임베디드 시스템을 위한 PSO 기반의 군집화 알고리즘의 구현

맹보연, 최옥주, 이민수
이화여자대학교 컴퓨터공학과
mngby@ewhain.net, pensica@ewhain.net, mlee@ewha.ac.kr

The implementation of PSO clustering Algorithm for Embedded Systems

Boyeon Meang, Ok-ju Choi, Minsoo Lee
Dept. of Computer Science and Engineering, Ewha Womans University

요 약

바이오 칩 분석 시스템은 유전자와 실험의 두 축으로 이루어진 바이오 칩에서 자료를 추출하고 필요한 정보를 얻기 위해 데이터를 분석하는 시스템이다. 유전자 데이터를 효율적으로 분석할 수 있는 방법으로 바이오 칩 분석 시스템이 각광받으면서 데이터의 양과 종류가 방대해지고 메모리의 효율적인 사용과 이에 따른 속도 개선을 위해 임베디드 시스템이 필요해지고 있다. 이에 따라 본 연구에서는 임베디드 시스템을 위한 PSO 기반의 군집화 알고리즘을 구현하였다. 방대한 양의 유전자 데이터를 분석하기 위해 생태계 모방 알고리즘인 Particle Swarm Optimization 알고리즘과 비슷한 유전자의 분류를 위한 기법으로 군집화를 사용하여 유전자 데이터의 통합 분석 시스템을 구현, 사용자에게 더욱 효율적으로 정보를 제공한다. 본 논문에서는 방대한 양의 데이터의 최적화에 효율적인 생태계 모방 알고리즘 Particle Swarm Optimization 을 이용하여 데이터들을 군집화하는 알고리즘을 임베디드 시스템을 위해 구현한 방법을 기술하고 있다

1. 서론[1]

바이오 칩을 사용하기 이 전의 유전자의 분석은 시간이 많이 소요 되었을 뿐만 아니라 인력 소요가 불가피 하였다. 또한 시간을 줄이는 방법은 정확도가 떨어지거나 비용이 많이 들었다. 바이오 칩은 기존의 분자생물학적 지식에다 고도의 기계 및 전자공학 기술을 접목하여 만들어 졌으므로 인간의 유전자 뿐 아니라 미생물의 유전자도 실제 실험에 비해 적은 비용과 시간으로 분석할 수 있다. 이로 인한 국내의 바이오 칩[1]에 대한 관심은 선진국과 비슷한 시기에 시작되었고, 실험 정보를 담은 바이오 칩에 대한 통합 분석[2]의 필요성이 크게 대두 되고 있다. 또한 데이터의 양이 점점 커짐에 따라 수행속도가 점차 느려지게 되었다. 이에 따라 메모리의 효율성이 바이오 칩에 대한 통합 분석을 위해 본 논문에서는 바이오 칩 통합 분석 시스템으로 임베디드 시스템을 위한 PSO 군집화 알고리즘을 구현하였다. 바이오 칩에 대한 유전자 정보의 유사성을 알아 내기 위해 군집화 기법과 많은 양의 정보에 의한 수행속도의 한계를 극복하기 위하여 PSO 알고리즘으로 이루어진 PSO 군집화 알고리즘을 구현 및 개발하였다.

[1] 이 논문은 정부(교육과학기술부)의 재원으로 한국과학재단 지원을 받아 수행된 연구임(No. R01-2008-000-20029-0).

논문의 구성은 다음과 같다. 2 장에서 K-means 군집화 알고리즘과 Particle Swarm Optimization 을 각각 기술하고 PSO 군집화 알고리즘에 대해서 언급한다. 3 장에서 임베디드 시스템을 위한 PSO 구현 방법에 대해 설명한다. 4 장에서는 PSO 군집화 알고리즘의 구현 결과를 설명하고 마지막으로 5 장에서 결론과 향후 연구 과제를 기술한다.

2. 관련연구

2.1 K-means 군집화

군집화란, 트레이닝 없이 유사도에 근거하여 데이터를 군집들로 구분한다. 군집 내의 유사성을 극대화하고, 군집간의 유사성을 최소화 하여 데이터의 객체를 분석한다. 군집 간의 유사도를 평가하기 위해 사용되는 거리 측정 함수로는 Euclidean distance, Mahalanobis distance 등이 있다. 군집화 기법은 다양한데 그 중 K-means 군집화는 거리에 기반을 둔 기법으로 기준점에 가까운 곳의 데이터들을 하나의 군집으로 묶는 방법이다. K 개의 군집 수와 위치가 임의로 초기화 되고 각각의 데이터에 대해 K 개의 위치까지의 거리를 구한 후, 가장 가까운 군집으로 분할한다. 군집으로 나뉘어진 데이터를 기준으로 군집 중앙의 위치를 계산하고 새롭게 구해진 군집 중앙의 위치가 기존과 동일하거나 주어진 조건을 만족하면 알

고리즘을 종료한다. 군집의 개수인 K 는 사용자에 의해 주어지는 값에 따라 군집화에 많은 영향을 받는다[3]. K-means 군집화는 분할적 군집화로써 데이터의 객체들을 평면적으로 중복이 없는 부분집합으로 나눈다[4][5].

2.2 Particle Swarm Optimization 알고리즘

PSO (Particle Swarm Optimization) 알고리즘은 생태계 모방알고리즘 중 하나로써 군집 생활을 하는 동물들의 행동 습성을 모방하여 최적의 해를 찾는 알고리즘이다. 반복을 거치며 각 입자들은 위치를 변화시키게 되는데, 두 개의 참고지점을 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 최고 지점 (Global Best Position, gbest) 라고 한다.

적합도 함수를 이용하여 각 입자의 우수성을 평가하여 새로운 지역적 최고 지점과 전역적 최고지점 (Global Best Position, gbest)이 생기면 정보를 업데이트 하게 된다. 최종적으로 모든 과정이 끝났을 때의 저장된 전역적 최고지점 위치 벡터가 가장 최적의 해로 도출된다.

PSO 알고리즘은 방대한 양의 데이터를 분석하고 그 데이터를 분류할 규칙을 찾아내는데 매우 적합하며 특징으로는 복수의 탐색점을 가지며 각 탐색점의 pbest 와 gbest 를 이용하여 각 탐색점을 확률적으로 변경시켜가는 것에 의해 전역적인 최적해를 발견하며 연속형의 변수와 이상형의 변수가 혼합되어 있는 경우에도 전체적인 집단화가 가능하다. 또한 n 차원 공간으로 확장할 수 있다[6].

2.3 PSO 군집화 알고리즘

PSO 군집화는 K-means 군집화와 비슷한 과정을 거쳐 수행되는데, 무작위로 선택된 입자가 군집의 중심이 되어 다른 입자와의 거리를 계산하게 되고 이 거리에 따라 군집을 결정한 후 다시 중심을 계산하는 방식으로 진행한다. 이 군집들은 SSE 즉 Sum Squared Error 를 통해 입자와 중심의 거리를 계산하여 적합도를 구하며 이 후 최고값을 update 한다[7]. PSO 군집화 알고리즘의 pseudo code 는 다음과 같다.

1. Initialize each particle to contain N_c randomly selected cluster centroids.
2. For i = 1 to t_{max} do
   (a) For each particle i do
   (b) For each data vector z_p
      i . calculate the Eudidean distance d(z_p, m_{ij}) to all cluster centroid C_{ij}
      ii . assign z_p to cluster C_{ij} such that d(z_p, m_{ij}) = $\min_{\forall c=1_c,\ldots,N_n}$(d($z_p$, m_{ij}))
      iii . calculate the fitness
   (c) Update the global best and local best positions
   (d) Update the cluster centroids

(그림 1) PSO 군집화 알고리즘

가장 먼저 초기화를 위해 (1)각 입자는 무작위로 cluster 의 중심을 선택한다. 이후, (2)명시된 종료 시점까지 loop 을 돌면서 각 입자에 속하는 각 data vector 는 모든 군집 중심까지의 거리를 계산하고 거리가 가장 짧은 군집에 포함된다. 이후 전체에 적합도 함수를 적용하여 적합도를 계산한다. 이 작업이 모든 data vector 에 대해 완료되면 전역 최고 지점 값과 지역 최고 지점 값을 update 하고 PSO search 를 이용하여 군집의 중심을 update 한다.

이 알고리즘의 정확도를 확인하기 위해 Sum Squared Error 군집화를 사용한다. Sum Squared Error 는 제곱 값이 최소인 것으로 군집화를 한다. 적용되는 Sum Squared Error 는 다음과 같다.

$$seK_i = \sum_{j=1}^{m} \| t_{ij} - C_k \|^2$$

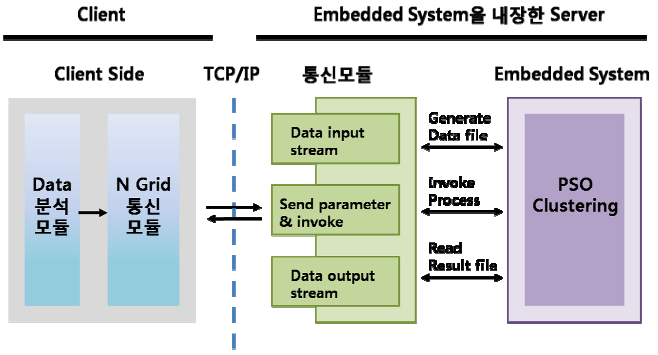
Squared Error 는 중심값이 Ck 인 군집에서 각각의 elements 들과 중심값 사이의 거리 제곱 값들의 합이다. 적용되는 적합도 함수는 다음과 같다. 이 값이 작을수록 적합도는 높다고 평가된다. 하지만 본 논문에서는 사용자의 이해성을 돕기 위해 - 값을 + 값으로 변환하여 적합도의 값이 클수록 군집이 잘 된 것으로 평가하게 구현되었다.

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

3. PSO 군집화 알고리즘을 임베디드 시스템에 적용

분산되고 방대한 데이터를 처리하기 위해서 기존의 데이터 전송방법으로는 성능 및 안정성에 한계가 있다. 즉, 많은 양의 데이터를 안정적으로 지원할 수

있는 확장성과 빠르게 변하는 환경에 대처할 수 있는 적응성 및 결함에도 대처할 수 있는 가용성이 필요하다. 이에 따라 본 연구에서는 대용량 DNA 칩 분석 데이터를 전송 시 임베디드 시스템에 전달하는 함수 호출 및 결과 반환 기법을 통신 모듈에서 지원할 수 있도록 API 를 설계하고 구현하였다. 또한 Stand-alone 으로 동작하던 생태계 모방 분석 알고리즘을 임베디드 시스템에 적용하기 위해서 구조를 아래와 같이 C/S 환경으로 바꾸어 개발하였다.



(그림 2) 임베디드 시스템을 내장한 C/S 환경

데이터 분석 SW 를 통해 바이오 데이터를 전송한 후 사용자의 편리에 따라 알고리즘이 선택되면 통신 모듈을 통해 전달되고 이를 통해 임베디드 시스템내에 내장된 생태계 모방 알고리즘인 PSO 군집화 알고리즘을 호출하게 된다. 호출된 알고리즘은 실행 후 결과를 다시 통신 모듈을 통해 클라이언트로 전송한다.

4. 실험 및 결과

간암 환자와 정상인의 데이터를 바탕으로 통합 데이터 마이닝을 적용하였다. 각기 N1, N2, N3 의 정상인과 C1, C2, C3 의 간암환자의 데이터를 사용하였다. 첫 번째 열은 이 데이터가 N1~ C3 중 어디에 속하는지를 말하며, 두 번째 열은 pathway 별로 묶은 그룹 (Up &Down)으로 예를 들어 Glutathine metabolism_UP 은 Glutathine metabolism 라는 pathway 에서 signal 값이 높은 그룹, Glutathine metabolism_DOWN 은 Glutathine metabolism 라는 pathway 에서 signal 값이 낮은 그룹을 말한다. 세 번째 열은 signal 값들이다.
아래는 데이터 파일을 표로 나타낸 것이다.

<표 1> 유전자 정보 파일

Class	Pathway (UP&DOWN)	Signal
N1	Alanine and aspartate metabolism_UP	544.825
N1	Alkaloid biosynthesis I_UP	565.6
N1	Alginine and proline metabolism_DOWN	1386.45
…	…	…
C3	Urea cycle and metabolism of amino groups _UP	6120.065
C3	Valine, leucine and isoleucine degradation_UP	2666.128

이 데이터를 C/S 환경으로 구성된 임베디드 시스템 환경에 맞추기 위해 알고리즘 수행을 다음의 API 로 개발하여 포팅하였다.

● system("./PSO 군집화/PSOCLU data.txt 10 10");
Client 에서 Select Job 중 3 번 PSO 군집화를 선택
이 작업은 Processor 로 넘어가서 Processor 는 PSO 군집화를 수행한다.
- pej_sort_n.txt :
각 실험에 대한 유전자의 정보 데이터
- pso_clu_pej_sort_n.txt

● **PSO 군집화내 주요 API**

- 입력 API
PSO_Simulator::BioDataRead(char*fileName)
: fileName, pej_sort_n.txt 로부터 데이터를 읽어와 토큰으로 실험, gene, 발현값을 구분한다
- 출력 API
PSO_Simulator::PrintResult(int iteration)
: iteration :반복횟수
결과를 result.txt 와 화면에 출력한다

- void PSO_Simulator::Iterate()
:반복 횟수만큼 반복하여 fitness 값을 구하고 pbest 와 gbest 를 선정한다

(그림 3) PSO 군집화 알고리즘 API

생물정보 통합 데이터를 입력받아 PSO 군집화 알고리즘을 적용하면 Client 에서 입력받은 데이터를 Server 로 보내고 Server 에서는 Processor 에게 데이터를 보내서 수행시킨다. Client 와 Processor 에 출력된 결과 값은 각 군집에 속하는 Pathway 와 10 개의 군집이 10 번 반복적으로 군집화를 하면서 반복횟수에 따라 군집들의 적합도를 나타내는 적합도의 값을 보여준다. 실험결과는 다음과 같다.

```
Client - [INFO] connect to server success

 - Bio Algorithm Demo -


======================
 [1] Ant_Clustering
 [2] Ant_Classfication
 [3] PSO_Clustering
 [4] PSO_Classfication
 [5] Exit
======================
Select Job : 3
Output file? [y/n] : n
Client - [INFO] send data file
```

(그림 4) 초기 Client 실행 화면

```
In file included from ./Processor/PSOClassification/main.c:3:
./Processor/PSOClassification/main.h:56: warning: built-in function 'exp' d
ed as non-function
In file included from ./Processor/PSOClassification/Read_Data.c:1:
./Processor/PSOClassification/main.h:56: warning: built-in function 'exp' d
ed as non-function
g++ -o ./Processor/Processor.o -c ./Processor/Processor.c -Wall -ansi
g++ -o ./Processor/Processor ./Processor/Processor.o -Wall -ansi
find . -type f -name "*.o" -exec rm -f {} \; -print
./Server/Server.o
./Client/Client.o
./Processor/Processor.o
./Processor/PSOClassification/Read_Data.o
./Processor/PSOClustering/PSO.o
dwlab@dwlab-desktop:~/dna[4]_by.tar$ cd Server
dwlab@dwlab-desktop:~/dna[4]_by.tar/Server$ ./Server
Server - [INFO] waiting
Server - [INFO] accept PROCESSOR
Server - [INFO] waiting
Server - [INFO] accept CLIENT
Server - [INFO] waiting
Server - [INFO] Transport Data from Client to Processor
Server - [INFO] waiting
```

(그림 5) Server 초기 실행 화면

```
*************************************************************************
                        Iteration 1 Result
*************************************************************************

 Last Global Fitness : 471596.2902580028
 Last Global Best : 471596.2902580028

 Cluster 1 has 3 gene :

Cyanoamino acid metabolism_UP Glutathione metabolism_DOWN Porphyrin and chloroph
yll metabolism_UP
************************** finish making cluster 1 ***************************

 Cluster 2 has 9 gene :

C21-Steroid hormone metabolism_UP Carbon fixation _UP Fatty acid biosynthesis (p
ath 2)_UP Pantothenate and CoA biosynthesis _UP Pentose and glucuronate intercon
versions_UP Pentose phosphate pathway _DOWN Starch and sucrose metabolism _UP St
arch and sucrose metabolism _DOWN Styrene degradation _UP
************************** finish making cluster 2 ***************************

 Cluster 3 has 7 gene :

gamma-Hexachlorocyclohexane degradation _UP Glyoxylate and dicarboxylate metabol
```

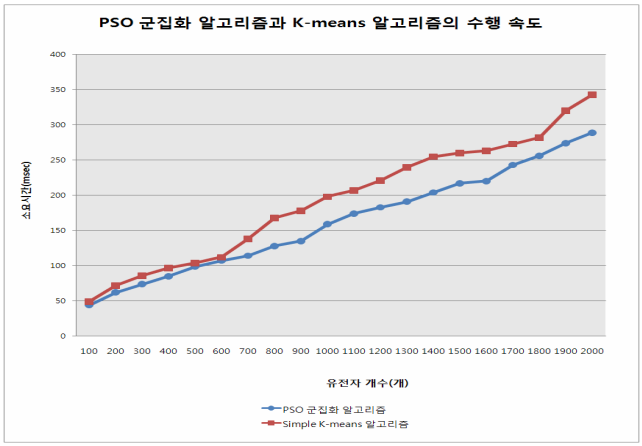
(그림 6) Processor 실행화면

이 결과는 각 군집 안에 속한 유전자의 Pathway 와 유전자의 개수, 군집화의 반복횟수 및 적합도의 값을 사용자에게 보여주며 분할된 군집 안에 있는 유전자들이 서로 유사하다고 평가한다. 적합도의 값이 클수록 군집화가 잘 된 군집이다.
반복 횟수에 따른 적합도의 값은 다음과 같다.

<표 2> 반복횟수에 따른 적합도의 값

Iteration No.	Fitness
Iteration 1	471596.2902580028
Iteration 2	507660.0965899325
Iteration 3	571800.967257312
Iteration 4	595022.7977038304
Iteration 5	595022.7977038304
Iteration 6	642515.4753359372
Iteration 7	703844.009616096
Iteration 8	753717.0720361953
Iteration 9	789205.976440297
Iteration 10	962872.1756034453

같은 데이터 셋과 실행환경에서 PSO 군집화 알고리즘과 Simple K-means 알고리즘의 성능을 비교하였다. x 축은 유전자의 개수, y 축은 수행속도(msec)를 나타낸다.



(그림 7) PSO 군집화와 Simple K-means 의 수행속도 비교

유전자의 개수가 늘어남에 따라 PSO 군집화 알고리즘의 수행속도의 변화가 크지 않으므로 유전자와 같이 데이터의 양이 많을 때 더 효율적이라는 것을 알 수 있다.

5. 결론 및 향후 연구

임베디드 시스템을 위한 PSO 군집화 알고리즘을 구현하였다. PSO 군집화 알고리즘은 생태계를 모방한 알고리즘을 기반으로 하여 군집화를 하는 방법으로 사용자로부터 군집의 개수와 반복 횟수를 입력 받아 군집의 중심을 찾고 그 중심을 기준으로 데이터들을 군집하며 거리가 짧은 유전자를 찾아 분할한다. 이 군집들의 적합도를 측정하기 위한 함수로 적합도 함수를 적용하였고 PSO 군집화 알고리즘을 제안 및 구현하여 임베디드 시스템에 탑재, 바이오 칩 분석을 위한 생태계 모방 군집화를 수행하였다. 정확도와 효율성을 평가하여 처리를 분산화 시키는 임베디드 시스템을 사용한 성능 및 개선 효과를 분석하였고 이로 인해 수행 속도 면에서 효과 적이라는 것을 알 수 있었다. 하지만 초기값 선택의 민감함 때문에 제대로 군집이 되지 않는 경우도 있으므로 임의로 추출하는 초기화 문제에 대해서 값이 변화하지 않고 일정하게 값을 얻을 수 있게 하여 분할된 군집들이 최적화 될 수 있는 방법과 임베디드 시스템의 성능 및 효율성을 높이기 위해 새로운 알고리즘을 탑재하는 방법을 연구 할 것이다.

참고문헌

[1] Sorin Craghici, “Data Analysis Tools for DNA Microarrays,” Chapmen & Hall, 2003
[2] 이윤경, 윤혜정, 이민수, 윤경오, 최혜연, 김대현, 이근일, 김대영 “Particle Swarm Optimization 알고리즘을 이용한 바이오칩 데이터의 군집화 및 분류화 기법,” 한국정보과학회 2007 가을 학술발표 논문집 제 34 권, 제 2 호, Page(s): 151~154, 2007
[3] Zhexue Huang, “Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values,” Data Mining and Knowledge Discovery 2, Page(s): 283~304, 1998
[4] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, “Introduction to Data Mining,” Addison Wesley, 2005
[5] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schroedl, “Constrained K-means Clustering with Background Knowledge,” Proceedings of the Eighteenth International Conference on Machine Learning, Page(s): 577~584,2001
[6] Yuhui Shi, Russell C.Eberhart, “Empirical Study of Particle Swarm Optimization,” Proceedings of the 1999 Congress on Evolutionary Computation, Page(s):1945~1950, 1999
[7] DW van der Merwe, AP Engelbrecht, “Data Clustering using particle swarm optimization,” Evolutionary Computation, Page(s):215~220, 2003