Probabilistic and Graphical Model Based Genetic Algorithm Driven Clustering with Instance-level Constraints

Yi Hong, Sam Kwong, Hanli Wang, Qingsheng Ren and Yuchou Chang

Abstract—Clustering is traditionally viewed as an unsupervised method for data analysis. However, several recent studies have shown that some limited prior instance-level knowledge can significantly improve the performance of clustering algorithms. This paper proposes a semi-supervised clustering algorithm termed as the Probabilistic and Graphical Model Based Genetic Algorithm Driven Clustering with Instance-level Constraints (Cop-CGA). In Cop-CGA, all prior knowledge about pairs of instances that should or should not be classified into the same groups is denoted as a graph and all candidate clustering solutions are sampled from this graph with different orders to assign instances into a certain number of groups. We illustrate how to design the Cop-CGA to guarantee that all candidate solutions satisfy the given constraints and demonstrate the usefulness of background knowledge for genetic algorithm driven clustering algorithm through experiments on several real data sets with artificial hard constraints. One advantage of Cop-CGA is both positive and negative instance-level constraints can be easily incorporated. Moreover, the performance of Cop-CGA is not sensitive to the order of assignment of instances to groups.

I. INTRODUCTION

Clustering analysis is of great importance with a wide range of applications, such as data mining, data compression and pattern recognition. In its traditional literature, clustering is considered as an unsupervised method for data analysis, that performs under the condition that no information is available concerning the memberships of instances to predefined groups. However in real application domains, it is often the case that some prior background knowledge can be easily obtained. Several recent studies have also shown that this prior background knowledge can greatly improve the accuracies of clustering algorithm [1] [2] [3]. Clustering algorithm incorporates the prior knowledge in a constrained format, which may come from several different sources such as partial labels, individual relationships, spatial contiguity and to just name a few [4].

This paper is mainly interested in instance-level constraints, that are more natural in some scenarios and easier to be collected than labels of instances. For example, in image retrieval systems with user feedback, users are more willing to provide whether a set of retrieved instances are similar or not than to specify labels of instances. Instance-level constraints place restrictions on pairs of instances with regards to their relative memberships. The concept of instance-level constraint was firstly introduced into the area of clustering analysis in [5] and [6]. To boost the accuracies of K-MEANS and COBWEB, two kinds of constraints: Must-link and Cannot-link were proposed and added. Must-link constrains represent that two instances must be partitioned into the same group, while Cannot-link ones specify that two instances must not be placed into the same group. Assignments of instances to a certain number of groups should satisfy the given constraints. Other kinds of instance-level constraints such as soft constraints and space-level constraints have also been suggested in [7] and [8]. In this paper, only Must-link and Cannot-link constraints are considered.

Although clustering algorithms with instance-level constraints greatly outperform their original ones, they still have several limitations. Firstly, in several constrained clustering algorithms, assignments of instances into groups are ordersensitive. An important fact is if a poor assignment order is made, the algorithm may encounter an instance that has no possible valid group. Secondly, clustering is an ill-posed combinatorial optimization problem. Commonly used iterative and hill-climbing methods usually converge to a local optimal partition. Although prior knowledge can direct its search and make it jump over the local optima, in order to obtain a satisfactory result, a large number of constraints should be added. However, it seems sometimes impossible to obtain enough prior knowledge. Moreover, constraints often spoil its search. When the number of constraints is too large, effects of unconstrained instances become weak. The algorithm turns into finding a partition that satisfies all given constraints, but not the one to optimize the clustering criterion [2].

Genetic algorithm (GA) is a class of nature-inspired global optimization methods, that have been widely applied into data clustering. A number of recent studies have shown that GA driven clustering algorithm is often able to find a good partition [9] [10]. However, several factors have limited its performance. First of all, when the clustering criterion can not be exactly defined (It is a common case in the clusterings of many real-world data sets), candidate solutions with low fitness values sometimes maybe even better than the ones with high fitness values. In this case, the potentials of GA loses and the solutions captured by GA may become bad. In addition, to obtain a high quality solution, the crossover and mutation operators should be designed with care. It can be imaged that some prior instance-level knowledge will improve the performance of GA driven clustering algorithm.

Yi Hong, Sam Kwong and Hanli Wang are with the Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong (email: {yihong, CSSAMK}@cityu.edu.hk, wanghl@cs.cityu.edu.hk). Qingsheng Ren is the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, P.R. China. (email: ren-qs@cs.sjtu.edu.cn). Yuchou Chang is with the Department of Electrical and Computer Engineering, Brigham Young University, U.S.A.

This work was supported by City University of Hong Kong, Research Grant: 114707.
However, the crossover and mutation operators of GA often reproduce unfeasible solutions and the conventional GA may not be able to handle so many constraints with commonly-used penalty functions.

In order to solve the problems mentioned above, this paper describes a genetic-guided semi-supervised clustering algorithm termed as the Probabilistic and Graphical Model based Genetic Algorithm Driven Clustering with Instance-level Constraints (Cop-CGA). In Cop-CGA, all prior knowledge about pairs of instances that should or should not be classified into the same groups is denoted as a graph and all candidate clustering solutions are sampled from this graph together with conditional densities with different orders to assign instances into a certain number of groups. In this case, both positive and negative instance-level constraints can be easily incorporated into genetic algorithm driven clustering. To the best knowledge of the authors’, this is the first time to use genetic algorithms to solve semi-supervised clustering with instance-level constraints.

The remainder of this paper is arranged as follows. Section 2 briefly introduces the probabilistic and graphical model based genetic algorithm. Section 3 goes into details of describing Cop-CGA. Our experimental results and their analysis are given in section 4. Section 5 concludes this paper.

II. PROBABILISTIC AND GRAPHICAL MODEL BASED GENETIC ALGORITHM

Genetic algorithm (GA) works with crossover and mutation operators to mix and reproduce good building blocks. The mutation operator performs to reproduce new building blocks, while the crossover combines building blocks contained in two different individuals to form new better ones. It has been known that these two operators often cause the disruption of good building blocks. Various methods have been proposed to prevent the disruption of important building blocks. Among them is the probabilistic and graphical model based genetic algorithm (PMGA), also called as estimation of distribution algorithms [15]. Completely abandoning commonly-used crossover and mutation operators, PMGA explicitly captures important building blocks through estimating the density of promising solutions. Numbers of recent studies have shown a close correspondence between PMGA and GA. The process of building and sampling a density of promising solutions in PMGA is in fact to mimic the behavior of an ideal crossover operator in GA that combines partial solutions with minimal disruption and maximal efficiency. Algorithm (1) gives the framework of PMGA. In Algorithm (1), different density estimating models lead to different kinds of PMGA with different characteristics, such as the population based incremental learning (PBIL) [11], the chain structure model based genetic algorithm (MIMIC) [12], the tree structure model based genetic algorithm (COMIT) [13] and the Bayesian network structure based genetic algorithm (BOA) [14]. More information about PMGA can be consulted in [15].

### Algorithm 1 Probabilistic and Graphic Model based Genetic Algorithm.

1. Set the initial density function \( p(x) \);
2. Generate \( M \) individuals from sampling the density function \( p(x) \);
3. Select \( N (N < M) \) individuals with high fitness to form a subpopulation;
4. Estimate the density function \( p(x) \) of individuals in the subpopulation;
5. If the termination condition is not met, go to (2).

PMGA will be a good alternative to solving problems of data clustering with instance-level constraints. The reasons include three folds. Firstly, like conventional GA, it is easy for PMGA to get a near optimal solution for most combinatorial optimization problem. Secondly, there is neither crossover nor mutation operator in PMGA, therefore, it does not require to set both the crossover or the mutation rate, that in fact is another optimization problems. Lastly, instance-level constraints are denoted as a graph and candidate solutions are sampled from this graph. PMGA handles all instance-level constraints during sampling candidate solutions from a learned density to make sure all of candidate solutions satisfy the constraints. PMGA has gained several applications in data analysis such as feature selection [16] and additive clustering [17].

III. PROBABILISTIC AND GRAPHICAL MODEL BASED GENETIC ALGORITHM DRIVEN CLUSTERING WITH CONSTRAINTS

A probabilistic and graphical model based genetic algorithm is proposed to solve k-means clustering with constraints, termed as Cop-CGA. Several notations used throughout this paper are illustrated as follows:

- \( k \): Number of groups;
- \( n \): Number of features;
- \( L \): Number of instances in data set;
- \( d_i \): The \( i^{th} \) instance and also the \( i^{th} \) node in the graph;
- \( p(d_i \rightarrow c_i) \): The probability that the \( i^{th} \) instance is partitioned into the group \( c_i \);
- \( p(d_i \rightarrow c_i | Con) \): Probability that \( d_i \) is partitioned into group \( c_i \) under constraint \( Con \);
- \( D(d_i) \): the number of nodes negatively connected (connected by cannot-link constraints) with the node \( d_i \).

Algorithm 2 gives the framework of Cop-CGA. In step (5), commonly used truncation selection is adopted. The finish condition is set as the maximal number of generations.

A. Instance-level knowledge representation

Although several kinds of background knowledge can be used, in this paper, we only consider the instance-level constraints because of their simplicity to be handled and wide applications. Instance-level constraints place restrictions on pairs of instances with regards to their relative memberships.
Algorithm 2 Framework of Cop-CGA.

(1) Preprocess instance-level constraints;
(2) Check and store critical instances which may trigger the stagnancy of assignment;
(3) Set initial probability of each instance \( p(i \rightarrow j) \), for \( i = 1 : L, \ j = 1 : k \), do \( p(d_i \rightarrow c_j) = \frac{k}{k} \);
(4) Generate \( M \) candidates as follows:
   (a) assign all critical instances, if the stagnancy of an assignment is met, backtrack and reassign the labeled critical instances;
   (b) assign other instances through sampling the probability function \( p(\cdot) \);
(5) Select \( N \ (N < M) \) individuals with high fitness to form a subpopulation;
(6) Estimate the probability \( p(d_i), \ i = \{1, 2, ..., L\} \) in the subpopulation;
(7) If the termination condition is not met, go to (4).

Algorithm 3 Merge Must-link constraints.

(1) Assign each instance \( d_i \) a label \( i \), for \( i = 1 : L \) do: label\((i) = i; \)
(2) For \( i = 2 : L \), \( j = 1 : i - 1 \)
   if there is a Must-link constraint between \( d_i \) and \( d_j \), for \( k = 1 : j - 1 \)
      if label\((k) = label\((i) \) or label\((k) = label\((j) \)
        label\((k) = \min\{label\((i) , label\((j) \}\);
        label\((i) , label\((j) \) = \min\{label\((i) , label\((j) \}\}.

Two kinds of constraints: Must-link and Cannot-link are considered in this paper. Must-link constrains represent that two instances have to be in the same group and Cannot-link ones specify that two instances must not be placed in the same group. Instances together with their pairwise relationships can be represented as a graph. In this graph, the vertex \( d_i \) corresponds to the instance \( d_i \) in data set, and an edge connecting two vertexes \( d_i \) and \( d_j \) means that there is a Must-link or Cannot-link constraint between instances \( d_i \) and \( d_j \). Fig. 1 is a case in which the prior instance-level knowledge is denoted as a graph. From Fig. 1, we can see that there are 13 instances in the original data set. Among them, instances \( \{d_1, d_2, d_3, d_{10}\} \), \( \{d_4, d_6\} \) and \( \{d_{11}, d_{12}\} \) must be partitioned into same groups, while instances \( \{d_1, d_3\} \), \( \{d_5, d_{10}\} \), \( \{d_3, d_4\} \), \( \{d_5, d_6\} \), \( \{d_1, d_7\} \), \( \{d_11, d_{12}\} \) and \( \{d_{12}, d_{13}\} \) can not be placed into same groups.

B. Prior knowledge preprocessing

There are many redundant connections in the graph. At the initial stage, we should prune these redundant connections to make the representation of prior knowledge more compact according to the following two steps:

1) Merging Must-link constraints: Noted that Must-link relationship is transitive. If both \( \{d_i, d_j\} \) and \( \{d_i, d_k\} \) have must-link constraints, then \( d_j \) and \( d_k \) must also have a Must-link constraint. In addition, instances connected by Must-link constraints must be partitioned into the same class, we directly merge them as one instance. To realize it, we calculate Must-link closure sets for all instances, and merge instances in its closure set as one vertex in the graph. Full steps to merge must-link constraints are given in Algorithm 3.

2) Removing redundant Cannot-link constraints: After several instances are merged together, there may be the case that there are several edges connecting two nodes in the graph. For example, \( d_i \) and \( d_j \) have a Must-link constraint, \( \{d_i, d_k\} \) and \( \{d_j, d_k\} \) have Cannot-link constraints. After instances \( d_i \) and \( d_j \) are merged together, there are two edges between \( d_k \) and the merged instance. The above problem can be solved by traveling over the graph to remain at most one edge connecting between two nodes.

Fig. 2 is the graph after preprocessing. From Fig.1 and Fig.2, we can observe that the instances \( \{d_1, d_2, d_3, d_{10}\} \), \( \{d_4, d_6\} \) and \( \{d_{11}, d_{12}\} \) are merged together, and many excessive Cannot-link connections are removed. The number of connections after pruned is much less than that of the original one, from 13 to 6, that is more than one half connections.
have been pruned. However, all prior knowledge about Must-link and Cannot-link relationships is remained. The time consumption of merging Must-link constraints is \( O(L^2) \), while the time consumption of removing redundant Cannot-link constraints is \( O(L^2) \).

C. Critical instances assigning
As Wagstaff et al. pointed out that if a poor assignment of instances was made, COP-KMEANS may later encounter an instance \( d_l \) that has no possible valid group [6]. They also suggest using backtracking method to rearrange the assignment. However backtracking over all assigned instances is time-consuming. A more efficient method is proposed in this paper which only backtracks a small number of assigned instances. One necessary condition for the stagnacy of assignment is as follows:

\[
\exists i, i \in \{1, 2, ..., L\} : D(d_i) \geq k
\]

(1)

where \( D(d_i) \) is the number of instances negatively connected with instance \( d_i \) in the graph after preprocessed. If \( D(d_i) < k \), the instance \( d_i \) can be assigned into a feasible group no matter what order is used. However, for all instances \( d_i \) \( (D(d_i) \geq k) \), if a poor assignment order is used, they may be stagnant. Instances which satisfy the following condition is called as critical instances.

\[
S = \{d_i \mid D(d_i) \geq k, i = 1, ..., L\}
\]

(2)

Critical instances are stored in the set \( S \) and assigned before other noncritical instances. If a stagnancy is met, Cop-CGA backtracks only over the assigned instances in \( S \). After instances in \( S \) have been successfully assigned, all unassigned instances can be partitioned no matter what order is adopted. Apart from the difference that critical instances may trigger stagnancy of assignment, assignments of critical instances are the same as that of other instances which is illustrated in following paragraphs.

D. Individuals coding
Like GA, a population of coded solutions are maintained in Cop-CGA. Each solution in the population is considered as a chromosome of length \( L \), where \( L \) is the number of instances to be partitioned. Providing that \( \{d_1, d_2, ..., d_L\} \) is the set of instances, \( k \) is the number of groups, a candidate partition can be coded as: \( \{i_1, i_2, ..., i_L\} \), where \( i_j \) is the corresponding group label of instance \( d_j \) and \( i_j \in \{1, 2, ..., k\} \). An example of a population of coded solutions when \( k = 5 \), \( L = 11 \) and the population size \( M = 2 \) is as follows:

Chromosome 1: 1 5 2 3 4 5 1 4 3 5 1
Chromosome 2: 1 4 2 3 4 1 3 4 5 3 1

Each chromosome represents a partition of the data. For example, chromosome 1 represents the partition \( \{\{d_1, d_7, d_{11}\}, \{d_2, d_{10}\}, \{d_3\}, \{d_4, d_6\}, \{d_5, d_8\}\} \). Given a chromosome \( \{i_1, i_2, ..., i_L\} \), its corresponding partition is \( \{\{d_{i_1}^1, d_{i_1}^2, ..., d_{i_1}^{L_1}\}, ..., \{d_{i_L}^1, d_{i_L}^2, ..., d_{i_L}^{L_L}\}\} \), the following equation must be satisfied:

\[
\sum_{i=1}^{k} L_i = L
\]

(3)

The centroid of the group \( \{d_1^l, ..., d_{L_l}^l\} \) can be calculated as:

\[
c_i = \frac{\sum_{j=1}^{L} d_i^j}{L_i}
\]

(4)

where \( i = \{1, 2, ..., k\} \). If the sum of distances between instances to centroids of their corresponding clusters is adopted as clustering criterion, fitness of chromosome \( I = \{i_1, i_2, ..., i_L\} \) can be calculated as:

\[
F(I) = \sum_{i=1}^{k} \sum_{j=1}^{L_i} e(c_i, d_j^i)
\]

(5)

where \( L_i \) is the number of instances which belong to the \( i^{th} \) group with the centroid \( c_i \), and \( e(d_i^j, c_i) \) is the distance from the instance \( d_i^j \) to the centroid \( c_i \). For numeric features, Euclidean distance is used; while for symbolic features, Hamming distance is adopted. It should be noted that the objective function defined in (5) can be revised for different measures to evaluate the quality of a candidate clustering solution.

E. Density function estimation
In probabilistic and graphical model based genetic algorithm (PMGA), the population is sampled from a density function \( p(x) \), which is learned from the selected subpopulation in the above generation. To build the selected subpopulation, commonly used selection methods such as tournament selection and truncation selection can be used. In this paper, truncation selection is adopted. Truncation selection ranks all solutions according to their fitness and chooses part of individuals with the highest fitness to form the selected subpopulation. Providing that \( \{I_1, I_2, ..., I_N\} \) are the chosen promising solutions in the selected subpopulation and \( I_l^k \) represents the instance \( d_i \) in the solution \( I_k \), the frequency of instance \( d_i \) to be partitioned into the \( j^{th} \) group in the selected subpopulation can be calculated as:

\[
q(d_i \mapsto c_j) = \frac{\sum_{l=1}^{N} \delta(I_l, c_j)}{N}
\]

(6)

where \( \mapsto \) means "be partitioned into", and

\[
\delta(I_l, c_j) = \begin{cases} 1 & \text{if } I_l \mapsto c_j; \\ 0 & \text{otherwise;} \end{cases}
\]

(7)

If \( p_i(d_i \mapsto c_j) \) represents the probability that the instance \( d_i \) is partitioned into group \( c_j \) in the \( l^{th} \) generation, then in the \( (l+1)^{th} \) generation, its probability \( p_{i+1}(d_i \mapsto c_j) \) is updated as:

\[
p_{i+1}(d_i \mapsto c_j) = (1 - \lambda) \cdot p_i(d_i \mapsto c_j) + \lambda \cdot q(d_i \mapsto c_j)
\]

(8)

where \( \lambda \) is the learning rate. If no constraint is added, the probability of the solution \( I \) in the \( (l+1)^{th} \) generation is
where $k$ is the number of groups. Let $I_{\text{best}}$ denote the partition with the minimal total within-cluster variation defined by formula (5) and $p_l(I_{\text{best}})$ is its probability in the $l^{th}$ generation, then its probability in the $(l + 1)^{th}$ generation can be calculated as [18]:

$$
p_{l+1}(I_{\text{best}}) = \frac{1}{\eta} p_l(I_{\text{best}})
$$

(14)

where $\eta$ is the selection proportion.

Formula (15) means the truncation selection chooses candidate solutions whose fitness are higher or equal to $F(T)$ to construct the selected subpopulation. If the population have not converged, GA must choose a $T$ that satisfies $F(T) > \min(F(.))$. Considered that $\int p_l(I)dI = 1.0$, therefore $\eta < 1$. In our experiments, $\eta$ is fixed to 0.3. Combining formula (14) and (15), we get:

$$
\lim_{l \to \infty} p_l(I_{\text{best}}) = 1
$$

(16)

The above equation tells us when the population size is infinite, Cop-CGA converges to the partition with the minimal total within-cluster variation satisfying the given constraints.

It should be noted that the structure of Cop-CGA is an undirected graph and sampling individuals from a graph is order-sensitive. To prevent this problem, Cop-CGA uses a flexible structure and different solutions may be sampled from different Bayesian networks deduced from the given graph through adding directions on edges. To realize it, Cop-CGA selects an unlabeled instance $d_i$ at random, then assigns the instance $d_i$ and all unlabeled instances connected with $d_i$ with its conditional density function calculated by formula (12), this step iterates until all directly or indirectly connected instances have been labeled. If there are still

---

**Fig. 3.** Graphic structures to generate new individuals.
some unlabeled instances, an unlabeled instance is randomly picked. The above steps run again. Fig. 3 gives an example. In this example, the number of groups equals to 3 and critical instances $\{d_1, d_2, d_3, d_4\}$ are firstly assigned. Constrained relationships among instances are denoted as a graph (see Fig. 3(a)) and three different Bayesian networks (b),(c),(d) are extracted from this graph and each Bayesian network corresponds to one order to assign the instances. All candidate solutions are sampled from these different Bayesian networks together with their conditional densities calculated by the formula (12).

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. Synthetic data

This data set includes 80 instances sampled from four Gaussian distributions with the centers: $G_1 = (-0.5, 0.5), G_2 = (-0.5, -0.5), G_3 = (0, 0.5), G_4 = (0, -0.5)$. All clusters have 20 instances coming from two groups: Class I and Class II (See Fig. 4(a)). Parameters setting for Cop-CGA are as follows: population size equals to 100, the learning rate $\lambda$ is fixed to 0.2 and the selection proportion $\eta$ equals to 0.2. Genetic algorithm driven K-means clustering algorithm is not able to solve this problem. Its clustering accuracy is around 50% and Fig. 4(c) gives its clustering result. If some prior instance-level constraints about pairs of instances should or should not be classified into the same group is available (See Fig. 4(b)), its clustering accuracy can be greatly improved. Fig. 4(d) shows that Cop-CGA obtains a clustering accuracy around 100% when the number of constraints is up to 60. This result lets us know that a competent search method such as genetic algorithm is not enough for many ill-posed clustering problems and some prior knowledge can significantly improve its clustering accuracy.

B. Real data

Three data sets from UCI machine learning repository are selected: Iris data set (150 instances, 4 attributes, 3 groups), Wine data set (178 instances, 13 attributes, 3 groups) and Glass data set (214 instances, 9 attributes, 6 groups). Clustering accuracies of Cop-CGA are tested and its results are compared with the constrained K-means (Cop-KMEANS) [6] and the constrained EM (Cop-EM)[19]. Our evaluation method is based on the Rand Index method. Parameters setting for Cop-CGA are as follows: population size equals to the number of instances, the learning rate $\lambda$ is fixed to 0.2 and the selection proportion $\eta$ equals to 0.2. All algorithms are executed for 30 independent runs and their average results are reported. Instance-level constraints are generated through randomly picking two instances from the data set and checking their labels.

Accuracies of clustering solutions are shown in Fig. 5. If no instance-level constraint is added, genetic driven K-means clustering algorithm can get around 90% for Iris data set, 75% for Wine data set, 70% for Glass data set. The results are better than K-means clustering algorithm and EM clustering algorithm, whose corresponding accuracies are around 84% for Iris, 71% for Wine, 68% for Glass. However, after some instance-level constraints are added, the performances of K-means and EM can be significantly boosted. Cop-KMEANS achieves an accuracy of 94.3% for Iris, 93.7% for wine, 78.2% for Glass, if the number of constraints increases up to 400. This great improvement tells us that instance-level constraints can significantly improve the accuracy of K-means and EM clustering algorithms. Like K-means and EM, the same improvement can also be observed from genetic algorithm driven K-means. Among these three compared algorithms, Cop-CGA is the best. Its clustering accuracies are much better than other two methods, 99.3% for Iris, 97.5% for Wine, 87.9% for Glass. The above comparisons demonstrate that a good clustering algorithm is the combination between an accurate clustering criterion and a competent search method. On one hand, since it is often impossible to define an accurate clustering criterion, even for a competent search method such as GA, some limited prior knowledge is necessary to be incorporated. On the other hand, the improvement of clustering algorithm through adding constraints (prior knowledge) is sometimes not enough, and a competent search method such as GA can significantly boost its accuracy.

Apart from clustering accuracies, convergent velocity of Cop-CGA is also tested. Generations when Cop-CGA converges under different numbers of instance-level constraints are shown in Fig. 6. An interesting phenomenon observed from Fig. 6 is when the number of constraints is small, Cop-CGA requires more generations to converge than that of Cop-CGA without constraints. However, the number of generations required for convergence starts to decrease, when the number of constraints increases up to a certain value. The above phenomenon indicates that although instance-level constraints can improve the accuracy of clustering algorithms, they also increase the time-consumption and difficulty of the search for a good clustering result.

V. CONCLUSIONS

In this paper, a novel semi-supervised clustering algorithm Cop-CGA is proposed, which can incorporate instance-level constraints in a natural way. Experimental results have demonstrated that the proposed Cop-CGA can achieve better performance than the other two state-of-the-art constrained clustering algorithms Cop-KMEANS [6] and Cop-EM [19]. Note that the proposed probabilistic and graphical model based genetic algorithm driven scheme not only can be used for constrained K-means clustering problems, but also can be easily extended for other kinds of semi-supervised clustering algorithm, e.g. EM clustering algorithm. In the future, it is also great of interest to incorporate other kinds of constraints such as the soft constrains besides the Must-link and Cannot-link constraints into the current framework of Cop-CGA.

REFERENCES


Fig. 4. Experimental results for synthetic data set.
The number of instance-level constraints

(A) IRIS DATA SET

Accuracy

Cop-CGA
Cop-EM
Cop-Kmeans

(B) WINE DATA SET

Accuracy

Cop-CGA
Cop-EM
Cop-KMeans

(C) GLASS DATA SET

Accuracy

Cop-CGA
Cop-EM
Cop-KMeans

Fig. 5. Clustering accuracies for real data sets.

Generations to converge

(A) IRIS DATA SET

(B) WINE DATA SET

(C) GLASS DATA SET

Fig. 6. Converged generations for real data sets.