MGKA: A genetic algorithm-based clustering technique for genomic data

Hung Nguyen
University of Nevada
Reno, NV, USA
hungnp@nevada.unr.edu

Sushil J. Louis
University of Nevada
Reno, NV, USA
sushil@cse.unr.edu

Tin Nguyen
University of Nevada
Reno, NV, USA
tinn@unr.edu

***Abstract*—Advances in high-throughput technologies have generated enormous amounts of high-throughput genomic data. Cluster analysis is often the first step to gain insights into genomic data. K-means, the most widely used clustering algorithm, is known to produce sub-optimal clusters depending on the choice of initialized centers. In this paper, we propose a genetic algorithm-based unsupervised clustering method that searches for the optimal centers of clusters based on the concept of k-means. The genetic algorithm reduces k-means sensitivity to randomly initialized centers and reduces the probability of converging to local minima. Two clustering validity indexes are introduced to the selection process to automatically determine the appropriate number of clusters. The proposed algorithm is applied to 16 disease datasets and four single-cell datasets to demonstrate its performance. Results show that our approach outperforms the current state of the art algorithms on a majority of the datasets.**

***Index Terms*—genetic algorithm, clustering, gene expression, single cell, disease subtyping**

I. Introduction

Cluster analysis has become a widely used tool for the exploration of high-dimensional data. Cluster analysis is an unsupervised approach to categorize objects without any predefined standards or knowledge for classification. In general, clustering methods aim to recognize the differences and similarities between objects so that the most similar objects will be grouped into one cluster and vice versa. Advances in high-throughput technologies, which produce a huge amount of genomic information, put a high demand on clustering methods that analyze gene expression data with disease subtypes and cell types discovery, two of the main application areas for clustering.

Due to the noisy nature of genomic data and its undefined structures, it is impossible to find a universal clustering approach that works efficiently on these data. Along with classical clustering methods such as k-means [1], partition around medoids [2] and hierarchical clustering, many other modern techniques have been developed recently to tackle the clustering problems of genomic data [3]. The k-means clustering method, which is a broadly used and well-considered clustering technique, was found to be efficient for clustering cancer datasets [3]. The k-means clustering technique is simple to use and easy to implement and one of the most straightforward algorithms to understand. With a predefined number of clusters k, the algorithm tries to find k centroids in the multiple-dimensional space from a set of random centers so that every data point is allocated to an adjacent centroid. A detailed discussion about the algorithm can be found in [4].

However, the k-means algorithm is known to be sensitive to initial conditions and does not guarantee to produce global optimal clusters. The clustering results heavily depend on the starting center points which are (usually) randomly initialized. Therefore, the algorithm is susceptible to converge into to a local optimum. Furthermore, the number of clusters must be given as an input parameter for the k-means clustering technique. Without any prior knowledge of the data, determining the appropriate number of clusters is considered a difficult task.

A few efforts have been accounted for to take care of the clustering initialization problem. The most common and naive technique is to attempt the k-means algorithm multiple times with different initial seeds and gather the best result. However, the best-obtained solution from this stochastic procedure does not often produce globally optimal clusters. Note that finding globally optimal clusters is known to be an NP-hard problem. Several techniques have been introduced to refine the starting points for the k-means clustering method [5]–[9]. On the other hand, many other studies have tried to combine k-means with other heuristic algorithms to prevent k-means from converging into local minima including simulated annealing [10], [11] and genetic algorithm [12].

The genetic algorithm (GA) is a powerful technique for optimization problems based on natural selection and genetics. GAs have been applied to many function optimization problems and have been shown to be good at finding optimal and near-optimal solutions. The basic methods of the genetic algorithm are designed to reproduce processes in normal systems necessary for evolution based on the principle of survival of the fittest. Although the initial population is randomized, the GA is by no means random. It chronologically directs the population in the search space by probabilistic applying genetic operators including selection, crossover, and mutation. In general, the selection operator selects individuals from the current population for the next population with a probability proportional to the individual's fitness relative to the fitness of the rest of the population. Crossover operates on two individuals (parents) to produce two new individuals (offspring) inheriting some of the attributes from their parents.

978-1-7281-2153-6/19/$31.00 ©2019 IEEE

The mutation operator changes the genomic structure of an individual at some point in hope that the mutated individual will help maintain diversity for crossover and selection to exploit. Depending on the specificity of the problem, the selection, the crossover, and the mutation procedures in the GA may vary.

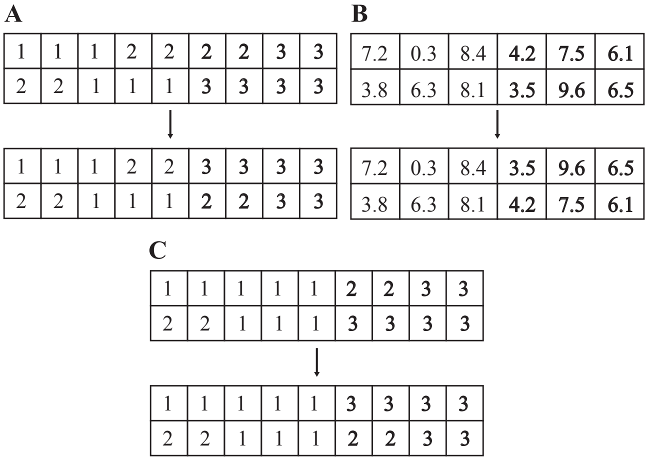
There have been many studies attempt to apply GA to refine the k-means algorithm [13]–[18]. However, many of them omit the crossover procedure and greatly depend on the selection and mutation operator. On the other hand, none of those methods has taken into account the problem of choosing the appropriate number of cluster for the k-means algorithm. In this paper, we proposed a Multi-objective Genetic algorithm-based K-means Algorithm (MGKA) to refine the k-means clustering algorithm and to automatically determine a suitable number of clusters. The algorithm presents each individual as a set of centroids for a solution. It at the same time holds a population of individuals that encode for different numbers of clusters. MGKA will concurrently optimize the k-means objective function and also a clustering validity index to evaluate the fitness of an individual. The performance of the proposed algorithm is evaluated by comparing with naive k-means on simulated datasets. With real datasets, we compare our method with five other methods that were particularly developed for clustering genomic data. We compare our algorithm with three other well-known methods developed for disease subtyping including Consensus Clustering [19], Similarity Network Fusion [20] and iClusterPlus [21] on 16 disease datasets. Lastly, we evaluate our algorithm on four single-cell datasets and compare it with two methods developed for single-cell clustering including SC3 [22] and SEURAT [23] methods.

II. Related work on genetic clustering algorithm

Several studies address genetic algorithm to solve clustering problems using label-based representation for solution [13], [15], [18], [24]. Label-based representation uses integer encoding to present cluster membership. For example, providing k number of clusters (e.g. $k = 3$), the integer vector [111222233] indicates that the first three data points belong to the cluster #1, the next four data points belong to cluster #2, and the last two data points belong to cluster #3. This encoding is however redundant. For example, the cluster membership integer vector [111222233] is equivalent to [222111133]. With the same solution, there will be $k!$ different encodings. Therefore, the size of the search space for genetic algorithm significantly increases when the number of clusters k increases, which may reduce the efficiency of the genetic algorithm.

Comparing to label-based representation, medoid-based and centroid-based representations, which encode only the centers of the clusters, are more efficient in terms of the size of the search space. However, the ultimate benefits of each representation are still hard to evaluate and compare because performance also greatly depends on the design of the fitness function. Several methods make use of medoid-based representation using integer encoding to encode the solution [25], [26]. The previous cluster membership example [111222233] can be encoded as [2 5 8] in k-medoids approach in which the second, fifth, and eighth data points are three centers represented for three clusters. Other data points are then assigned to each cluster using these centers. Centroid-based representation, on the other hand, uses real-number encoding to represent the center of clusters. Unlike medoid-based representation which uses data points in the input data as cluster centers, cluster centers in centroid-based representation can be any point in the multi-dimensions space. Therefore, a solution now is represented by a set of coordinates. For example, the real-number vector [7.2 0.3 8.4 4.2 7.5 6.1] illustrates three cluster centers $A(7.2, 0.3)$, $B(8.4, 4.2)$, and $C(7.5, 6.1)$. This representation is adopted by Maulik and Bandyopadhyay [27] and several other papers [28]–[30].

Traditional genetic crossover is strongly adopted in genetic-based clustering algorithms. Many studies applied one-point crossover to produce offspring for both integer encoding solutions and real-number encoding solutions [24], [27]–[29]. Figure 1 describes one point crossover for integer encoding (A) and real-number encoding (B). However, the naive one-point crossover can produce invalid offspring as described in Figure 1C. On the other hand, one-point crossover on a real-number encoding can be very destructive to the population since it can generate significantly different offspring compared to its parents. In high dimension data, this operation tends to swap the centers between parents rather than moving them in the high-dimensional space. Crossover can also be omitted, such as in Krishna and Murty's method [13].



A

1	1	1	2	2	**2**	**2**	**3**	**3**
2	2	1	1	1	**3**	**3**	**3**	**3**

↓

1	1	1	2	2	**3**	**3**	**3**	**3**
2	2	1	1	1	**2**	**2**	**3**	**3**

B

7.2	0.3	8.4	**4.2**	**7.5**	**6.1**
3.8	6.3	8.1	**3.5**	**9.6**	**6.5**

↓

7.2	0.3	8.4	**3.5**	**9.6**	**6.5**
3.8	6.3	8.1	**4.2**	**7.5**	**6.1**

C

1	1	1	1	1	**2**	**2**	**3**	**3**
2	2	1	1	1	**3**	**3**	**3**	**3**

↓

1	1	1	1	1	**3**	**3**	**3**	**3**
2	2	1	1	1	**2**	**2**	**3**	**3**

Fig. 1. One point crossover for (A) integer encoding and (B) real-number encoding. (C) Invalid offspring from one point crossover for integer encoding.

With label-based representation, mutation can be as simple as assigning a data point to a random cluster. However, it can generate invalid solutions. Krishna and Murty [13] design a mutation operator that assigns a new cluster for a data point based on the distances of the cluster centroids from the corresponding data point. The cluster that has the centroid closer to the data point will have a higher probability of being assigned to that data point. This mutation principle is also adopted by Lu et al. [15], [18]; however, such methods have been found to create empty clusters. Other papers using real-number encoding [27]–[29] operate mutation by slightly modifying the centroids. By modifying the centroids, this mutation may change the membership of some data points in relation to the clusters represented by the solution. It can also shake the centers out of the local optimum.

K-means is also used in several methods to refine the genetic algorithm generated results [18], [27], [29]. With each generation, one or multiple steps of k-means are applied to certain solutions during the mutation process or to all individuals in the population. This operation is especially helpful in urging the genetic algorithm to converge and in refining the ultimate solutions. However, it can also trap solutions at local optima.

In this paper, we make use of real-number encoding to encode the cluster centers in a way such that the number of clusters encoded by a solution is dynamic. We use simulated binary crossover [31] which applies crossover for every dimension of the centers in the solution. This crossover operation will generate offspring close to their parents. Besides adding noise to the cluster centers to avoid the convergence of the genetic algorithm to local optimum, our mutation operation can also change the number of clusters that a solution represents. We also make use of the k-means operator to refine the solutions. The details of each operation will be discussed in the next section.

III. Multi-objective genetic k-means clustering algorithm

We describe a new multi-objective genetic k-means clustering algorithm using real-number center-based encoding to present solutions with a dynamic number of clusters. Each solution (chromosome) has two encoded regions as shown in Figure 2. The first region encodes the status of each center, which is either active (1) or disabled (0). The second region encodes the coordinates of each center. Figure 2 represents a solution for two-dimensional input data where the number of clusters is two. In this example, the maximum number of clusters is three. However, this number can be higher. The number of coordinates for each center depends on the number of dimensions in the input data.

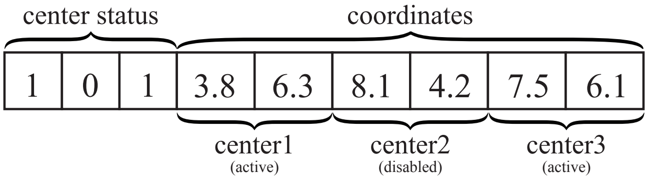


Fig. 2. Chromosome encoding of a two-cluster solution for two-dimension data with the maximum cluster it can encode is three.

This chromosome encoding allows us to hold solutions with different numbers of clusters in the same population. It also allows us to change the number of cluster of a solution through mutation, which prevents solutions with the same number of clusters from dominating the population.

The population is randomly initialized by selecting random data points and assigning their coordinates to cluster centers. With predefined maximum cluster numbers $kMax$ from users, each number of clusters k is initialized with the same number of solutions.

A. The fitness functions

The fitness of individuals is evaluated using three different criteria including: i) within-cluster sum of squares, ii) Davies and Bouldin index [32], and iii) Silhouette index [33]. We describe each of these in turn below.

Within cluster sum of squares (WCSS) is the objective function of the original k-means. Denoting k as the number of clusters, $\{c_i, i \in [1..k]\}$ as the cluster centers, and $\{C_i, i \in [1..k]\}$ as the k clusters (each cluster consists of many data points), the within-cluster sum of squares is defined as:

$$WCSS = \sum_{i=1}^{k} \sum_{j \in C_i} \|x_j - c_i\|^2$$

where $\|x_j - c_i\|^2$ is the Euclidean squared distance between data point x_j and center c_i. A better solution will have a smaller $WCSS$ value.

Davies and Bouldin (DB) index is a function of the sum of within-cluster scatter to between-cluster separation. A better solution will have a smaller $DB(k)$ value. DB index is calculated as follows:

$$DB(k) = \frac{1}{k} \sum_{i=1}^{k} \underset{i \neq j}{max} (\frac{\delta_i + \delta_j}{d_{ij}})$$

where

- k is the number of clusters,
- i, j are the i^{th} and j^{th} cluster respectively,
- d_{ij} is the distance between centers c_i and c_j,
- δ_i and δ_j are the dispersion measure of a cluster C_i and C_j, respectively. For example, C_i the standard deviation of the distance of data points in cluster C_i to the center of this cluster c_i.

Silhouette index (SI) measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). A better solution will have higher SI value. The silhouette index is computed as follows:

$$SI = \frac{\sum_{i=1}^{n} \frac{b(i) - a(i)}{max\{a(i); b(i)\}}}{n}$$

where

- $a(i) = \frac{\sum_{j \in \{C_r \setminus i\}} d_{ij}}{n_r - 1}$ is the average dissimilarity of the i^{th} object to all other objects of cluster C_r,
- $b(i) = \underset{s \neq r}{min}\{d_{iC_s}\}$, in which $d_{iC_s} = \frac{\sum j \in C_s d_{ij}}{n_s}$ is the average dissimilarity of the i^{th} object to all objects of cluster C_s.

By using all of the three metrics (WCSS, DB, and SI), the fitness function will evaluate how similar each member is in the same cluster and how well the clusters are separated.

B. The crossover operator

The crossover operator is performed by using the simulated binary crossover proposed by Agrawal, R. B. et al. [31] on parents that have the same number of clusters. The crossover operator is performed by using the simulated binary crossover proposed by Agrawal et al. [31]. The crossover procedure is described as in Figure 3. For each pair of parents ($Parent_1$ and $Parent_2$) that have the same number of clusters selected randomly from the population, the simulated crossover is applied to each coordinate of the centers in $Parent_1$ with the corresponding coordinate of the centers in $Parent_2$.

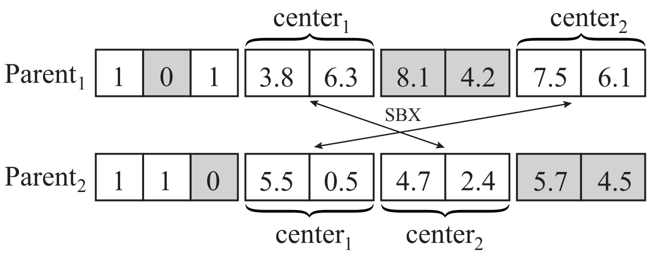


Fig. 3. Crossover procedure between two parents. Simulated binary crossover is applied to (1) each coordinate of $center_1$ of $Parent_1$ with corresponding coordinate of $center_2$ of $Parent_2$, and (2) each coordinate of $center_2$ of $Parent_1$ with corresponding coordinate of $center_1$ of $Parent_2$.

First, the Euclidean distance is calculated between any centers of $Parent_1$ and $Parent_2$. Simulated binary crossover is then applied to each coordinate of the corresponding dimension of the closest centers between two parents. These centers are then removed from the crossover center list. The procedure is applied to the rest of the centers of the two parents until no center is left. The results of another example of the crossover operator can be seen in Figure 4.

C. The mutation operator

Mutation shakes the centers out of a local optimum and moves them, hopefully, towards the global optimum. Within the mutation operator, there are two functions that can be applied to each solution including (1) adding noise to the centers of each cluster and (2) changing the number of clusters.

Noise is added to the centers of each cluster using Gaussian noise. The noise added to the data will have the variance equal to the variance of the data. By setting the variance of the added noise equal to the median variance of the data, we aim to sufficiently shake the centers out of local optima. If the added noise is considerably higher, the new centers will be moved further from the original points, which can destroy the solution. On the other hand, if the noise is low, the new centers will only move close to the original centers which can result in being trapped in a local optimum.

The mutation operator can also change the number of clusters and solutions by activating or disabling a center. Activating a center will select a random data point and add its coordinates to a new center. Disabling a center will select a random center in the solution and mark it as disabled.

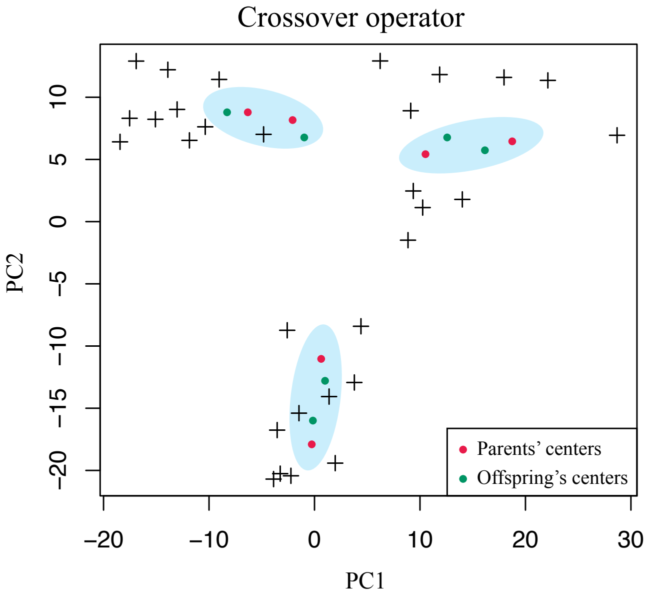


Fig. 4. Offspring resulted from simulated binary crossover. Red dots represent centers of two parents and green dots represent centers of two corresponding offspring

.

D. The k-means operator

The k-means operator is applied to speed up the convergence of the algorithm by applying one step of the k-means algorithm to the solution. For each generation, the k-means operator is applied with a predefined probability by the users for each solution. The procedure starts with assigning each data points to the closest center in the solution, the centers are then adjusted using the mean of the data points assigned to that center.

The k-means operator, however, can produce an illegal solution with empty clusters. If the adjusted solution contains empty clusters, a random data point will replace the center having empty members. The k-means operator is then re-applied until a valid solution is produced.

E. The selection operator

The goal of the selection operator is to find the Pareto front of the three objective functions. In this paper, we make use of the selection procedure proposed by Deb, K. et al. [34]: Non-dominated Sorting Genetic Algorithm (NSGA-II). The principle of the selection is to arrange the population into a hierarchy of non-dominated Pareto fronts and use a crowding distance to prevent solutions from concentrating in the region at the level of the Pareto fronts (Figure 5). The detail implementation of the algorithm is described in [34].

F. Evaluating the ultimate solution

Although NSGAII was designed to produce a dispersed pareto front and in practice we can present the entire pareto front to a domain expert user to choose from, we may still wish to identify an "ultimate" solution as the result of our algorithm. We start by considering all individuals in the final Pareto front.

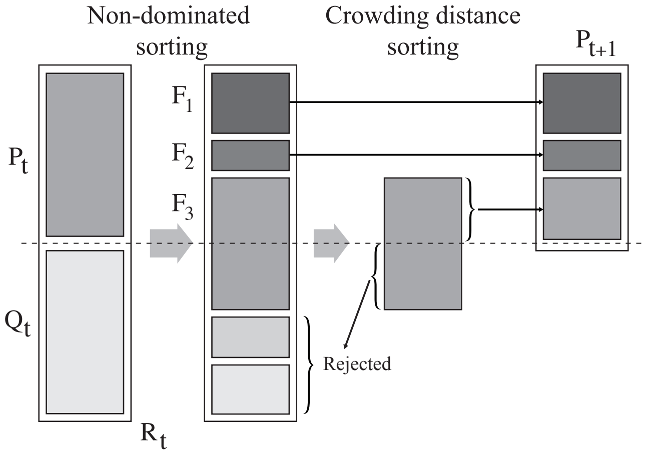


Fig. 5. Non-dominated Sorting Genetic Algorithm (NSGA-II) where t is the population generation, P is the parents, Q is the offspring, and F_i is the Pareto front level i^{th}.

We then select the best solution along each objective. If the best solutions for two index value are different, each solution will be ranked based on its other index value compared to other solutions in our pareto set. The solution that has better rank will be extracted as the ultimate solution.

IV. Experimental results

A. Results on simulation

We first validate the framework from a theoretical perspective by comparing the new method with the original k-means. In this section, we compare the performance of MGKA with k-means on generated datasets with a large number of clusters. It is known that k-means does not produce a global optimum. Therefore, we run k-means multiple times in order to obtain results that are at least close to global optimum. Here we set the number of times we run k-means equal to the population size of MGKA, which is 50. The simulation generates datasets with the number of clusters from 10 to 15; each cluster is well separated and has 10 members. The landscape of the simulated data with $k = 10$ is described as in Figure 6. We use the $kmeans$ function in $stats$ package, R programming language to obtain the clustering result from k-means algorithm.

The average result of 30 runs for each k is represented in Table I. We use the within-cluster sum of square errors and Adjusted Rand Index (ARI) to compare the result between two algorithms. Table I shows that MGKA outperforms k-means in all of the datasets. Adjust Rand Index (ARI) values for clusters produced by MGKA in all datasets show that MGKA can easily achieve the global optima in all simulated datasets. K-means, on the other hand, produces sub-optimal solutions most of the times. The average ARI of 30 runs also shows that MGKA is much more stable compared to k-means. The within-cluster sum of squares shows significant differences among clusters produced by MKGA and k-means. The results from k-means are also too far away from optimal solutions.

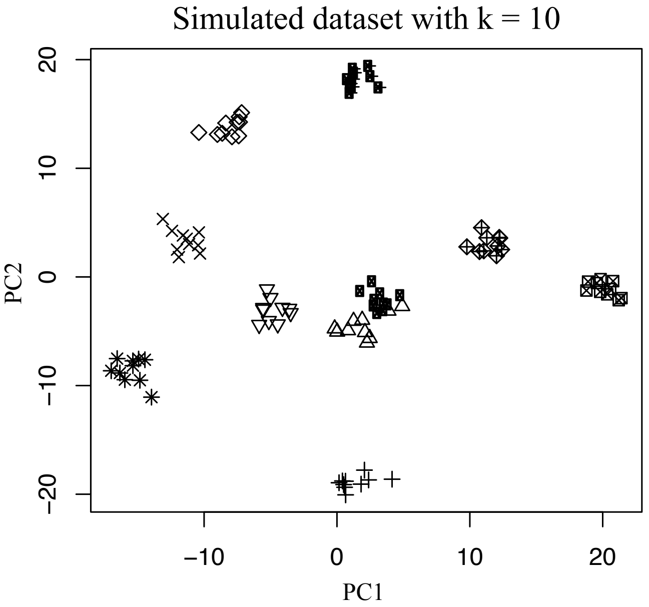


Fig. 6. The landscape of simulated dataset with the number of cluster is ten. Each cluster is well separated to each other and has ten members.

TABLE I
Within cluster sum of square errors and Adjust Random Index (ARI) of clustering result produced by MGKA and k-means with restarts.

#k	#Samples	WithinSS		ARI	
		MGKA	k-means	MGKA	k-means
10	100	457.237	782.051	1	0.963
11	110	461.326	996.554	1	0.954
12	120	520.686	913.989	1	0.939
13	130	598.247	910.19	0.993	0.914
14	140	547.731	1136.477	1	0.931
15	150	630.188	1074.967	1	0.929

B. Results on cancer omics data

Here we demonstrate the application MGKA in the context of cancer subtyping using multi-omics data. In order to assess the performance of MGKA, we compare the results of MGKA with those of wide used methods in this field, including Consensus Clustering (CC) [19] – a resampling-based approach, Similarity Network Fusion (SNF) [20] – a graph-theoretical approach, and iClusterPlus [21] – a mixture model approach. We CC, SNF, and iClusterPlus, we use the default parameter settings. The parameters for MGKA after this section are: population size = 20, the number of generations = 20, crossover probability = 1, mutation probability = 0.01, and k-means operator probability = 0.5.

First, we compare the four methods using eight mRNA gene expression datasets with known disease subtypes. The 5 datasets with accession id GSE10245, GSE19188, GSE43580, GSE15061, and GSE14924 were downloaded from Gene Expression Omnibus (www.ncbi.nlm.nih.gov/geo/). The other three datasets were downloaded from

TABLE II
DESCRIPTION OF THE EIGHT MRNA DATASETS USED IN OUR ANALYSIS. THE TOP FIVE DATASETS WERE DOWNLOADED FROM THE GENE EXPRESSION OMNIBUS. THE BOTTOM THREE DATASETS WERE DOWNLOADED FROM THE BROAD INSTITUTE WEBSITE.

Datasets	#Class	#Sample	#Feature	Platform	Description
GSE10245 [35]	2	58	19851	hgu133plus2	40 adenocarcinomas and 18 squamous cell carcinomas
GSE19188 [36]	3	91	19851	hgu133plus2	45 adenocarcinomas, 19 large cell carcinomas, and 27 squamous cell carcinomas
GSE43580 [37]	2	150	19851	hgu133plus2	77 adenocarcinomas and 73 squamous cell carcinomas
GSE14924 [38]	2	20	19851	hgu133plus2	10 acute myeloid leukemia CD4 T cell and 10 CD8 T cell
GSE15061 [39]	2	366	19851	hgu133plus2	202 acute myeloid leukemia samples and 164 myelodyplastic syndrome samples
Lung2001 [40]	4	237	8641	hgu95a	190 adenocarcinomas, 21 squamous cell carcinomas, 20 carcinoid, and 6 small-cell lung carcinomas
AML2004 [41], [42]	3	38	5000	hgu6800	11 acute myeloid leukemia, 19 acute lymphoblastic leukemia B cell, and 8 T cell
Brain2002 [43]	5	42	5299	hgu6800	10 meduloblastomas, 10 malignant gliolas, 10 atypical teratoid/rhaboid tumors, 4 normal cerebellums, and 8 primitive neuroectodermal tumors

the Broad Institute: Lung2001 (www.broadinstitute.org/mpr/lung/), AML2004 (www.broadinstitute.org/cancer/pub/nmf), and Brain2002 (www.broadinstitute.org/MPR/CNS/). Details of the 8 datasets are described in Table II. The results of clustering for eight mRNA datasets are represented in Table III. We use the Adjusted Rand Index (ARI) to assess the performance of the resulted subtypes. Among the eight datasets that we tested, MGKA outperforms other methods in six methods. SNF and iClusterPlus however crashed with GSE14924 and AML2004 and are represented with *NA* in the table.

TABLE III
THE PERFORMANCE OF MGKA, CONSENSUS CLUSTERING (CC), SIMILARITY NETWORK FUSION (SNF), AND ICLUSTERPLUS IN DISCOVERING SUBTYPES FROM GENE EXPRESSION DATA. FOR EACH DATASET (ROW), CELLS HIGHLIGHTED IN GREEN HAVE THE HIGHEST ADJUSTED RAND INDEX (ARI).

Dataset	Samples	#Class	MGKA	CC	SNF	iCluster+
GSE10245	58	2	0.80	0.32	0.38	0.22
GSE19188	91	3	0.84	0.6	0.12	0.19
GSE43580	150	2	0.44	0.37	0.15	0.21
GSE15061	366	2	0.78	0.43	0.05	0.15
GSE14924	20	2	1.00	0.25	NA	0.73
Lung2001	237	4	0.54	0.11	0.28	0.11
AML2004	38	3	0.41	0.56	0.17	NA
Brain2002	42	5	0.15	0.46	0.13	0.32

Secondly, we compare the four methods using DNA methylation datasets from The Cancer Genome Atlas (TCGA). In the comparison, we use eight datasets downloaded from the TCGA website (cancergenome.nih.gov and firebrowse.org). Eight datasets include Glioblastoma multiforme (GBM), Thymoma (THYM), Glioma (GBMLGG), Kidney renal papillary cell carcinoma (KIRP), Kidney Chromophobe (KICH), Uveal Melanoma (UVM), Pancreatic adenocarcinoma (PAAD), and Adrenocortical carcinoma (ACC). These datasets, however, do not contain subtypes for each disease. Instead, with known survival outcome, we use Cox regression to assess the survival difference of the discovered subtypes. The Cox p-values of the subtypes discovered by each of the four approaches are presented in table IV. Again, among eight datasets, MGKA outperforms other methods in six datasets. Moreover, while MGKA can discover subtypes with significant cox-p value (at the threshold of 5%) for all datasets, CC, SNF, and iClusterPlus can only discover subtypes with significant cox-p value for three, seven, and five datasets respectively.

TABLE IV
THE PERFORMANCE OF MGKA, CONSENSUS CLUSTERING (CC), SIMILARITY NETWORK FUSION (SNF), AND ICLUSTERPLUS IN DISCOVERING SUBTYPES FROM DNA METHYLATION DATA. CELLS HIGHLIGHTED IN YELLOW HAVE SIGNIFICANT COX P-VALUES AT THE THRESHOLD OF 5%. FOR EACH DATASET (ROW), CELLS HIGHLIGHTED IN GREEN HAVE THE MOST SIGNIFICANT COX P-VALUE.

Dataset	Samples	MGKA	CC	SNF	iCluster+
GBM	273	1.2e−4	0.075	0.017	0.103
THYM	119	0.006	0.053	0.04	0.068
GBMLGG	510	3.3e−16	3e−9	1.9e−12	5.4e−14
KIRP	271	5.1e−18	0.299	2.8e−13	0.013
KICH	65	1e−4	0.88	1e−4	0.788
UVM	80	7.1e−4	9.8e−4	0.005	0.003
PAAD	178	0.002	6.6e−4	0.346	3.8e−4
ACC	79	6.2e−4	0.06	0.047	6.6e−5

C. *Results on single-cell transcriptomics data*

We also test our method on four different single-cell datasets with known cell types (Table V). Yan's dataset contains 90 human embryo samples in six different stages. Goolam's, and Deng's datasets contain mouse embryo samples in different stages. Pollen's dataset contains 301 samples of different human tissues. The references for each dataset are given in Table V. We compare our method with SC3 [22] method - a consensus clustering method of single-cell RNA-seq data, and SEURAT [23] - a graph-based clustering approach for single-cell RNA-seq data. Table V shows the ARI values obtained by MGKA, SC3, and SEURAT on those four datasets. MGKA produces the best clusters in three out of four tested datasets.

V. CONCLUSION AND FUTURE WORK

K-means clustering is a simple, fast and unsupervised approach. However, it suffers from some limitations such as the initial centroids problem and the selection of the appropriate number of clusters. This paper describes and evaluates

TABLE V

THE PERFORMANCE OF MGKA, SC3, AND SEURAT IN DISCOVERING CELL TYPES FROM GENE EXPRESSION DATA. FOR EACH DATASET (ROW), CELLS HIGHLIGHTED IN GREEN HAVE THE HIGHEST ADJUSTED RAND INDEX (ARI). MGKA PRODUCES CLUSTERS WITH HIGHEST ARI VALUE FOR THREE OUT OF FOUR DATASETS.

Dataset	Samples	#Class	MGKA	SC3	SEURAT
Yan (GSE36552) [44]	90	6	0.67	0.63	0.53
Goolam (E-MTAB-3321) [45]	124	5	0.72	0.63	0.57
Deng (GSE45719) [46]	268	6	0.60	0.55	0.51
Pollen (SRP041736) [47]	301	11	0.88	0.93	0.70

a new approach that uses an evolutionary multi-objective algorithm to find a set of pareto optimal solutions along three measures of cluster goodness. A new representation directly addresses the initial centroid problem and the non-dominated sorting genetic algorithm maintains a population with a diverse number of high performing clusters. That is, while many current approaches integrate genetic algorithm with k-means to find the global optimum for a fixed number of clusters, our method, MGKA, is able to maintain and evaluate solutions with different numbers of clusters at the same time. By using simulated binary crossover, our crossover operator is less destructive compared to naive one-point crossover and generates offspring close to the parents rather than exchanging dataset members or center coordinates.

The multi-objective genetic algorithm allows us to optimize the solution with different cluster validity index so that at the same time, we can also evaluate the appropriate number of clusters. By using Davies & Bouldin index and Silhouette index, the best solutions will have the most similar members in the same cluster and have well separated clusters. Our experiment on different simulated datasets shows that MGKA is better than naive k-means in finding the global optimum. Other experiments on 16 disease datasets and five single-cell datasets indicate that MGKA outperforms other state-of-the-art algorithms discovering disease subtypes and cell types. This provides strong evidence of the viability of our approach for clustering applications especially in the biomedical domain.

REFERENCES

[1] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, Oakland, CA, USA. Berkeley, USA: University of California Press, 1967, pp. 281–297.

[2] L. Kaufman and P. J. Rousseeuw, "Partitioning around medoids (program pam)," *Finding groups in data: an introduction to cluster analysis*, pp. 68–125, 1990.

[3] J. Oyelade, I. Isewon, F. Oladipupo, O. Aromolaran, E. Uwoghiren, F. Ameh, M. Achas, and E. Adebiyi, "Clustering algorithms: Their application to gene expression data," *Bioinformatics and Biology insights*, vol. 10, pp. BBI–S38 316, 2016.

[4] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[5] P. S. Bradley and U. M. Fayyad, "Refining initial points for k-means clustering." in *ICML*, vol. 98. Citeseer, 1998, pp. 91–99.

[6] S. J. Redmond and C. Heneghan, "A method for initialising the k-means clustering algorithm using kd-trees," *Pattern Recognition Letters*, vol. 28, no. 8, pp. 965–973, 2007.

[7] M. Laszlo and S. Mukherjee, "A genetic algorithm that exchanges neighboring centers for k-means clustering," *Pattern Recognition Letters*, vol. 28, no. 16, pp. 2359–2366, 2007.

[8] J.-F. Lu, J. Tang, Z.-M. Tang, and J.-Y. Yang, "Hierarchical initialization approach for k-means clustering," *Pattern Recognition Letters*, vol. 29, no. 6, pp. 787–795, 2008.

[9] X. Qin and S. Zheng, "A new method for initialising the k-means clustering algorithm," in *2009 2nd International Symposium on Knowledge Acquisition and Modeling, KAM 2009*, vol. 2. IEEE, 2009, pp. 41–44.

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[11] S. Bandyopadhyay, U. Maulik, and M. K. Pakhira, "Clustering using simulated annealing with probabilistic redistribution," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 02, pp. 269–285, 2001.

[12] J. Holland, "Adaptation in natural and artificial systems: an introductory analysis with application to biology," *Control and artificial intelligence*, 1975.

[13] K. Krishna and M. N. Murty, "Genetic k-means algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 3, pp. 433–439, 1999.

[14] M. Anusha and J. Sathiaseelan, "An enhanced k-means genetic algorithms for optimal clustering," in *2014 IEEE International Conference on Computational Intelligence and Computing Research, IEEE ICCIC 2014*. IEEE, 2014, pp. 1–5.

[15] Y. Lu, S. Lu, F. Fotouhi, Y. Deng, and S. J. Brown, "Fgka: A fast genetic k-means clustering algorithm," in *Proceedings of the 2004 ACM symposium on Applied computing*. ACM, 2004, pp. 622–623.

[16] D. K. Roy and L. K. Sharma, "Genetic k-means clustering algorithm for mixed numeric and categorical data sets," *International Journal of Artificial Intelligence & Applications*, vol. 1, no. 2, pp. 23–28, 2010.

[17] Z. Feng, "Data clustering using genetic algorithms," *Evolutionary Computation: Project Report, CSE484*, 2012.

[18] Y. Lu, S. Lu, F. Fotouhi, Y. Deng, and S. J. Brown, "Incremental genetic k-means algorithm and its application in gene expression data analysis," *BMC bioinformatics*, vol. 5, no. 1, p. 172, 2004.

[19] S. Monti, P. Tamayo, J. Mesirov, and T. Golub, "Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data," *Machine Learning*, vol. 52, no. 1-2, pp. 91–118, 2003.

[20] B. Wang, A. M. Mezlini, F. Demir, M. Fiume, Z. Tu, M. Brudno, B. Haibe-Kains, and A. Goldenberg, "Similarity network fusion for aggregating data types on a genomic scale," *Nature Methods*, vol. 11, no. 3, p. 333, 2014.

[21] Q. Mo, S. Wang, V. E. Seshan, A. B. Olshen, N. Schultz, C. Sander, R. S. Powers, M. Ladanyi, and R. Shen, "Pattern discovery and cancer gene identification in integrated cancer genomic data," *Proceedings of the National Academy of Sciences*, p. 201208949, 2013.

[22] V. Y. Kiselev, K. Kirschner, M. T. Schaub, T. Andrews, A. Yiu, T. Chandra, K. N. Natarajan, W. Reik, M. Barahona, A. R. Green *et al.*, "Sc3: Consensus clustering of single-cell rna-seq data," *Nature methods*, vol. 14, no. 5, p. 483, 2017.

[23] A. Butler, P. Hoffman, P. Smibert, E. Papalexi, and R. Satija, "Integrating single-cell transcriptomic data across different conditions, technologies, and species," *Nature biotechnology*, vol. 36, no. 5, p. 411, 2018.

[24] R. Krovi, "Genetic algorithms for clustering: a preliminary investigation," in *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, vol. 4. IEEE, 1992, pp. 540–544.

[25] C. B. Lucasius, A. D. Dane, and G. Kateman, "On k-medoid clustering of large data sets with the aid of a genetic algorithm: background, feasiblity and comparison," *Analytica Chimica Acta*, vol. 282, no. 3, pp. 647–669, 1993.

[26] W. Sheng and X. Liu, "A hybrid algorithm for k-medoid clustering of large data sets," in *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)*, vol. 1. IEEE, 2004, pp. 77–82.

[27] S. Bandyopadhyay and U. Maulik, "An evolutionary technique based on k-means algorithm for optimal clustering in rn," *Information Sciences*, vol. 146, no. 1-4, pp. 221–237, 2002.

[28] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern recognition*, vol. 33, no. 9, pp. 1455–1465, 2000.

[29] P. Scheunders, "A genetic c-means clustering algorithm applied to color image quantization," *Pattern recognition*, vol. 30, no. 6, pp. 859–866, 1997.

[30] J. Kivijärvi, P. Fränti, and O. Nevalainen, "Self-adaptive genetic algorithm for clustering," *Journal of Heuristics*, vol. 9, no. 2, pp. 113–129, 2003.

[31] R. B. Agrawal, K. Deb, and R. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.

[32] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224–227, 1979.

[33] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[34] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[35] R. Kuner, T. Muley, M. Meister, M. Ruschhaupt, A. Buness, E. C. Xu, P. Schnabel, A. Warth, A. Poustka, H. Sultmann, and H. Hoffmann, "Global gene expression analysis reveals specific patterns of cell junctions in non-small cell lung cancer subtypes," *Lung Cancer*, vol. 63, no. 1, pp. 32–38, 2009.

[36] J. Hou, J. Aerts, B. Den Hamer, W. Van Ijcken, M. Den Bakker, P. Riegman, C. van der Leest, P. van der Spek, J. A. Foekens, H. C. Hoogsteden, F. Grosveld, and S. Philipsen, "Gene expression-based classification of non-small cell lung carcinomas and survival prediction," *PLoS ONE*, vol. 5, no. 4, p. e10312, 2010.

[37] A. L. Tarca, M. Lauria, M. Unger, E. Bilal, S. Boue, K. K. Dey, J. Hoeng, H. Koeppl, F. Martin, P. Meyer, P. Nandy, R. Norel, M. Peitsch, J. J. Rice, R. Romero, G. Stolovitzky, M. Talikka, Y. Xiang, C. Zechner, and IMPROVER DSC Collaborators, "Strengths and limitations of microarray-based phenotype prediction: lessons learned from the IMPROVER diagnostic signature challenge," *Bioinformatics*, vol. 29, no. 22, pp. 2892–2899, 2013.

[38] R. Le Dieu, D. C. Taussig, A. G. Ramsay, R. Mitter, F. Miraki-Moud, R. Fatah, A. M. Lee, T. A. Lister, and J. G. Gribben, "Peripheral blood T cells in acute myeloid leukemia (AML) patients at diagnosis have abnormal phenotype and genotype and form defective immune synapses with AML blasts," *Blood*, vol. 114, no. 18, pp. 3909–3916, Oct. 2009.

[39] K. I. Mills, A. Kohlmann, P. M. Williams, L. Wieczorek, W.-m. Liu, R. Li, W. Wei, D. T. Bowen, H. Loeffler, J. M. Hernandez, W.-K. Hofmann, and T. Haferlach, "Microarray-based classifiers and prognosis models identify subgroups with distinct clinical outcomes and high risk of AML transformation of myelodysplastic syndrome," *Blood*, vol. 114, no. 5, pp. 1063–1072, 2009.

[40] A. Bhattacharjee, W. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. Mark, E. Lander, W. Wong, B. Johnson, T. Golub, D. Sugarbaker, and M. Meyerson, "Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses," *Proceedings of the National Academy of Sciences*, vol. 98, no. 24, pp. 13 790–5, Nov. 2001.

[41] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[42] J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov, "Metagenes and molecular pattern discovery using matrix factorization," *Proceedings of the National Academy of Sciences*, vol. 101, no. 12, pp. 4164–4169, Mar. 2004.

[43] S. Pomeroy, P. Tamayo, M. Gaasenbeek, L. Sturla, M. Angelo, M. McLaughlin, J. Kim, L. Goumnerova, P. Black, C. Lau, J. Allen, D. Zagzag, J. Olson, T. Curran, C. Wetmore, J. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. Louis, J. Mesirov, E. Lander, and T. Golub, "Prediction of central nervous system embryonal tumour outcome based on gene expression," *Nature*, vol. 415, no. 6870, pp. 436–442, January 2002.

[44] L. Yan, M. Yang, H. Guo, L. Yang, J. Wu, R. Li, P. Liu, Y. Lian, X. Zheng, J. Yan *et al.*, "Single-cell rna-seq profiling of human preimplantation embryos and embryonic stem cells," *Nature Structural & Molecular Biology*, vol. 20, no. 9, p. 1131, 2013.

[45] M. Goolam, A. Scialdone, S. J. Graham, I. C. Macaulay, A. Jedrusik, A. Hupalowska, T. Voet, J. C. Marioni, and M. Zernicka-Goetz, "Heterogeneity in oct4 and sox2 targets biases cell fate in 4-cell mouse embryos," *Cell*, vol. 165, no. 1, pp. 61–74, 2016.

[46] Q. Deng, D. Ramsköld, B. Reinius, and R. Sandberg, "Single-cell rna-seq reveals dynamic, random monoallelic gene expression in mammalian cells," *Science*, vol. 343, no. 6167, pp. 193–196, 2014.

[47] A. A. Pollen, T. J. Nowakowski, J. Shuga, X. Wang, A. A. Leyrat, J. H. Lui, N. Li, L. Szpankowski, B. Fowler, P. Chen *et al.*, "Low-coverage single-cell mrna sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex," *Nature Biotechnology*, vol. 32, no. 10, p. 1053, 2014.