x}) = \sum_{i=1}^m p_i x_i$$

를 최대로 하는 아이템의 부분 집합을 선택하는 문제이다. 이때 선택된 아이템들의 무게의 총 합은 주머니의 용량 C 를 넘지 않아야 한다:

$$\sum_{i=1}^m w_i x_i \leq C.$$

여기서, $\mathbf{x} = (x_1 \cdots x_m)$ 이고, x_i 는 0 또는 1, p_i 는 i 번째 아이템의 이득, w_i 는 i 번째 아이템의 무게를 나타낸다. 만약 $x_i = 1$ 이면, i 번째 아이템이 선택된 것이다.

본 논문에서는 각 아이템들의 이득과 무게를 다음과 같이 강한 연관성(strongly correlated)을 갖도록 설정하였다:

$$\begin{aligned} w_i &= \text{uniformly random}[1, v] \\ p_i &= w_i + r. \end{aligned}$$

본문의 실험에서는 $v = 10, r = 5$ 로 설정하였다.

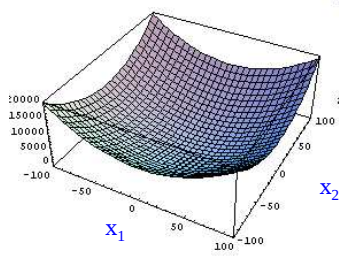
주머니의 무게 용량 C 는 다음과 같이 2가지의 형태를 고려할 수 있다[7].

i) 평균 주머니 용량(average knapsack capacity)은 모든 아이템들의 무게의 평균을 이용하여 다음과 같이 구한다:

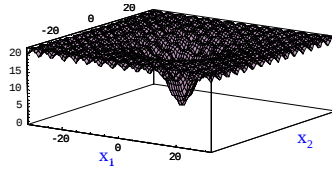
$$C = \frac{1}{2} \sum_{i=1}^m w_i.$$

ii) 한정적 주머니 용량(restrictive knapsack capacity)은 하나의 아이템의 최대 무게 v 를 이용하여 다음과 같이 구한다:

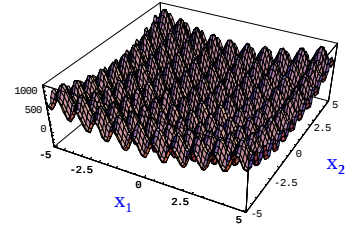
$$C = 2v.$$



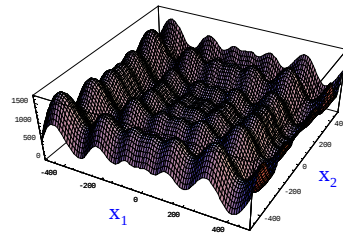
(a) Sphere 함수



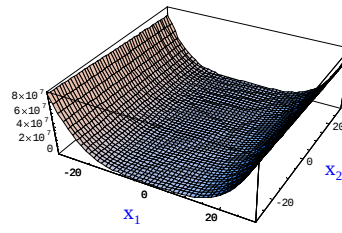
(b) Ackley 함수



(c) Rastrigin 함수



(d) Schwefel 함수



(e) Rosenbrock 함수

그림 15: 수치 최적화 함수 (14)-(18)의 개략적 모양: 실제 함수는 30개의 변수를 갖지만, 개략적인 모양을 관찰하기 위해 변수가 2개일 경우를 도시하였다.

부록 B. 수치 최적화 문제 (Numerical optimization problems)

Sphere 함수: 진화 연산의 수렴성을 평가하기 위해 고안되었다. 진화연산의 탐사(exploration) 특성보다 개척(exploitation) 특성의 평가에 비중을 둔 문제이다.

Ackley 함수: 함수의 두번째 지수 함수가 작은 봉우리와 골을 만들어서 많은 국부 최소점을 형성한다.

Rastrigin 함수: Ackley 함수보다 더 깊은 봉우리와 골짜기를 가지는 함수로, 국부 최소점을 빠져나오는 특성을 평가하기 위해 제안된 함수이다.

Schwefel 함수: 상당히 큰 봉우리와 골짜기를 가지고 있다. 또한, 전역 최소점과 가장 작은 국부 최소점 간의 거리가 상당히 멀리 떨어져 있어 쉽게 국부 최소점에 빠질 수 있다.

Rosenbrock 함수: $x_{i+1} = x_i^2$, $x_i = 1$ 인 점에서 전역 최소점이 나타난다. 하지만, 전역 최소점을 이끄는 $x_{i+1} = x_i^2$ 점이 또한 국부 최소점을 이끈다. 변수들이 곱으로 묶여있기 때문에 병렬 진화 연산을 사용해도 성능이 좋아지지 않는다고 알려져 있다.

참고 문헌

- [1] A. S. Fraser, "Simulation of Genetic Systems by Automatic Digital Computers," *Australian Journal of Biological Sciences*, vol. 10, pp. 484-491, 1957.
- [2] H. J. Bremermann, "Optimization through Evolution and Recombination," in *Self-Organizing Systems*, eds. M. C. Yovits, G. T. Jacobi, and G. D. Goldstine, Washington D.C.: Spartan Books, pp. 93-106, 1962.
- [3] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press, 1975.
- [4] L. J. Fogel, A. J. Owens and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*. New York: John Wiley, 1966.
- [5] I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart, Germany: Frommann-Holzboog, 1973.
- [6] H.-P. Schwefel, *Evolution and Optimum Seeking*. New York: Wiley InterScience, 1995.
- [7] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd, revised and extended ed. Berlin: Springer-Verlag, 1999.
- [8] K.-H. Han and J.-H. Kim, "Quantum-inspired Evolutionary Algorithm for a Class of Combinatorial Optimization," *IEEE Transactions on Evolutionary Computation*, Piscataway, NJ: IEEE Press, vol. 6, no. 6, pp. 580-593, Dec. 2002.
- [9] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines," *Journal of Statistical Physics*, vol. 22, pp. 563-591, 1980.
- [10] R. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6, pp. 467-488, 1982.

- [11] D. Deutsch, "Quantum Theory, the Church-Turing principle and the universal quantum computer," in *Proceedings of the Royal Society of London A*, vol. 400, pp. 97-117, 1985.
- [12] D. Deutsch, "Quantum computational networks," in *Proceedings of the Royal Society of London A*, vol. 425, pp. 73-90, 1989.
- [13] D. Deutsch and R. Jozsa, "Rapid solution of problems by quantum computation," in *Proceedings of the Royal Society of London A*, vol. 439, pp. 553-558, 1992.
- [14] D. R. Simon, "On the Power of Quantum Computation," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, Piscataway, NJ: IEEE Press, pp. 116-123, Nov. 1994.
- [15] P. W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, Piscataway, NJ: IEEE Press, pp. 124-134, Nov. 1994.
- [16] P. W. Shor, "Quantum Computing," *Documenta Mathematica*, vol. Extra Volume ICM, pp. 467-486, 1998.
- [17] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the 28th ACM Symposium on Theory of Computing*, pp. 212-219, 1996.
- [18] L. K. Grover, "Quantum Mechanics Helps in Searching for a Needle in a Haystack," *Physical Review Letters*, American Physical Society, vol. 79, no. 2, pp. 325-328, July 1997.
- [19] L. Spector, H. Barnum, H. J. Bernstein and N. Swamy, "Finding a Better-than-Classical Quantum AND/OR Algorithm using Genetic Programming," in *Proceedings of the 1999 Congress on Evolutionary Computation*, Piscataway, NJ: IEEE Press, vol. 3, pp. 2239-2246, July 1999.
- [20] B. I. P. Rubinstein, "Evolving Quantum Circuits using Genetic Programming," in *Proceedings of the 2001 Congress on Evolutionary Computation*, Piscataway, NJ: IEEE Press, vol. 1, pp. 144-151, May 2001.

- [21] M. Lukac and M. Perkowski, "Evolving Quantum Circuits using Genetic Algorithm," in *Proceedings of the 2002 NASA/DOD Conference on Evolvable Hardware*, Piscataway, NJ: IEEE Press, pp. 177-185, July 2002.
- [22] A. Narayanan and M. Moore, "Quantum-inspired genetic algorithms," in *Proceedings of IEEE International Conference on Evolutionary Computation*, Piscataway, NJ: IEEE Press, pp. 61-66, May 1996.
- [23] K.-H. Han and J.-H. Kim, "Genetic Quantum Algorithm and its Application to Combinatorial Optimization Problem," in *Proceedings of the 2000 Congress on Evolutionary Computation*, Piscataway, NJ: IEEE Press, vol. 2, pp. 1354-1360, July 2000.
- [24] K.-H. Han, K.-H. Park, C.-H. Lee and J.-H. Kim, "Parallel Quantum-inspired Genetic Algorithm for Combinatorial Optimization Problem," in *Proceedings of the 2001 Congress on Evolutionary Computation*, Piscataway, NJ: IEEE Press, vol. 2, pp. 1422-1429, May 2001.
- [25] M. Moore and A. Narayanan, *Quantum-inspired Computing*, Technical report, Department of Computer Science, University of Exeter, UK, 1995.
- [26] K.-H. Kim, J.-Y. Hwang, K.-H. Han, J.-H. Kim, and K.-H. Park, "A Quantum-inspired Evolutionary Computing Algorithm for Disk Allocation Method," to appear in *IEICE Transactions on Information and Systems*, IEICE, Japan, 2003.
- [27] R. Hinterding, "Representation, Constraint Satisfaction and the Knapsack Problem," in *Proceedings of the 1999 Congress on Evolutionary Computation*, Piscataway, NJ: IEEE Press, vol. 2, pp. 1286-1292, July 1999.
- [28] M. R. Garey and D. S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness*, San Francisco: W. H. Freeman & Company, 1979.
- [29] S.-H. Park, *Efficient Migration Methods and Topology in Fine-Grained Parallel Evolutionary Algorithm*, Master thesis, Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, Korea, 1999.