DOI: 10.3745/KIPSTD.2010.17D.3.175

DNA Chip 데이터의 군집화 성능 향상을 위한 Particle Swarm Optimization 알고리즘의 적용기법

이 민 수†

요 약

최근 DNA 칩의 등장으로 유전자 관련 실험과 연구가 매우 용이해졌으며 이를 활용한 다양한 실험 결과로 대량의 데이터가 제공되고 있다. DNA칩에 의해 제공된 데이터는 2차원 행렬로 표현되며 하나의 축은 유전자를 나타내고 다른 하나의 축은 샘플정보를 나타낸다. 이러한 데이터에 대하여 빠른 시간 안에 좋은 품질의 군집화를 수행함으로써 이후의 분석 단계인 분류화 작업의 정확도와 효율성을 높일 수 있다. 본 논문에서는 생태계 모방 알고리즘의 하나인 Particle Swarm Optimization 알고리즘을 사용하여 방대한 양의 DNA칩 데이터에 대한 효율적인 군집화 기법을 제안하였으며 실험을 통해서 PSO 기반의 군집화 알고리즘이 기존의 군집화 알고리즘들보다 수행속도 및 품질 면에서 우수한 성능을 가짐을 보였다.

키워드: Particle Swarm Optimization 알고리즘, 군집화, DNA칩 분석

Applying Particle Swarm Optimization for Enhanced Clustering of DNA Chip Data

Minsoo Lee†

ABSTRACT

Experiments and research on genes have become very convenient by using DNA chips, which provide large amounts of data from various experiments. The data provided by the DNA chips could be represented as a two dimensional matrix, in which one axis represents genes and the other represents samples. By performing an efficient and good quality clustering on such data, the classification work which follows could be more efficient and accurate. In this paper, we use a bio-inspired algorithm called the Particle Swarm Optimization algorithm to propose an efficient clustering mechanism for large amounts of DNA chip data, and show through experimental results that the clustering technique using the PSO algorithm provides a faster yet good quality result compared with other existing clustering solutions.

Keywords: Particle Swarm Optimization Algorithm, Clustering, DNA Chip Analysis

1. 서 론

다양한 생물 정보에 대한 편리하면서도 빠르고 정확한 분석 기술이 중요해짐에 따라 실험 정보를 담은 바이오 칩(Biochip)에 대한 통합 분석의 필요성이 크게 대두되고 있다. 바이오칩은 분자 생물학적 지식에 기계 및 전자공학의 기술을 접목해서 만들어졌다. DNA칩과 기존의 유전공학 방법들과의 가장 큰 차이점은 아주 적은 양의 유전물질을 고밀도로 붙이는 것이 가능하고 동시에 최소한 수 백 개 이상의 유전자를 빠른 시간 안에 다룰 수 있고, 분석이 가능하기 때문에 비용과 시간을 절약할 수 있다는 것이다. 바이오칩 기술을 이용하면 생물의 생명현상에 대한 근본적인 원리와 구조를 분석할 수 있고 새로운 신약을 개발하거나 인간의 뇌의 정보처리 메커니즘을 분석할 수 있으며 질병을 진단하고 예측할 수 있다.

바이오칩을 사용하여 유전자 실험을 한 후에는 바이오칩으로부터 데이터를 추출하고 유전자 사이에 유사성 유형을 발견하기 위해 데이터를 조사하는 분석 과정이 필요하다. 유전자 데이터를 사용해 질병의 발전 방향을 예측하는 등[1, 2] 특히 의생명 분야에서 그 분석이 중요시되는데, 데이터의 분석을 위해 사용하는 방법으로 군집화나 분류화 등의 방법

※ 본 논문은 정부(교육과학기술부)의 재원으로 한국연구재단 지원을 (No. 2009-0083992) 받아 수행된 연구임.
† 종신회원 : 이화여자대학교 컴퓨터공학과 부교수
논문접수 : 2009년 7월 9일
수 정 일 : 1차 2010년 1월 10일
심사완료 : 2010년 4월 13일

이 있지만 본 논문에서는 데이터 마이닝 기법의 하나인 군집화를 이용하여 의미 있는 정보를 찾아내고 분석한다. 그리고 본 논문에서는 생태계 모방 알고리즘의 하나인 PSO(Particle Swarm Optimization) 알고리즘을 사용하여 군집화를 수행하는 PSO기반의 군집화 알고리즘을 제안한다. PSO 알고리즘은 개체들 간의 상호 작용을 통해 최적의 해를 찾아가는 방법을 모방하는 알고리즘으로서 단순하고 계산시간이 짧으며 대량의 메모리가 필요 없다는 장점이 있다. 따라서 PSO 알고리즘을 이용하여 군집화를 수행하고 수천 개부터 많게는 수십만 개의 기록을 가진 대량의 바이오칩 데이터를 처리할 때, 높은 성능을 얻을 수 있다. 본 논문에서는 PSO 알고리즘을 이용하여 바이오 칩으로부터 간암환자와 정상인의 데이터를 받아 성질이 비슷한 유전자끼리 군집화를 함으로써 특징추출을 통한 차원 축소 및 중복성 제거로 이후의 분류화 과정에서 효율적이고 정확하게 간암 환자인지 아닌지를 판별할 수 있도록 군집화를 한다. 이때, 군집화 과정은 빠른 수행시간을 가져야 하며 간단하게 판별이 되어야 한다. PSO 군집화 알고리즘의 성능이 기존의 K-means 알고리즘과 비교하였을 때 수행속도 면에서 약 10% 정도 향상되며 KNN 군집화 알고리즘과 비교하였을 때, 데이터의 수가 많을수록 수행속도 면에서 월등히 향상 되었다는 것을 실험을 통하여 보였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 DNA칩 분석과정, 데이터마이닝 기법인 군집화에 대한 설명을 하고, 생태계 모방 알고리즘의 하나인 PSO 알고리즘에 대해 설명한다. 3장에서는 PSO 알고리즘 기반의 데이터 분석 방법인 PSO 군집화 알고리즘에 대해 제안하고 4장에서는 시스템 개발 환경과 구현에 대해서 설명하며 5장에서는 실험 및 성능을 평가한다. 마지막으로 6장에서는 결론 및 향후 연구 방향을 기술하고 있다.

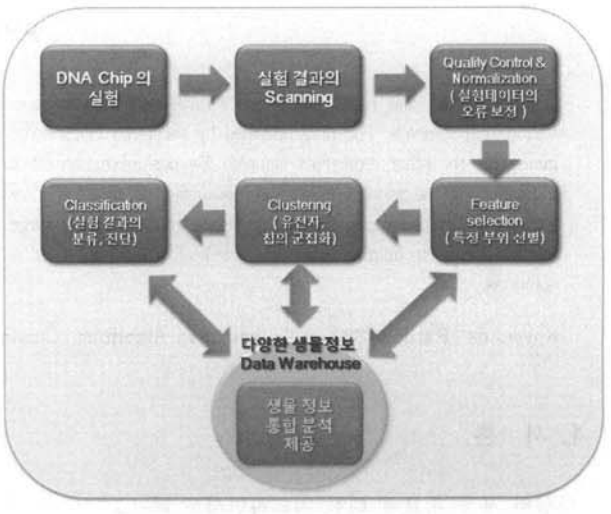
2. 관련 연구

2.1 바이오칩 분석시스템

바이오 칩(Biochip)이란 DNA나 단백질과 같은 생체물질을 유리, 실리콘, 나일론 등의 재질로 된 고체기질 위에 고밀도로 집적화한 것으로 유전자 발현 양상, 유전자 결함, 단백질 분포 등의 생물학적 정보를 얻는데 쓰이며 생화학적 공정 및 반응속도 또는 정보처리 속도를 높이는 도구나 장치를 말한다[3, 4, 5]. 바이오 칩 중의 하나인 마이크로어레이(microarray)는 수천 혹은 수만 개의 DNA, 단백질, 탄수화물, 펩타이드(peptide) 등을 일정간격으로 배열하여 붙이고, 분석대상 물질을 처리하여 결합 양상을 분석할 수 있는 칩(DNA칩, 단백질 칩)이다[6].

DNA 칩은 실리콘, 표면개질 유리, 폴리프로필렌, 활성화 폴리아크릴 아마이드와 같은 고체 표면에 염기서열을 알고 있는 8~25 염기 크기의 탐침 올리고뉴클레오티드를 적게는 1000개 많게는 1,000,000개 까지 정해진 위치에 부착시켜 놓는다. 이러한 DNA 칩에 분석하고자 하는 표적 DNA 절편을 결합시키면 DNA 칩에 부착되어 있는 탐침들과 표적 DNA 절편의 염기서열의 상보성에 따라 서로 다른 혼성화 양상을 나타내는데, 이를 광학적인 방법이나 방사능 화학적 방법 등을 통해 관찰 해석함으로써 표적 DNA의 염기 서열 또는 유전자형 분석을 하거나 시료 중의 특정 유전자 발현 양상을 분석할 수 있다[3].

바이오칩 분석 시스템에서의 분석과정은 (그림 1)에 보인 것처럼 DNA칩 실험에서 화상처리(image-processing)를 통해 이상 유무를 확인한 뒤 얻어진 수치화된 데이터(data)를 생성한 뒤 여러 오류를 보정해 주기 위해 품질관리 (quality control)및 정규화(normalization)분석을 거치게 되고, 이 데이터를 다시 특징 선택(feature selection)을 통해 특징을 추출한다. 마이크로어레이 데이터의 특징 선택을 위한 LSSVM (Least squares support vector machine)과 이를 위한 최적화 방법으로 PSO를 이용하는 연구가 진행되었으며[7], PSO를 사용하여 특징 선택을 한 후 SVM(Support vector machine)로 분류하는 방법이 연구되었다[8, 9]. 이와 같이, 마이크로어레이 데이터의 차원 축소와 중복성을 제거하기 위한 특징 선택과정으로 PSO(Particle Swarm Optimization)을 적용하기도 한다. 이 후 서로 연관성 있는 유전자들을 그룹화 해주는 군집화(clustering) 및 분류화(classification) 과정을 거친다[10-12]. 그리고 관련 유전자의 기능별 분류, 생체경로 정보 분석 등에 대한 결과 내용을 통합 데이터베이스 시스템을 통해 제공한다[13].



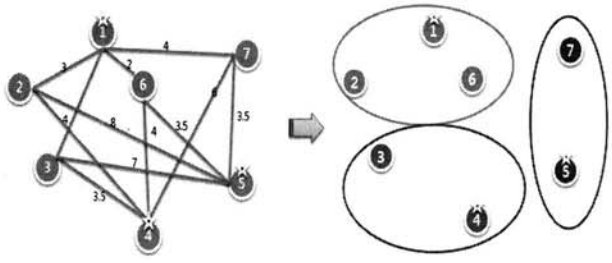
(그림 1) 바이오칩 분석 시스템

2.2 군집화 (Clustering)

군집화(clustering)는 훈련(training)과정 없이 유사도에 근거하여 군집들을 구분하는 데이터마이닝 기법으로서 (그림 2)와 같이 군집 내의 유사성을 극대화 하고 군집간의 유사성을 최소화 하여 데이터의 객체를 분석한다. 이 때, 군집간의 유사도를 평가하기 위해 사용되는 거리 측정 함수로는 Euclidean distance, Mahalanobis distance 등이 있다[11, 12].

다양한 군집화 기법 중 계층적 군집화는 트리구조로 내포

된 군집들의 집합으로 트리의 각 노드는 자식들의 합집합이며 트리의 루트는 모든 객체를 포함하는 군집이다. 부모, 자식 관계처럼 군집이 다른 하위 군집을 가질 때 계층적 군집화가 되며 가끔 트리의 말단 노드는 한 개의 객체만을 갖기도 하는데 이것은 단일 원소 군집이라고 한다. 이처럼 계층적 군집화는 상위의 군집이 하위의 군집을 포함하는 형태로 군집의 소속이 중복적으로 일어나는데 반하여 분할적 군집화는 데이터 객체들을 평면적으로 중복이 없는 부분집합으로 나누는 것이다. 그리고, 미리 몇 개의 군집으로 나누어질 것이라고 예상하고 개수를 정한다. 대표적인 예로 K-means 군집화 알고리즘이 있으며, K개의 군집을 만들고 하나의 객체가 하나의 그룹에 속해 있는 경우에 사용된다. 이외에도 하나의 객체를 하나의 군집에 지정하는 배타 군집화, 데이터 집합 내의 잡음이나 이상치 같은 객체들을 고려해 따로 군집화 해주는 부분 군집화 기법 등이 있다.



(그림 2) 군집화 과정

2.3 PSO(Particle Swarm Optimization) 알고리즘

생태계 모방 알고리즘은 생태계에서 일어나는 생물체들의 행동이나 습성을 관찰하여 알고리즘에 적용시킨 것으로 다양한 진화적 알고리즘 중 한 부분을 차지한다. 기본적으로 어느 주어진 생태계 안에서 각각 하나씩의 가능한 해(solution)를 갖는 개체들이 모인 개체군이 각 알고리즘 나름의 진화 연산을 수행하면서 최적의 해 집단을 형성해 가는 것을 주요 목적으로 한다[11]. 이러한 생태계 모방 알고리즘이 갖는 주요 특성으로는 연산과정에 있어서 확률론적인 접근방법을 사용한다는 것과 각 개체들 간, 즉 해 간의 상호 작용을 통해 최적의 해를 찾아가는 방법을 사용한다는 것이다.

PSO(Particle Swarm Optimization) 알고리즘은 잘 알려진 생태계 모방 알고리즘 중 하나로써, 새, 물고기, 벌 등 군집 생활을 하는 동물들의 행동 습성을 모방하여 최적의 해를 찾는 알고리즘이다. 이 방법은 여러 개의 입자(particle)들이 탐색공간 안에서 흩어져 있어, 반복을 거듭하면서 좀 더 나은 해에 가까운 위치로 자신들의 위치를 변화시켜 가며 점점 입자들의 집단이 최적의 해를 찾아가는 방향으로 수렴하게 된다.

일반적인 PSO 알고리즘의 연산과정은 다음과 같다. D-차원(D-Dimension)의 탐색공간에서 N개의 입자들이 매번 반복을 거듭하여 움직인다고 할 때, t번째 반복에서의 i 번째 $(1 \le i \le N)$ 입자의 위치를 다음과 같은 벡터형으로 표현하며, 이 위치에 의해 입자의 우수성이 평가된다[14].

$$X_i(t) = (x_{i1}, x_{i2}, \cdots, x_{iD})$$

반복을 거치며 각 입자들은 위치를 변화시키게 되는데, 두 개의 참고점을 고려하게 된다. 하나는 각각의 입자들이 현재까지 자신들의 위치를 변화하는 동안 가장 우수성이 좋았을 때의 위치 정보, 즉 위치 벡터이고 이것을 지역적 최고지점 (Local Best Position, *lbest*) 또는 개인적 최고지점 (Personal Best Position, *pbest*) 이라고 한다. 그리고 다른 하나는 전체 입자들의 현재까지 위치 변화를 통틀어서 가장 우수성이 좋았을 때의 위치 정보이고 이것을 전역적 최고지점 (Global Best Position, *gbest*) 이라고 한다. 따라서 *pbest*는 총 N개, *gbest*는 1개가 존재하게 된다. i번째 입자의 *pbest* 정보 P_i와 전체 입자 안의 *gbest* 정보 P_g는 다음과 같이 표현한다.

$$P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$$

$$P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$$

그리고 매 t 번째의 반복 때마다 입자의 위치 벡터를 변화시켜주는 요소인 속도 벡터 $V_i(t)$는 다음과 같이 표현된다.

$$V_i(t) = (v_{i1}, v_{i2}, \cdots, v_{iD})$$

이상의 변수들을 사용하여 매 t번의 반복마다 i번째 입자 $X_i(t)$의 위치 벡터는 다음의 식에 의해 업데이트되며 이때 c_1, c_2는 양의 정수 값이다[13].

$$V_i(t) = V_i(t-1) + c_1(P_i - X_i(t-1)) + c_2(P_g - X_i(t-1))$$
$$X_i(t) = X_i(t-1) + V_i(t)$$

매 반복마다 위와 같은 과정을 거치며 각 입자의 위치를 변화시키게 되고, 위치 변화 후에는 적합도 함수(Fitness function)를 이용하여 각 입자의 우수성을 평가하여 새로운 *pbest*와 *gbest*가 생기면 정보를 업데이트 하게 된다. 최종적으로 모든 과정이 끝났을 때의 저장된 *gbest* 위치 벡터가 가장 최적의 해에 가까운 해로 도출 된다.

PSO 알고리즘은 처음에 각 입자의 위치와 속도를 초기화 시킨다. 그리고 모든 입자의 적합도 값을 계산하고 계산된 적합도 값과 *pbest*를 비교하여 새로운 *pbest*를 구한다. 그리고 *pbest*중에서 가장 좋은 값을 *gbest*로 선정하게 된다. *pbest*와 *gbest*를 구한 뒤 입자의 속도를 계산하고 현재 입자의 위치와 계산된 속도 값을 이용하여 입자가 다음에 이동할 위치를 결정한다. (그림 3)은 PSO 알고리즘의 pseudo-code를 보여준다.

PSO 알고리즘은 다음과 같은 특징을 갖는다. 첫째, PSO 알고리즘은 복수의 탐색점을 가지며 각 탐색점의 *pbest*(각 개체별로 알고 있는 최적의 해)와 *gbest*(무리가 알고 있는 최적의 해)를 이용하여 각 탐색점을 확률적으로 변경시켜가는 것에 의해 global한 최적 해를 발견한다. 둘째, 연속형의

변수와 이산형의 변수가 혼합되어 있는 경우에도 전체적인 집단화가 가능하다. 셋째, PSO의 개념은 원래 2차원 공간에서 고안되었으나 n차원 공간으로 확장할 수 있다.

```
PSO Algorithm
for each particle do  // 입자의 초기화
    initialize position and velocity of particle
endfor
for each particle do
    calculate fitness value // 적합도 계산
    if current fitness value is better than local best fitness value then
        take current fitness as new local best fitness // pbest구함.
    endif
endfor
choose as the particle with best fitness value among all particles in current iteration // gbest 구함.
for each particle do
    calculate velocity of particle // 속도 계산
    update position of particle // 위치 업데이트
endfor
```

(그림 3) PSO 알고리즘

3. PSO 기반의 군집화 알고리즘

PSO 알고리즘을 기반으로 한 PSO 군집화 알고리즘은 수행과정은 크게 세단계로 볼 수 있다. 첫 번째 단계는 품질관리와 정규화 과정을 거친 바이오 칩 데이터를 입력데이터로 받고 두 번째 단계는 PSO 군집화 알고리즘을 사용하는 단계로서 이 단계에서는 각각의 유전자에 대해서 입자(particle)들이 초기화 되고 입자와 속도의 매핑이 이루어진다. 이 때 임의로 각 유전자의 군집을 정해준다. 입자는 모든 유전자에 대해 군집 결과를 인코딩하고 N개의 유전자 데이터를 가진 N차원 벡터가 된다. 임의로 생성된 군집의 중심을 결정하기 위해 군집에 속한 모든 유전자의 발현 값을 더하고 발현 값의 합을 유전자의 수로 나눈다. 이렇게 군집의 중심이 결정되면 오차제곱의 합(Sum of Squared Error, SSE)를 사용하여 모든 입자의 적합도를 계산하고, 적합도를 계산한 뒤에는 *pbest*와 계산된 적합도를 비교하여 *pbest*를 업데이트한 뒤 *pbest*중에서 가장 좋은 값을 *gbest*로 선정한다. 그리고 속도를 계산한 뒤 다음 이동할 입자의 위치를 결정하게 된다. 이러한 과정을 반복 횟수만큼 수행을 하게 된다. 마지막 세 번째 단계는 결과를 도출하는 단계로써 입력 받은 데이터의 군집화 결과를 도출한다. (그림 4)는 PSO 군집화 알고리즘을 보여준다.

실제 예제를 들어 PSO 군집화 알고리즘의 입자의 속도와 위치 매핑 및 적합도 함수는 다음과 같다.

```
PSO Clustering Algorithm
  Step 1
  input(bio data): Xi(t)=(xi1, xi2, ···, xid)
        /* 실험과 유전자 값으로 구성된 데이터 파일 */
  number of clusters k    /* 군집의 개수를 정해줌 */
  number of iterations I  /* 알고리즘의 반복횟수를 정해줌 */

  Step 2
  Initialize the position and velocity of particles /* 입자의 위치와 속도 초기화 */
  for number of iterations do
      for each data vector do
          Calculate the centroid of clusters /* 군집의 중심을 구함 */
      endfor
      for each cluster do /* 군집의 적합도 계산 */
          Calculate the fitness using SSE :
      (∑_{j=1}^{Nc} [∑_{∀Zp∈Cij} d(Zp,mj)/|Cij|]) / Nc
      endfor
      for each particle do /* SSE에 기반하여 입자의 지역 최적값을 구함 */
          Find the local best positions
      endfor
      for each particle do
              /* 지역 최적값중 가장 좋은 값을 전역 최적값으로 선정 */
          Find the global best position
              /* 입자의 속도 계산 */
          Calculate the velocity :
      Vi(t) = Vi(t-1) + c1(Pi - Xi(t-1)) + c2(Pg - Xi(t-1))
              /* 입자의 위치 업데이트 */
          Update the particle : Xi(t) = Xi(t-1) + Vi(t)
      endfor
  endfor

  Step 3
  output: clustering result for each gene corresponding to gbest
          /* 가장 좋은 적합도 값을 갖는 군집 결과 출력 */
```

(그림 4) PSO 군집화 알고리즘

3.1 위치 매핑

PSO 군집화 알고리즘에서 각 입자(particle)는 최종 답안의 후보 해(solution)로써 각 입자는 D 차원의 벡터로 표현하고 한 개의 차원은 한 개의 유전자와 매핑이 된다. D-차원(D-Dimension)의 탐색공간에서 N개의 입자들이 매번 반복을 거듭하여 움직인다고 할 때, t번째 반복에서의 i 번째 $(1 \le i \le N)$ 입자 X의 위치는 다음과 같은 벡터형으로 표

현하게 된다.

$$X_i(t) = (x_{i1}, x_{i2}, \cdots, x_{iD})$$

$$x_{ij} = C_k$$

입자의 위치를 표현하는 각 원소 x_{ij}는 i번째 입자에 속한 j번째 $(1 \le j \le D)$ 유전자를 의미하고 유전자가 속한 군집의 번호와 같다. 이는 다음과 같이 표현할 수 있으며 여기에서 C_k는 $(1 \le k \le K)$군집의 번호를 나타낸다.

각각의 입자는 D개의 유전자에 대한 해당 군집 값을 가지며 입력받는 군집의 수에 따라 난수를 사용하여 1~K 까지 어느 한 값에 매핑을 시켜 초기화를 시킨다.

(그림 5)는 100개의 유전자를 5개의 군집으로 나눈다고 하였을 때 입자의 위치를 매핑한 예를 보여주는 것이다.

입자 1	유전자	1	2	3	4	···	99	100
	군집	4	1	4	5	···	3	2
입자 2	유전자	1	2	3	4	···	99	100
	군집	5	1	2	5	···	1	4

(그림 5) 입자의 위치 매핑의 예

3.2 속도 매핑

PSO 군집화 알고리즘에서 속도는 다음과 같이 매핑이 이루어진다. 입자는 하나의 속도 벡터를 가지며 각 속도는 D-차원의 벡터로 이루어진다. 매 t 번째의 반복 때마다 i번째 $(1 \le i \le N)$ 입자의 위치 벡터를 변화시켜주는 요소인 속도 벡터 $V_i(t)$는 다음과 같이 표현된다.

$$V_i(t) = (v_{i1}, v_{i2}, \cdots, v_{iD})$$

속도 벡터를 구성하는 원소인 v_{ij}는 i번째 입자에 속한 j번째 $(1 \le j \le D)$ 유전자의 속도를 나타낸 것으로써 (그림 6)과 같이 한 개의 차원은 한 개의 유전자의 속도와 매핑이 되고 유전자가 속한 군집을 변경시키는데 기여를 한다. 속도 값은 난수를 사용하여 초기화 한다.

입자는 속도에 의해서 다음 위치를 선정하게 되는데 이때 다음 속도 값은 현재의 속도에 *pbest*와 *gbest*를 이용한 위치 값을 더하여 구하고 다음 위치는 현재의 위치 값에 다음의 속도 값을 더하여 구한다. *pbest*는 각 입자의 현재까지 이동 이력 중 최고의 적합도에 해당하는 위치 좌표이고, *gbest*는 모든 입자의 현재까지 이동 이력 중 최고의 적합도에 해당하는 위치 좌표를 표현한다. i번째 $(1 \le i \le N)$ 입자의 *pbest* 정보 P_i와 전체 입자의 *gbest* 정보 P_g는 다음과 같이 표현한다.

입자 1	유전자	1	2	3	4	···	D-1	D
	속도	1	-4	0	2	···	3	3
입자 2	유전자	1	2	3	4	···	D-1	D
	속도	3	-2	1	0	···	0	4

(그림 6) 속도 매핑의 예

$$P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$$

$$P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$$

(그림 7)은 다음 속도와 위치를 구하는 공식을 표현한 것으로서 c_1, c_2는 양의 정수 값을 갖고, 다음 이동할 위치는 현재의 위치에 다음 속도값을 더해주는 것으로 구할 수 있으며 다음 속도값은 현재의 속도값과 *pbest*와 현재 위치의 차, *gbest*와 현재 위치의 차를 더해서 구할 수 있다.

속도 : $V_i(t) = V_i(t-1) + c_1(P_i - X_i(t-1)) + c_2(P_g - X_i(t-1))$
위치 : $X_i(t) = X_i(t-1) + V_i(t)$

(그림 7) PSO 군집화 알고리즘의 속도와 위치 계산 공식

3.3 적합도 함수

유전자 데이터가 PSO 군집화 알고리즘을 이용하여 군집화가 얼마나 잘됐는지를 확인하기 위해 적합도 함수를 사용하고 적합도 값이 클수록 군집화가 잘된 것으로 평가한다. 각 입자의 적합도 값은 오차제곱의 합을 이용하여 계산하고 적합도 값을 구하는 과정은 크게 두 단계로 나뉜다. 첫 번째 단계는 입자의 중심을 구해주는 단계로써 입자의 군집마다 각 차원의 중심(Centroid)을 구한다. 입자의 중심은 군집에 속한 유전자들의 중앙값으로 구한다. x_1 ~ x_{24} 는 첫 번째 군집에서 각 차원의 중심을 나타내며 모든 군집에 대해서 (그림 8)의 계산을 수행한다.

그리고 입자의 오차제곱의 합을 구하기 위해서는 유클리디안 거리(Euclidean distance) 공식을 사용한다. 각 유전자와 군집의 중심 사이의 거리의 차를 제곱하여 더한 뒤 군집의 오차제곱의 합을 구하고 군집의 오차제곱의 합을 더하여 군집화의 오차제곱의 합을 구해준다. 오차제곱의 합 값이 작을수록 군집화의 정확도는 높아진다. 아래는 사용하는 적합도 함수를 나타낸다. N_c는 군집의 수, C_{ij}는 i번째 군집에 속한 j번째 입자를 나타내며, Z_p는 C_{ij}에 속해 있는 데이터

x_1=sum of x_1 val in cluster1 / count of cluster's gene
x_2=sum of x_2 val in cluster1 / count of cluster's gene
...
x_{24}=sum of x_{24} val in cluster1 / count of cluster's gene

(그림 8) 중앙값 계산

벡터, m_j는 각 군집들의 중심을 나타낸다.

$$Fitness = \frac{\sum_{j=1}^{N_c}\left[\sum_{\forall Z_p \in C_{ij}} d(Z_p, m_j)/|C_{ij}|\right]}{N_c}$$

본 실험에서 적용된 적합도 함수를 보면 유전자들을 입자로 선정하여 각 군집안의 유전자들의 발현 값을 다 더해주고 각 군집의 중심을 구한다. 각 군집의 중심과 그 군집에 속해 있는 유전자들의 거리를 구해 더하여 오차제곱의 합을 구한다. 이렇게 구해진 각 클러스터들의 오차제곱의 합 값이 적합도 값이 되고 값이 작으면 작을수록 적합도 값은 높게 평가된다.

4. 구 현

본 장에서는 입력 데이터로 유전자 데이터를 적용하여 구현한 PSO 군집화 알고리즘에 대해 설명한다. 1절에서는 구현환경에 대하여 설명하고 2절에서는 사용한 실험 데이터인 유전자 데이터에 대하여 설명하고 3절에서는 PSO 군집화 알고리즘의 실험 결과에 대하여 기술한다.

4.1 구현환경

PSO 군집화 알고리즘의 성능을 평가하기 위한 시스템 환경은 다음 <표 1>과 같다.

〈표 1〉 구현 환경

구현 환경	
시스템환경	- OS: MS window XP professional spv2 - CPU: Pentium 4 3.20GHz - RAM: 2.00GB
개발환경	- Visual studio 6
개발언어	- C++ 언어

4.2 실험데이터

PSO 군집화 알고리즘의 수행을 위해 사용한 입력 데이터 파일은 3개의 속성을 가진 유전자 파일로써 유전자명(Gene Name), 샘플명(Sample Name), 발현 값(Expression Value)로 이루어졌고 <표 2>에서 이들의 관계를 보여준다. (주)마크로젠에서 제공한 유전자 파일의 샘플 수는 24개이고 각 샘플마다 2000개의 유전자 데이터에 대해 실험을 하였다.

〈표 2〉 유전자 데이터 파일의 값

유전자 / 실험	297784	297912	297990	...
N000310	4.70523273962433	-1.06540479678734	-1.71125530371349	...
N000287	5.11359232185826	-0.30576094332653	1.82213569753198	...
N000288	-1.3389279170472	2.70243282124646	2.70243282124646	...
...	...	...	...	...

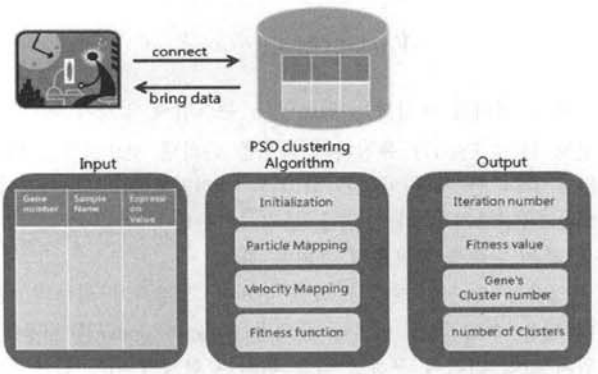
4.3 PSO 군집화 알고리즘 실험 결과

PSO 군집화 알고리즘은 입자의 초기화 과정을 거치면서 처음에는 입자 내 유전자 데이터의 군집을 무작위로 선택하고 그 군집의 중심을 구한다. 그리고 군집의 중심과 다른 데이터와의 거리를 계산하게 되고 이 거리에 따라 군집을 묶은 후 다시 중심을 계산하는 방식으로 진행한다. 이 군집들은 오차제곱의 합을 통해 거리를 입자와 중심의 거리를 계산하여 적합도를 구한다. 이 작업이 모든 데이터에 대해 완료되면 지역적 최고값과 전역적 최고값을 업데이트하고 PSO 알고리즘을 이용하여 탐색을 하면서 군집의 중심을 업데이트한다. PSO Clustering은 Clustering.cpp, PSO.cpp와 PSO.h의 헤더파일로 구현되었으며 현재 Cluster 5개에 100번 반복하여 수행을 하도록 되어있다.

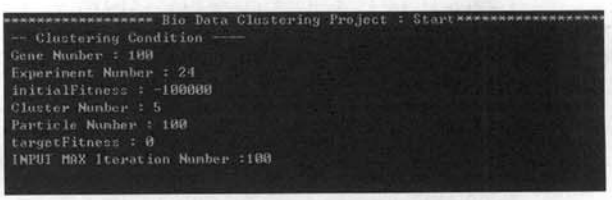
(그림 9)의 PSO 군집화 알고리즘 수행 흐름도를 보면 정규화된 바이오 데이터를 입력 데이터로 사용하여, PSO 군집화 알고리즘을 실행시킨다. PSO 알고리즘이 수행될 때 시스템은 각 입자를 초기화하고 입자와 속도를 매핑한다. 각 유전자가 속하는 군집을 정하기 위해서 적합도 함수를 적용하여 적합도를 계산한 다음 결과 값이 화면으로 출력된다. 결과 화면에는 유전자의 개수, 샘플의 개수, 반복 횟수와 각 유전자가 속한 군집 등에 대한 내용이 포함되어 있다.

(그림 10)은 PSO 군집화 알고리즘을 수행한 화면으로써 알고리즘을 반복 수행할 횟수를 입력받는 화면이다. 군집은 5개를 생성한다. 초기 적합도 값은 -100,000으로써 점점 적합도가 증가하는 형식으로 결과가 나타난다.

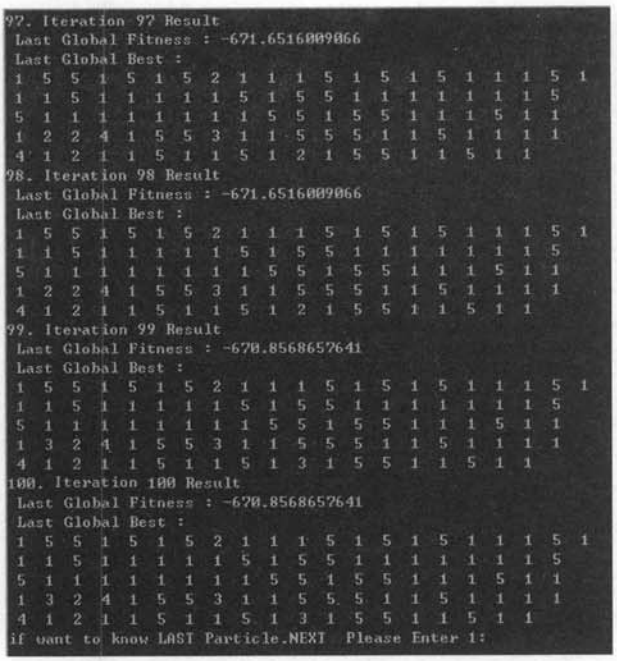
(그림 11)은 알고리즘을 100번 반복수행하고 난 뒤 각 반복마다의 gbest값에 해당하는 유전자가 속한 군집의 번호를 보여주고 있다.



(그림 9) PSO 군집화 알고리즘 수행 흐름도



(그림 10) PSO 군집화 알고리즘 입력 화면



(그림 11) PSO 군집화 알고리즘 결과 화면

5. 성능평가

본 장에서는 PSO 군집화 알고리즘의 성능 평가에 대해 설명한다. 1절에서는 실험 환경에 대하여 설명하고 2절에서는 PSO 군집화 알고리즘의 성능 평가를 위해 기존의 알고리즘인 K-means 알고리즘의 성능과 비교하여 수행한 결과에 대하여 기술한다.

5.1 실험 환경

바이오 칩 데이터 분석을 위한 PSO 군집화 알고리즘의 성능을 평가하기 위한 시스템 환경은 다음 <표 3>과 같다.

〈표 3〉 실험 환경

실험 환경	
시스템환경	- OS: MS window XP professional sp2 - CPU: Pentium 4 3.20GHz - RAM: 2.00GB
데이터	- (주)마크로젠에서 제공한 간암 환자 DNA 칩 데이터 (sample 수: 24개, 유전자 수: 2000개)
군집화 알고리즘	- 비교 성능 평가 PSO 군집화 알고리즘 simple K-means 알고리즘

5.2 실험 결과

PSO 군집화 알고리즘의 성능을 측정하기 위해 PSO 군집화 알고리즘의 수행시간과 군집화 품질을 측정하였다. 먼저 반복 횟수와 군집의 수에 따라 수행시간을 측정하여 PSO 군집화 알고리즘의 수행속도를 측정하였다. 이에 따른 수행속도를 비교한 표와 분석 그래프를 각각 <표 4>와 (그림 12)에서 보여준다.

(그림 12)에서 x축은 반복횟수, y축은 수행속도를 나타낸다. 군집의 수와 반복횟수를 늘려가며 수행속도를 측정하였다. 반복횟수가 적을 경우, 군집의 개수는 수행속도에 큰 영향을 미치지 않는 것으로 나타났지만 반복횟수가 많아질수록 군집의 개수가 수행속도에 영향을 미치는 것으로 나타났다.

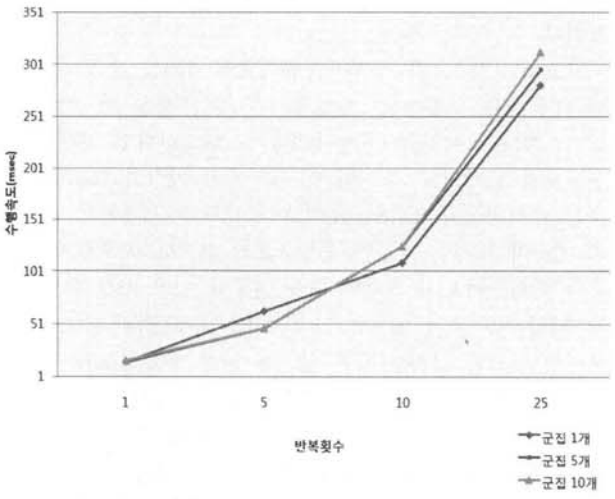
다음은 유전자의 개수에 따른 PSO 군집화 알고리즘의 수행속도를 측정하고, 기존 데이터 분석 소프트웨어로 Simple K-means 알고리즘의 수행속도를 측정하여 비교 분석하였다. 이 때 K=5로 설정하였다. 이에 따른 수행속도를 비교한 표와 분석 그래프는 <표 5>와 (그림 13)에서 보여준다.

(그림 13)에서 x축은 유전자의 개수, y축은 수행속도(msec)를 나타낸다. 같은 데이터 셋과 실행 환경을 가지고 PSO 군집화 알고리즘의 수행시간과 기존의 Simple K-means 알고리즘 소프트웨어와 수행시간을 비교했을 때 높은 성능 향상을 나타내었다. 특히 유전자의 개수가 늘어남에 따라 PSO 군집화 알고리즘의 수행속도의 변화가 크지 않으므로 많은 데이터를 사용하여 분석을 할 때 PSO 군집화 알고리즘이 효율적이라는 것을 알 수 있다. 예를 들어 유전자의 수가 200개인 경우에는 PSO 분류화 알고리즘의 수행시간은 62msec이고 K-means 알고리즘의 수행시간은 72msec으로써 10msec의 시간이 단축되었고 평균적으로 10% 이상의 성능 향상을 보인다.

그리고 샘플 수를 다르게 하여 PSO 군집화 알고리즘과 K-means 알고리즘의 수행속도를 비교해 보면 같은 수의

〈표 4〉 반복횟수와 군집의 수에 따른 성능 비교 표

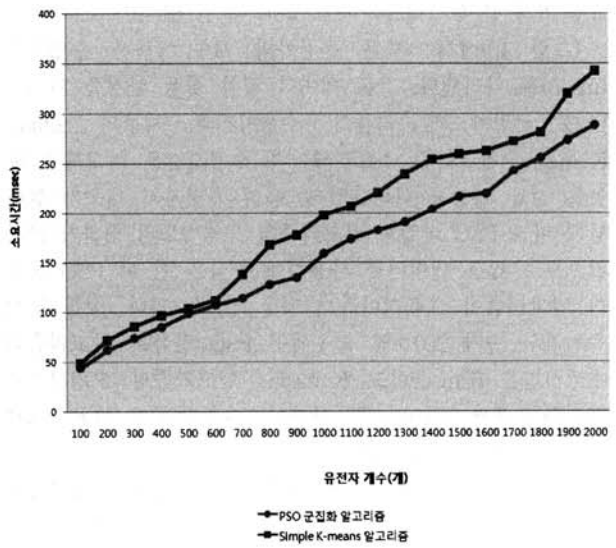
반복횟수	군집1개	군집5개	군집10개
1번	15	15	16
5번	63	47	47
10번	109	125	125
25번	281	296	313



(그림 12) 반복횟수와 군집의 수에 따른 수행속도 비교 그래프

〈표 5〉 PSO 군집화 알고리즘과 K-means 알고리즘의 성능 비교 표

유전자개수 (샘플 24개)	PSO 군집화 알고리즘 (msec)	K-means 알고리즘(msec)
200개	62	72
400개	85	97
800개	128	168
1200개	183	221
1600개	220	263
2000개	289	343



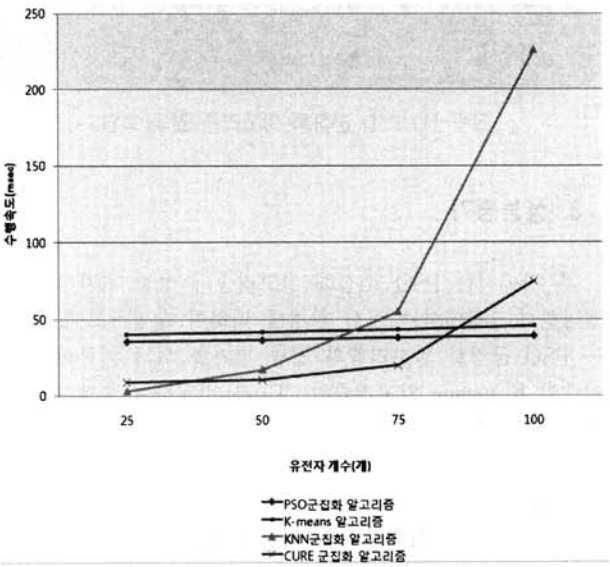
(그림 13) PSO 군집화 알고리즘과 K-means 알고리즘 성능 비교 그래프

데이터라도 샘플 수가 적을수록 군집화의 수행속도가 빠르다는 것을 알 수 있다. 더 자세한 수행성능의 측정을 위해 KNN 군집화 알고리즘과 CURE 군집화 알고리즘의 샘플 수에 따른 수행속도와 함께 비교하였다. <표 6> 및 (그림 14)와 (그림 15)는 샘플 수가 12개와 24개일 때 유전자의 수를 다르게 하여 수행시간을 비교한 표와 이를 그래프로 나타낸 것이다.

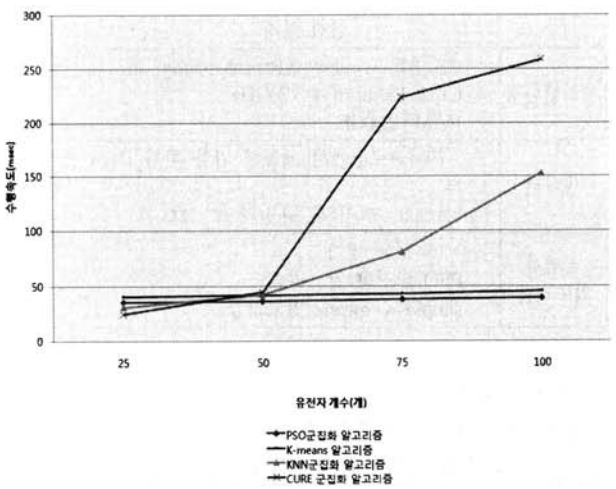
<표 6> 및 (그림 14)와 (그림 15)의 결과를 보면 샘플 수와 유전자의 수에 따라 수행속도가 갖는 값을 비교해볼 수 있다. 예를 들어 1200개의 데이터를 비교한다고 하였을 때 샘플수가 12개이고 각 샘플별 유전자의 개수가 100개인 경우는 수행속도가 39.18msec이고 샘플수가 24개이고 유전자의 개수가 50개인 경우는 수행속도가 41.87msec으로써 같은 수의 데이터라도 유전자의 수와 샘플의 수에 따라 수행속도가 달라지는 것을 알 수 있으며 KNN 군집화 알고리즘은 데이터의 수와 샘플의 수의 증가에 따른 수행속도가 급격히 증가하는 반면에 PSO 군집화 알고리즘은 데이터의 수와 샘플의 수의 증가에 따른 수행속도의 증가가 크지 않음을 알 수 있다. 특히 CURE 군집화 알고리즘은 계층적 군집화 수행 과정으로 인해 유전자 수의 증가에 따른 수행 시간의 증

〈표 6〉 샘플 수에 따른 PSO 군집화 알고리즘과 K-means 알고리즘, KNN 군집화 알고리즘, CURE 군집화 알고리즘의 성능 비교 표

Sample의 수	Gene의 개수	PSO 군집화 알고리즘 수행시간 (msec)	K-means 알고리즘 수행시간 (msec)	KNN 군집화 알고리즘 수행시간 (msec)	CURE 군집화 알고리즘 수행시간 (msec)
12	25	35.59	40.43	3.12	9.03
	50	36.42	41.88	17.4	10.51
	75	37.92	43.10	54.96	20.08
	100	39.18	45.59	226.35	74.82
24	25	39.26	44.13	30.84	24.35
	50	41.87	46.17	42.29	44.72
	75	42.61	47.65	80.87	224.01
	100	43.60	49.82	152.7	258.83



(그림 14) PSO 군집화 알고리즘과 K-means 알고리즘, KNN 군집화 알고리즘의 성능 비교 그래프 (샘플 수: 12개)



(그림 15) PSO 군집화 알고리즘과 K-means 알고리즘, KNN 군집화 알고리즘의 성능 비교 그래프 (샘플 수: 24개)

〈표 7〉 실험에 대한 Class

실험	Class	실험	Class
N00287	-1	N00299	1
N00288	1	N00300	1
N00289	1	N00301	-1
N00290	1	N00302	1
N00291	-1	N00303	-1
N00292	1	N00304	-1
N00293	1	N00305	1
N00294	-1	N00306	-1
N00295	-1	N00307	1
N00296	-1	N00308	-1
N00297	-1	N00309	1
N00298	1	N00310	1

가가 비약적으로 커지는 것을 알 수 있다.

군집화의 정확도는 군집들의 분별력을 이용하여 판단할 수 있다. 이를 위하여 Cluster_Diff라는 수치를 토대로 다른 기법들과 정확도를 비교하였다. 본 실험에서 샘플을 구성하고 있는 실험은 암에 대한 판단에 효용성이 있느냐에 따라서 Class가 1(암에 대한 판단에 영향을 미침)과 -1(암에 대한 판단에 영향을 미치지 않음)로 구분된다. 예를 들어, 실험 N00287~ N00310의 Class는 <표 7>과 같다.

이를 이용해 각 Cluster에 대한 성능을 비교할 수 있다. 군집은 각 군집에 포함된 유전자로 구성된다. <표 8>은 유전자 0, 유전자7로 구성된 임의의 군집을 나타낸다.

여기서 Xn과 Yn 등은 각 유전자와 실험군에 대한 구별을 쉽게 하기 위한 표기이다. 실제 이들 각각은 실험에 대한 발현값이다. 위와 같은 임의의 군집 α에 대해서, 아래와 같은 Cluster_Diff를 계산한다.

$$\text{Cluster\_Diff} = \frac{}{+\begin{pmatrix}\{(X0, X1, X2\text{의 평균}) - (Y3, Y4\text{의 평균})\}^2 \\ \{(Z0, Z1, Z2\text{의 평균}) - (A3, A4\text{의 평균})\}^2\end{pmatrix}}$$

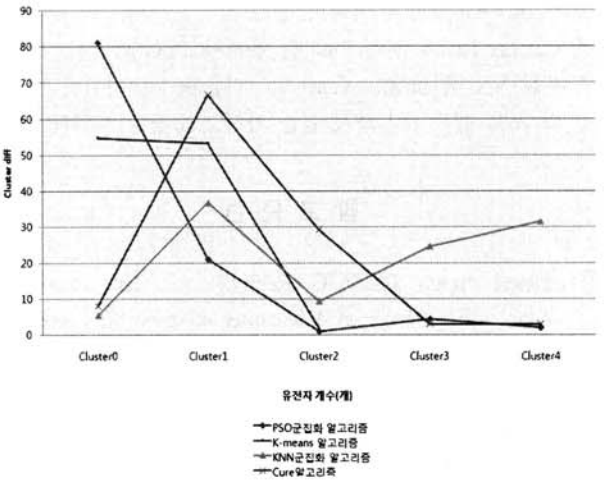
Cluster_Diff의 값에 따라서 값이 매우 큰 경우, 실험군(1)과 실험군(-1)의 차이가 매우 커서 차별화를 하기 좋은 유전자들끼리 군집을 이룬다는 의미로 Positive Good Cluster_Diff. 그 값이 매우 작은 경우, 실험군(1)과 실험군(-1)의 차이가 매우 작아서 차별화를 하기 어려운 유전자들끼리 군집을 이룬다는 의미로 Negative Good Cluster_Diff라고 할 수 있다. 이 Cluster_Diff값들은 각 Cluster에 대해 존재하며, (그림 16)의 그래프로 나타낼 수 있다.

이 때, 이 Cluster_Diff에 대한 분산으로 각 군집의 분별력을 판단할 수 있다. 분산이 크면 클수록, 각 군집의 특징이 분명하다는 의미이므로, 좋은 군집화 결과라는 말과 같다. 즉, 간암환자와 비환자의 차별화를 할 수 있는 군집들과 차별화를 하지 못하는 군집들로 나눈 성질이 분명하다는 것이다. PSO 군집화 알고리즘과 K-means, KNN, Cure 군집화 알고리즘의 Cluster_Diff에 대한 분산은 <표 9>와 같다.

다른 알고리즘에 비해 PSO 군집화 알고리즘의 분산이 비교적 높은 값을 보이며 각 군집의 분별력이 높다는 것을 알 수 있다.

〈표 8〉 클러스터 α의 구성 예

실험 / 유전자	실험 Class =1인 실험들			실험 Class=-1인 실험들	
gene0	X0	X1	X2	Y3	Y4
gene7	Z0	Z1	Z2	A3	A4



(그림 16) 각 알고리즘의 군집화 결과에 대한 Cluster_Diff 비교

〈표 9〉 군집화 알고리즘들에 대한 Cluster_Diff의 분산

군집화 알고리즘	Cluster_Diff의 분산
PSO	30.4599
K-NN	12.2661
K-means	24.7381
CURE	24.3249

6. 결 론

다양한 생물 정보에 대한 편리하면서도 빠르고 정확한 분석 기술이 중요해짐에 따라 실험 정보를 담은 바이오 칩에 대한 통합 분석의 필요성이 크게 대두되고 있다. 바이오 칩을 사용하면 수 시간 내 분석이 가능하기 때문에 비용과 시간을 절약할 수 있고 바이오 칩 기술을 이용하여 생물의 생명현상에 대한 근본적 원리와 구조를 분석, 새로운 신약을 개발하거나 인간의 뇌의 정보처리 메카니즘을 분석할 수 있을 뿐만 아니라 질병을 진단하고 예측할 수 있다. 본 연구에서는 새, 물고기, 벌 등 군집 생활을 하는 동물들의 행동 습성을 모방하여 최적의 해를 찾는 Particle Swarm Optimization (PSO) 알고리즘에 기반한 군집화 알고리즘을 제안하고 구현하였다. PSO 군집화 알고리즘을 바이오 칩 데이터에 적용함으로써 KNN 군집화 알고리즘, CURE 알고리즘, K-means 알고리즘과 비교하여 데이터가 많아질수록 수

행성능이 우수함을 실험을 통해 증명하였다. PSO 군집화 알고리즘은 빠른 시간 안에 좋은 품질의 군집화를 수행함으로써 이후의 분석 단계인 분류화 작업의 정확도와 효율성을 높일 수 있다. 그리하여 PSO 군집화 알고리즘이 향후 효율적인 질병의 판단에 적용할 수 있는 방향을 제시하였다.

PSO 군집화 알고리즘은 데이터의 크기가 다항 증가하더라도 수행속도는 반복횟수에 근거하여 선형에 근사한 증가를 보여준다는 것을 실험을 통해 확인할 수 있었다. PSO 군집화 알고리즘을 이용하여 수천 개부터 많게는 수십만 개의 기록을 가진 대량의 바이오 칩 데이터를 보다 빠르고 효율적으로 처리하고 분석해서 높은 성능을 얻을 수 있다. 또한 Cluster_Diff의 분산 비교를 통해서 PSO 군집화 알고리즘이 K-NN, K-means, Cure 알고리즘과 비교하였을 때, 최소 약 25% 높은 군집화 품질을 가지고 있음을 보였다.

참 고 문 헌

[1] Dinesh Singh, Phillip G. Febbo, et al, Gene expression correlates of clinical prostate cancer behavior, the Center Cell vol.1, issue2, March 200, pp.200-209

[2] Barrett MT, Scheffer A, et al, Comparative genomic hybridization using oligonucleotide microarrays and total genomic DNA, Proc. National Academy of Sciences USA, 2004 Dec. 21

[3] D.J. Lockhart, H.L. Dong, M.C. Byrne, M.T. Follettie, M.V. Gallo, M.S. Chee, M. Mittmann, C.W. Wang, M. Kobayashi, H. Horton, E.L. Brown, Expression monitoring by hybridization to high-density oligonucleotide arrays, Nature Biotechnology, Vol.14 No.13, pp.1675-1680, 1996.

[4] J.L. DeRisi, V.R. Iver, P.O. Brown, Exploring the metabolic and genetic control of gene expression on a genomic scale, Science, Vol.278 No.5338, pp.680-686, 1997.

[5] C. Debouck, P.N. Goodfellow, DNA microarrays in drug discovery and development, Nature Genetics, Vol.21 No.1 suppl, pp.48-50, 1999.

[6] D. Bowtell, J. Sambrook, DNA Microarrays, CSHL Press, 2002.

[7] E.K. Tang, P.N. Suganthan and X.Yao, Feature Selection for Microarray Data Using Least Squares SVM and Particle Swarm Optimization, Computational Intelligence in Bioinformatics and Computational Biology(CIBCB), pp.1-8, 2005

[8] Qi Shen, Wei-Min Shi, Wei Kong, Bao-Xian Ye, A combination of modified particle swarm optimization algorithm and support vector machine for gene selection and tumor classification, Talanta, Vol.71, No.4, pp.1679-1683, 2007

[9] Hualong Yu, Guochan Gu, Haibo Liu, Jing Shen, Changming Zhu, A Novel Discrete Particle Swarm Optimization Algorithm for Microarray Data-based Tumor Marker Gene Selection, International Conference on Computer Science and Software Engineering, pp.1057-1060, 2008

[10] M.B. Eisen, P.T. Spellman, P.O. Browndagger, D. Botstein, Cluster analysis and display of genome-wide expression patterns, Proceedings of the National Academy of Sciences of the United States of America (PNAS), Vol.95, No.25, pp. 14863-14868, 1998.

[11] X. Xiao, E.R. Dow, R. Eberhart, Z.B. Miled, R.J. Oppelt, Gene Clustering Using Self-Organizing Maps and Particle Swarm Optimization, IEEE International Workshop On High Performance Computational Biology, 2003.

[12] I.D. Falco, A.D. Cioppa, E. Tarantino, Facing classification problems with Particle Swarm Optimization, Soft Computing, Vol.7, No.3, pp.652-658, June 2007.

[13] DNA chip. http://mbel.kaist.ac.kr/research/DNAchip_en.html

[14] J. Han, M. Kamber, Data Mining Concepts and Techniques, Morgan Kaufmann, 2001.

[15] I.H. Witten, E. Frank, Data Mining : Practical Machine Learning Tools and Techniques, 2nd edition, Morgan Kaufmann, 2005.

[16] J.H. Holland, Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, 1992.

[17] Sudipto Guha, Rajeev Rastogi, Kyuscok Shim, CURE: And Efficient clustering Algorithm for Large Databases, Proc. ACM SIGMOD Int Conf. on Management of Data, pp.73-84, New York, 1998

[18] K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through Particle Swarm Optimization, Natural Computing Vol.1, No.2-3, pp.235-306, June 2002.

[19] Y. Shi, R.C. Eberhart, Parameter selection in particle swarm optimization, Proceedings of Evolutionary Programming VII, pp.591-600, 1998

이 민 수

e-mail : mlee@ewha.ac.kr

1992년 서울대학교 컴퓨터공학과(학사)

1995년 서울대학교 컴퓨터공학과(공학석사)

2000년 University of Florida 컴퓨터공학과(공학박사)

1995년~1996년 LG전자미디어통신연구소 연구원

2000년~2002년 미국 Oracle Corporation, Senior Member of Technical Staff

2002년~현 재 이화여자대학교 컴퓨터공학과 부교수

관심분야 : 데이터웨어하우스, 데이터마이닝, XML, 스트림 데이터 처리