Electrical Load Pattern Shape Clustering Using Ant Colony Optimization

Fernando Lezama(✉), Ansel Y. Rodríguez, Enrique Muñoz de Cote, and Luis Enrique Sucar(✉)

Department of Computer Science, Instituto Nacional de Astrofísica, Óptica Y Electrónica, Puebla 72840, Mexico
{f.lezama,ansel,jemc,esucar}@inaoep.mx

Abstract. Electrical Load Pattern Shape (LPS) clustering of customers is an important part of the tariff formulation process. Nevertheless, the patterns describing the energy consumption of a customer have some characteristics (e.g., a high number of features corresponding to time series reflecting the measurements of a typical day) that make their analysis different from other pattern recognition applications. In this paper, we propose a clustering algorithm based on ant colony optimization (ACO) to solve the LPS clustering problem. We use four well-known clustering metrics (i.e., CDI, SI, DEV and CONN), showing that the selection of a clustering quality metric plays an important role in the LPS clustering problem. Also, we compare our LPS-ACO algorithm with traditional algorithms, such as k-means and single-linkage, and a state-of-the-art Electrical Pattern Ant Colony Clustering (EPACC) algorithm designed for this task. Our results show that LPS-ACO performs remarkably well using any of the metrics presented here.

Keywords: Ant colony optimization · Electrical load patterns · Clustering · Clustering quality metrics

1 Introduction

With the inclusion of energy markets into the smart grid, customer clustering plays an important role in identifying client niches to assist the tariff formulation process. To this end, the curves known as load pattern shape (LPS) represent a customer's typical energy consumption during a day (24 h periods). LPSs are key to design effective customer tariffs and can be crucial to take decisions like promoting strategies for peak load reduction or to exploit the willingness of a client to accepts price-based demand conditions by demand programs [1].

The formation of customer classes based in LPS is a multi-stage approach involving the gathering of LPSs curves from customers, and applying "*clustering*" techniques based on pre-defined features extracted from the gathered information.

Broadly speaking, clustering is an unsupervised machine learning technique for partitioning an unlabeled data set into groups, in which the objects that belong to each group share some similarity between themselves, and are dissimilar to objects of other groups [2]. However, electrical LPSs have some characteristics that make their analysis different from other pattern recognition applications. For instance, the number of features (e.g., time series reflecting the electrical consumption of customer's typical day) is relatively high. Moreover, the power of such curves normalized in the [0,1] range and considering the same period of observation causes the overlapping of the patterns.

For the LPS clustering, some of the most studied algorithms include the classical k-means, fuzzy k-means and hierarchical clustering [3,4]. More recently, a clustering algorithm based on a centroid model using some concepts of the Ant Colony Optimization (ACO) algorithm was proposed for LPS clustering [1,5]. The algorithm accepts the number of cluster as input and incorporates a mechanism to guarantee the persistence of the centroids in the iteration process.

The ACO algorithm was first introduced by Dorigo et al. [6]. ACO is a swarm intelligence technique inspired by the behavior of real ants and targets discrete combinatorial optimization problems. The ACO algorithm is also a source of inspiration for the design of novel algorithms for the solution of optimization and distributed control problems.

In this paper, a modified ACO algorithm is applied to the LPS clustering problem, considering a specific number of clusters and an initial centroid model. Our proposed LPS-ACO algorithm implement two traces of pheromone that allow a more efficient information exchange between ants. We compare our LPS-ACO implementation against some classical algorithm, such as k-means and single-linkage, and also against the Electrical Pattern Ant Colony Clustering (EPACC) which is an heuristic algorithm that uses some ACO principles [1].

2 Related Work

Electrical LPS clustering has particular characteristics concerning other pattern recognition application. For this reason, new and modified clustering algorithms have been proposed to date. In [4], a framework for the consumer electricity characterization based on a combination of unsupervised and supervised learning techniques, such as k-means or fuzzy k-means, was proposed. Other works, as in [7], used a probabilistic neural network (PNN) as a method for allocating consumer's load profiles. An interesting comparison on the use of various unsupervised clustering algorithms (e.g., modified follow-the-leader, hierarchical clustering, K-means, fuzzy K-means) and the self-organizing maps (SOM) can be found in [3]. Also, LPSs clustering can be used in forecasting household-level electricity demand to assure balance in low-voltage networks [8].

Different approaches using evolutionary and bio-inspired algorithms have been proposed in the literature for the general problem of clustering [2,9,10]. More recently, In particular for the LPS problem an algorithm based on a user centroid model and assisted by ant colony principles was presented in [1,5]. The authors cleverly shape the information related to the distances of each LPS to the initial model selected by user. Nevertheless, their approach fails by ignoring useful information that ants could provide for the reinforcement of the pheromone trail, making its algorithm less flexible. Moreover, they compare their results against k-means only and under a criterion of just two quality metrics.

Against this background, here we propose an algorithm inspired by [1,5], but as opposed to them, our LPS-ACO implement two pheromone trails. One pheromone trail captures the information on the distance to the initial centroids defined by the user. The other pheromone trail uses the information related to the fitness (measured with a quality metric) of the solutions found. In this way, we allow information exchange between ants in a more efficient way, providing our algorithm with more flexibility to optimize different metrics.

3 Problem Formulation

A pattern is an object, physical or abstract, that has attributes called features that make it distinguishable from others [2]. In this paper, each customer is characterized as an LPS $p_i \in P$ containing $d \in \mathcal{R}$ points representing the energy consumption for a given period of observation. The initial profile data matrix $P_{n \times d}$ contains the n LPSs corresponding to each customer.

In general, given a set of objects P (customers LPSs in this case), a clustering algorithm tries to find a partition $C = \{c_1, c_2, ..., c_K\}$ of K classes such that the similarity between patterns of each class is maximum, and the patterns of different classes differ as much as possible. The partition should maintain three properties [2]: (1) Each cluster should have at least one pattern assigned; (2) Two different clusters should have no pattern in common; (3) Each pattern should be attached to a cluster.

The clustering problem can be formally defined as an optimization problem of finding the clustering partition C^* for which [9]:

$$f(C^*) = \min_{C \in \Omega} f(C) \tag{1}$$

where Ω is the set of feasible clusterings, C is a clustering of a given set of data P, and f is a statistical-mathematical function (i.e., the objective function) that quantifies the goodness of a partition by similarity or dissimilarity between patterns.

3.1 Clustering Quality Metrics

Clustering quality metrics are mathematical functions used to evaluate the accuracy of a clustering algorithm. Ideally, a quality metric should take care of two aspects: Cohesion (i.e., the patterns in one cluster should be as similar as possible) and separation (i.e., the clusters should be well separated).

Some of the most popular metrics used in the literature are the Dunn's index (DI) [11], the Davies-Bouldin index (DB) [12], the Chou-Su (CS) index [13], among others. Because of their optimizing nature (i.e., the maximum or minimum value indicates a proper partition), these metrics are used in association with evolutionary algorithms, such as GA, PSO or DE [2]. For the particular problem of LPS clustering, in [5], other quality metrics named the Clustering Dispersion Indicator (CDI) and the Scatter Index (SI), were used to guide the population and assess the quality of the solutions found.

Despite the effectiveness of such metrics for some applications, it is important to keep in mind that these metrics by themselves optimize some criterion but sometimes do not reflect good partitions from a broad perspective.

For that reason, in the next subsection, we explain the use of some other metrics used as objective functions (i.e., fitness function) in this paper.

3.2 Objective Functions

The selection of a proper metric as an objective function to guide the algorithm is a key aspect of our LPS-ACO algorithm. In this paper, we analyze two quality metrics (CDI and SI) defined in [5], and two complementary metrics, one based on compactness and another based on connectedness, defined in [9]. The analysis of these metrics gives a broad perspective of the quality of the solutions found.

Before presenting the mathematical definition of the analyzed metrics, some considerations have to be stated:

- A pattern is a physical or abstract structure of objects distinguish from others by a set of attributes called features [2]. In the LPS problem, $P_{n\times d}$ is a data matrix with n (i.e., the number of LPSs) d-dimensional patterns p_i. Each element in $p_i \in P$ corresponds to the jth real-value (i.e., feature $j = 1, 2, ..., d$) of the ith pattern.
- Clustering is a process of grouping patterns based on some similarity measures. The most used way to evaluate similarity is by using some distance measure. In this paper, we adopt the use of Euclidean distance as a similarity measure, since is widely used and is well-known [2].

CDI and SI Metrics. The CDI metric is defined under the principle that better clustering partitions have relatively low internal variation, and the centroids ($\delta_j \in \varDelta$) of each clusters $c_j \in C$ are as far as possible from each other. CDI is defined as:

$$\mathrm{CDI}(C) = \frac{\sqrt{\frac{1}{K}\sum_{j=1}^{K} d^2(c_j)}}{d(\varDelta)} \tag{2}$$

where $d(\bullet)$ is a function that computes the average distance between patterns in cluster $c_j \in C$, and $\varDelta$ is the set of centroids belonging to each cluster.

The SI metric is calculated with reference to a pooled scatter $\bar{p} = \frac{1}{n}\sum_i^n p_i$, corresponding to the average of the data set P. SI is defined as:

$$\mathrm{SI}(C) = \frac{\sum_{i=1}^{n} d^2(p_i, \bar{p})}{\sum_{j=1}^{K} d^2(\delta_j, \bar{p})} \tag{3}$$

where δ_j is the centroid of cluster c_j, n is the number of LPS and K is the number of clusters.

Lower values of CDI and SI indicate better clustering solutions. The main problem with these two metrics, in the particular problem of the LPS clustering, is that the optimal value of CDI or SI favors the creation of clusters under the principle of connectedness. That behavior can be explained observing Eq. (2). Since all the LPS are normalized in the [0,1] range (i.e., there is a little spatial separation between them), the distance of an LPS to its centroid is on average the same. On the other hand, if a cluster is made of just one LPS, the distance of that LPS to its centroid is 0 (i.e., the LPS $p_i \in c_j$ is its centroid). Due to this behavior, the metric will be optimized putting many LPSs as single clusters (i.e., the contribution of the distance to its centroids is 0), and letting a unique cluster with the rest of the patterns.

For that reason, we also analyze two complementary metrics, overall deviation (DEV) and Connectivity (CONN), as proposed in [9]. These two metrics are related to compactness and connectedness. In that way, we can analyze from a broad perspective which metric suits better for the specific problem of LPS clustering.

DEV and CONN Metrics. The DEV metric measures the compactness of a solution and is defined as:

$$\mathrm{DEV}(C) = \sum_{c_j \in C} \sum_{p_i \in c_j} d(p_i, \delta_j) \tag{4}$$

where C is the set of clusters, δ_j is the centroid of cluster c_j and $d(\bullet)$ is a distance function (e.g., Euclidean distance). This metric computes the overall summed distances between patterns and their corresponding cluster center. Again, lower values of DEV indicates better clustering performance.

On the other hand, to compute the CONN metric, first it is necessary to compute a proximity matrix $Prox_{n\times L}$. $Prox_{n\times L}$ will contain the L closer patterns $\forall p_i \in P$. A distance function (e.g., Euclidean distance) is used once in the initialization process to determine the L closer patterns to each p_i. Once $Prox_{n\times L}$ is computed, CONN is defined as:

$$\mathrm{CONN}(C) = \sum_{i=1}^{n} \left(\sum_{l=1}^{L} x_{i,nn_{il}} \right)$$

$$\text{where} \quad x_{i,nn_{il}} = \begin{cases} 1/l, \text{if } \nexists \quad c_j : p_i \in c_j \wedge p_{nn_{il}} \in c_j \\ 0 \quad \text{otherwise} \end{cases} \tag{5}$$

where nn_{ij} is the jth nearest neighbor of pattern p_i, n is the number of LPSs, and L is a parameter determining the number of neighbors that contribute to the connectivity measure. CONN metric evaluates the degree of neighboring data points placed in the same cluster by penalizing neighboring data belonging to different clusters. As DEV, lower values of CONN indicates better clustering quality.

4 Ant Colony Optimization and LPS-ACO

ACO algorithm exploits self-organized principles of collaboration between artificial agents (ants) to converge to a good solution. The artificial ants represent specific solutions to the problem and share information via pheromone trails to converge to an optimal solution.

In ACO, iterations involve three main phases: an initialization phase, a solution construction phase and an updating of the pheromone trail phase. The algorithm is run until a stop criterion is met.

4.1 Load Pattern Shape Ant Colony Optimization Algorithm

In the initialization phase, the LPS-ACO parameters are set. These parameters include the size of the colony (M), the threshold γ reflecting a number of iterations without a change in the solutions found and used as a stop criterion, among others specified in Sect. 5.1. After set the parameters, the proximity matrix $Prox_{n \times L}$ (defined in Sect. 3.2) with the L neighbors of each LPS p_i is computed.

The encoding of a solution is crucial in the LPS-ACO algorithm. Each ant is represented as a vector of size n (i.e., the number of LPSs) such as: $a_m = [p_{1 \to c_j}, p_{2 \to c_j}, ..., p_{n \to c_j}]$. The elements of the ant represent the allocation of the ith pattern p_i to a specific class $c_j \in C$.

The pheromone trails are also necessary to the success of the algorithm. In this approach, two pheromone trails that store the information learned from the ants are modeled as matrixes τ and η of dimension $n \times K$. Both pheromone trails reflect the desirability of assigning the LPS $p_i \in P$ to the class $c_j \in C$.

The first pheromone trail (also called the fitness-pheromone trail) $\tau_{i,j}$ is reinforced by the contribution of ants based on the quality of the solution found (measured by a metric). On the other hand, the second pheromone trail (also called the distance-pheromone trail), $\eta_{i,j}$, contains information related to the distance of LPSs to the corresponding centroids of the clusters.

In the solution construction phase, each ant a_m assigns the pattern p_i to the cluster c_j based on the next decision rule: if a random number in the range $[0, 1]$ is less than a probability θ, the ant a_m assigns the pattern p_i to the class c_j that has the major pheromone concentration. Otherwise, the assignation of p_i is determined thorough a stochastic decision policy based on the pheromone levels with probability:

$$prob^m_{p_i \to c_j} = \alpha * \tau_{i,j} + (1 - \alpha) * \eta_{i,j} \tag{6}$$

where α is a constant weight that controls the importance of one or another pheromone matrix.

After the construction of all the solutions, the pheromone trails are updated with particular rules as explained next.

First, the fitness-pheromone trail values are increased on the clusters that ants have selected during their solution construction phase. The fitness-pheromone trail update is implemented as:

$$\tau_{i,j} \leftarrow \tau_{i,j} + \sum_{m=1}^{M} \Delta\tau_{i,j}^{m} \tag{7}$$

where $\Delta\tau_{i,j}^{m}$ is the amount of pheromone ant a_m deposits on the clusters it has selected. $\Delta\tau_{i,j}^{m}$ is defined as:

$$\Delta\tau_{i,j}^{m} = \begin{cases} \mathrm{fit}_m, \text{if } p_i \in c_j \in a_m \\ 0 \quad \text{otherwise} \end{cases} \tag{8}$$

where fit_m is the cost of the solution built by a_m. Moreover, we use the concept of elitist ant system (EAS) to provide additional reinforcement to the assignations that belong to a_{best}, which is the best solution found by the algorithm at each iteration. The additional reinforcement is achieved by adding $e_w * \mathrm{fit}_{best}$ to the assignations done by a_{best}.

On the other hand, the distance-pheromone $\eta_{i,j}$ is updated with the information of the inverse distance from each LPS to the centroids δ_{best} corresponding to the solution found by a_{best}.

After the initial calculation and every update of both pheromone matrixes, using the concept of hyper-cube ACO [14], the values of both trails are normalized. This normalization not only avoids the continuous increase of pheromone trails during iterations but also eliminates the necessity of an evaporation rate constant. The normalization is done by dividing each row by its greater value.

A detailed pseudocode of the LPS-ACO is shown in Fig. 1.

5 Results and Discussion

We applied the LPS-ACO algorithm for the clustering of LPSs. The initial data set $P_{n\times d}$ has been obtained from real measures considering $n = 250$ customers[1]. Each LPS was normalized, in the [0,1] range, and was obtained taking the average measurements of the five days of a regular week (without weekend days) in 30 minutes intervals (i.e., each LPS is an average of five days with $d = 48$ points). Figure 2 shows the set of LPS of each customer.

The numerical results section is divided into two parts. First we present the parameter tuning of the LPS-ACO algorithm. After that, the performance of the LPS-ACO algorithm using different metrics and comparing our results with some classical algorithms for clustering is presented.

[1] For simplicity, in this work only the first 250 LPSs were considered from the available data on-line: http://www.ucd.ie/issda/data/commissionforenergyregulationcer.

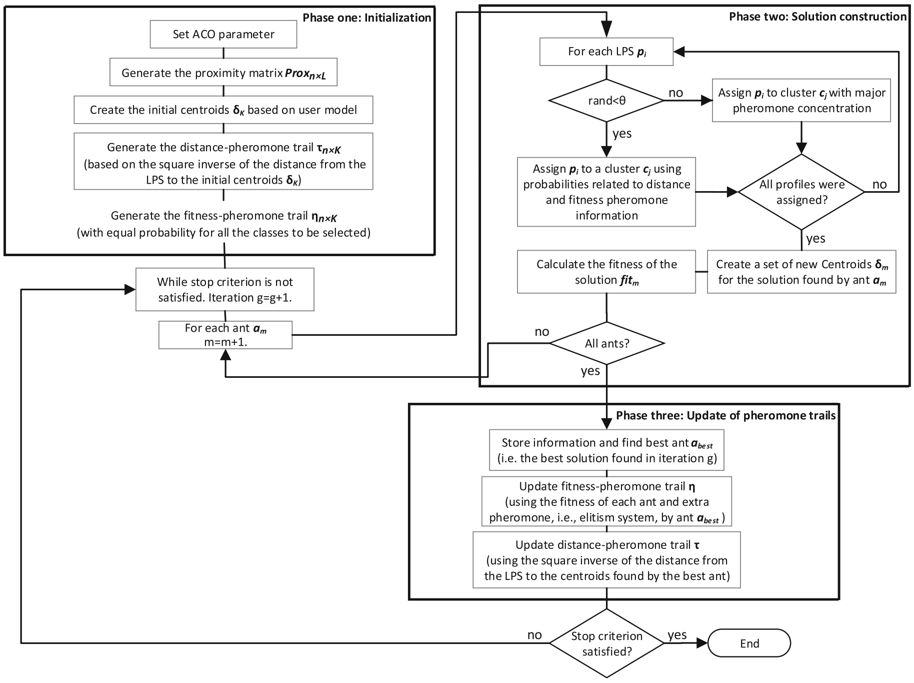
Phase one: Initialization
Set ACO parameter
Generate the proximity matrix Prox n×L
Create the initial centroids δK based on user model
Generate the distance-pheromone trail τn×K (based on the square inverse of the distance from the LPS to the initial centroids δK)
Generate the fitness-pheromone trail ηn×K (with equal probability for all the classes to be selected)
While stop criterion is not satisfied. Iteration g=g+1.
For each ant am m=m+1.
Phase two: Solution construction
For each LPS pi
rand<θ
no
yes
Assign pi to cluster cj with major pheromone concentration
Assign pi to a cluster cj using probabilities related to distance and fitness pheromone information
All profiles were assigned?
Calculate the fitness of the solution fitm
Create a set of new Centroids δm for the solution found by ant am
All ants?
Phase three: Update of pheromone trails
Store information and find best ant abest (i.e. the best solution found in iteration g)
Update fitness-pheromone trail η (using the fitness of each ant and extra pheromone, i.e., elitism system, by ant abest)
Update distance-pheromone trail τ (using the square inverse of the distance from the LPS to the centroids found by the best ant)
Stop criterion satisfied?
End

Fig. 1. LPS-ACO clustering algorithm.

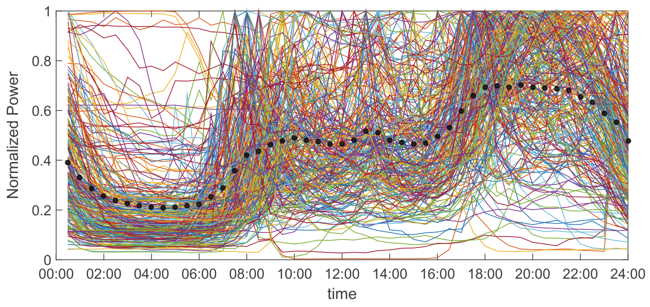

Fig. 2. 250 normalized load pattern shapes. The black dotted line correspond to the pooled scatter $\bar{p}$ used to compute Eq. (3).

5.1 Tuning of the Parameters

It is worth noting that the LPS-ACO algorithm requires the manipulation of few control parameters. These parameters include: θ controlling whether the assignation of p_i to c_j is done based on the major pheromone concentration or in a probabilistic choice; α controlling whether the ants follow the trace of fitness-pheromone, the trace of distance-pheromone, or a combination of both; υ used as stop criterion reflecting the maximum number of iterations without change in the value of the objective function; M is the size of the colony (i.e., the number of ants); and e_w which is an Elitism weight used in the EAS to provide extra pheromone by the best ant.

These parameters have an impact on the quality of the solutions. We performed a parametric analysis using a default value for each parameter. The default values are set to: $\theta = 0.95$, $\alpha = 0.5$, $\upsilon = 20$, $M = 20$ and $e_w = 1$. We use a sweeping technique to assess their impact on the solutions, one at a time. The reported values correspond to the average of 50 independent tests. The 95 % confidence intervals (CI) and the standard deviation (Std) were also calculated. The experiments were done optimizing CDI, DEV and CONN metrics[2]. Lower values reflect better clustering performance. The best values found in each experiment are shown in bold.

The results of the first experiment are present in Table 1. This table shows the importance of selecting a proper value of the θ parameter. Notice that the best results with the three metrics were obtained when the θ value is set to 1 (shown in bold). The value of 1 means that the decision on the assignation of p_i to the class c_j is always done selecting the cluster with the major pheromone concentration and letting aside the probabilistic decision rule. Nevertheless, in Table 2, a more fine tuning is presented in the range of [0.9,1] with steps of 0.01. In this range of analyzed values, the best setting for θ to optimize the CDI and the DEV metrics were 0.97 and 0.95 respectively. On the other hand, if the user is interested in optimizing the CONN metric, the results suggest that the probabilistic choice is not necessary, and we can limit our algorithm to follow the trace of the major pheromone concentration.

For the second experiment, we analyze the impact of the α parameter used in Eq. (6). Table 3 shows an interesting behavior related to this parameter. For instance, if we are interested in optimizing CDI or CONN metrics, the results suggest that the α parameter should be set to 1. A value of 1 implies that we neglect the effect of the distance-pheromone matrix η in Eq. (6). In other words, the ants will take the decision based just on the fitness-pheromone matrix τ. On the other hand, if the user wants to optimize the DEV metric, the value of α should be set to 0, which means exactly the opposite (i.e., ants will only follow the trace of the distance-pheromone matrix η). Even when the best values for each metric were found activating a single pheromone trace (i.e., either η or τ), it is important to recall that the real optimal solution might be found in a trade-off of different metrics (e.g., the complementary metrics DEV and CONN) [9]. This suggest the possibility of an application of a multi-objective approach, let it as a further work and out of the scope of this research.

When using heuristic algorithms, it is well-known that setting a maximum number of iterations as a stop criterion is practically ineffective. A best practice is the use of a criterion that reflect the evolution and no variations of the solutions during the iteration process, and the use a maximum number of iterations just to prevent lack of convergence or an excessive computation time. For this reason,

[2] SI metric was not taken into account in the parameter tuning since its behavior is similar to the results obtained with CDI metric.

the third experiment is related to the γ parameter. This parameter is used as a stop criterion, reflecting a max number of consecutive iterations without a change in the best solution found. In this way, the stop criterion reflects the evolution of the solutions in the iterative process. Table 4 shows variations of the quality of the solutions when γ vary from 1 to 100. The best values are presented in bold. However, notice that the difference between the means of the tested values do not have a high variation (e.g., for the CONN metric, the best value with $\gamma = 100$ is almost equal to that with $\gamma = 70$). Moreover, increasing the value of γ will have an impact on the number of iterations needed before the algorithm stops. For that reason, a medium value (i.e., in the [20,50] range) is enough to find acceptable solutions without increase the execution time.

The results of the fourth experiment are related to the size of the colony M and are presented in Table 5. It can be noticed that the variations of the quality of the solutions in the range of [10,100] are small. However, more ants imply a greater number of constructions in the second phase of the algorithm, taking more time of execution if the ants construct their solutions sequentially rather than in parallel. Moreover, with the increase in the size of the colony, more objective functions evaluations are needed to determine the fitness of each ant slowing the execution of the algorithm. To reduce the number of objective function evaluations an M value in the range of [10,30] is recommended.

Finally, the fifth experiment is related to the e_w parameter used to control an additional contribution of pheromone by the best ant in the elitist ant system. Table 6 shows the values of the analyzed metrics varying e_w parameter. It can be noticed that this parameter has a small impact on the CDI metric. However, when the DEV metric is under optimization, small values of e_w are preferred for the algorithm (i.e. the best value was found when the e_w is set to 1). On the other hand, for the CONN metric an intermediate value of the parameter e_w (e.g., in the range of [20,50]) gives the best results.

Table 1. θ parameter tuning in the range $[0, 1]$ with 0.1 steps

		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	**1**
CDI	Mean	1.7036	1.5728	1.5455	1.3494	1.1804	0.9788	0.8341	0.7268	0.5972	0.4792	**0.4378**
	± CI 95 %	0.1002	0.0699	0.0659	0.0420	0.0320	0.0243	0.0187	0.0196	0.0118	0.0070	**0.0000**
	Std	0.3525	0.2460	0.2318	0.1476	0.1127	0.0855	0.0657	0.0688	0.0416	0.0247	**0.0000**
DEV	Mean	331.06	326.24	317.33	307.75	294.42	281.26	265.57	249.89	235.97	224.85	**224.54**
	± CI 95 %	0.54	0.40	0.80	1.01	1.32	1.33	1.86	1.81	1.43	1.39	**0.00**
	Std	1.92	1.40	2.80	3.55	4.64	4.69	6.56	6.38	5.04	4.90	**0.00**
CONN	Mean	838.17	835.12	818.29	792.67	760.38	723.36	666.10	612.55	529.97	453.83	**366.08**
	± CI 95 %	1.09	2.60	3.97	4.68	5.69	5.97	6.33	7.60	8.12	5.61	**0.00**
	Std	3.84	9.14	13.96	16.48	20.03	21.00	22.27	26.74	28.57	19.72	**0.00**

As a summary, the recommended values for each metric and each parameter are presented in Table 7. These values are used in Sect. 5.2 when comparing our LPS-ACO against other clustering algorithms.

Table 2. θ parameter fine tuning in the range $[0.9, 1]$ with 0.01 steps

		0.9	0.91	0.92	0.93	0.94	**0.95**	0.96	**0.97**	0.98	0.99	**1**
CDI	Mean	0.4736	0.4676	0.4556	0.4522	0.4460	0.4419	0.4338	**0.4299**	0.4311	0.4309	0.4378
	± CI 95 %	0.0054	0.0087	0.0046	0.0064	0.0066	0.0088	0.0034	**0.0022**	0.0069	0.0067	0.0000
	Std	0.0190	0.0307	0.0163	0.0225	0.0231	0.0310	0.0118	**0.0078**	0.0244	0.0234	0.0000
DEV	Mean	223.74	223.24	223.05	222.23	220.88	**220.71**	220.91	220.89	222.05	223.15	224.54
	± CI 95 %	1.25	1.02	0.89	1.22	0.58	**0.80**	0.72	0.68	0.57	0.20	0.00
	Std	4.40	3.59	3.13	4.29	2.06	**2.80**	2.53	2.40	2.01	0.69	0.00
CONN	Mean	447.64	444.90	437.41	427.43	419.02	411.44	398.53	393.24	386.17	376.38	**366.08**
	± CI 95 %	5.86	6.42	5.39	5.43	4.59	4.22	3.32	3.66	2.92	2.09	**0.00**
	Std	20.61	22.59	18.97	19.10	16.14	14.86	11.68	12.86	10.29	7.35	**0.00**

Table 3. α parameter tuning

		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	**1**
CDI	Mean	0.4228	0.4175	0.4270	0.4436	0.4384	0.4407	0.4376	0.4367	0.4336	0.4296	**0.1759**
	± CI 95 %	0.0089	0.0047	0.0057	0.0040	0.0033	0.0056	0.0040	0.0034	0.0026	0.0026	**0.0065**
	Std	0.0314	0.0164	0.0201	0.0139	0.0116	0.0197	0.0139	0.0119	0.0092	0.0092	**0.0229**
DEV	Mean	**215.18**	215.79	217.02	217.73	219.70	221.38	223.57	225.84	232.29	241.35	311.54
	± CI 95 %	**0.77**	0.61	0.82	0.72	0.86	0.71	0.87	0.76	1.13	0.67	0.91
	Std	**2.69**	2.14	2.88	2.53	3.04	2.49	3.06	2.68	3.98	2.37	3.21
CONN	Mean	401.83	406.61	406.18	406.12	413.24	408.22	405.98	410.27	409.92	395.71	**129.12**
	± CI 95 %	3.98	4.17	4.48	4.57	3.69	3.86	4.57	4.51	4.34	5.35	**5.08**
	Std	14.00	14.67	15.77	16.08	12.99	13.59	16.07	15.87	15.27	18.81	**17.86**

Table 4. γ parameter tuning

		1	10	20	30	40	50	**60**	**70**	80	90	**100**
CDI	Mean	0.6429	0.4433	0.4391	0.4356	0.4307	0.4308	0.4305	**0.4284**	0.4270	0.4241	0.4263
	± CI 95 %	0.0299	0.0052	0.0043	0.0038	0.0032	0.0034	0.0030	**0.0031**	0.0046	0.0034	0.0039
	Std	0.1051	0.0183	0.0151	0.0135	0.0113	0.0118	0.0104	**0.0110**	0.0161	0.0118	0.0138
DEV	Mean	254.23	221.92	220.65	220.06	219.37	219.43	**218.91**	218.96	219.15	218.54	219.02
	± CI 95 %	1.30	0.91	0.76	0.73	0.81	0.71	**0.62**	0.54	0.76	0.68	0.67
	Std	4.57	3.20	2.67	2.56	2.86	2.50	**2.16**	1.89	2.67	2.41	2.36
CONN	Mean	408.69	407.85	411.79	407.36	410.61	406.85	407.93	405.39	410.50	408.08	**405.30**
	± CI 95 %	4.45	3.94	4.84	4.87	4.62	4.91	4.20	4.51	4.62	5.02	**4.50**
	Std	15.66	13.85	17.02	17.15	16.26	17.29	14.77	15.85	16.24	17.67	**15.84**

Table 5. M parameter tuning

		1	**10**	20	30	40	50	60	70	80	90	**100**
CDI	Mean	0.6467	0.4421	0.4413	0.4389	0.4394	0.4336	0.4322	0.4318	0.4355	0.4308	**0.4301**
	± CI 95 %	0.0256	0.0049	0.0069	0.0046	0.0063	0.0035	0.0031	0.0038	0.0118	0.0028	**0.0031**
	Std	0.0903	0.0172	0.0242	0.0162	0.0221	0.0122	0.0109	0.0133	0.0415	0.0098	**0.0109**
DEV	Mean	254.01	**219.87**	220.55	222.47	223.52	224.63	226.12	226.72	228.18	228.74	230.92
	± CI 95 %	0.93	**0.96**	0.78	0.62	0.58	0.71	0.61	0.72	0.61	0.68	0.51
	Std	3.27	**3.37**	2.75	2.17	2.04	2.49	2.14	2.53	2.14	2.38	1.81
CONN	Mean	412.11	409.20	405.81	412.30	409.56	407.45	403.38	405.00	404.02	405.35	**401.71**
	± CI 95 %	3.94	5.60	4.00	4.28	5.22	4.54	5.31	4.94	5.21	5.64	**5.46**
	Std	13.85	19.69	14.06	15.06	18.38	15.98	18.68	17.40	18.33	19.86	**19.20**

Table 6. e_w parameter tuning

		1	**10**	**20**	30	40	50	60	70	80	90	100
CDI	Mean	0.4391	**0.4241**	0.4306	0.4444	0.4312	0.4318	0.4389	0.4335	0.4323	0.4353	0.4256
	± CI 95 %	0.0050	**0.0065**	0.0089	0.0138	0.0098	0.0090	0.0143	0.0094	0.0107	0.0120	0.0084
	Std	0.0175	**0.0229**	0.0313	0.0486	0.0345	0.0315	0.0502	0.0330	0.0378	0.0421	0.0296
DEV	Mean	**220.31**	221.75	223.69	225.83	227.51	229.67	231.25	229.39	233.84	231.03	235.23
	± CI 95 %	**0.75**	0.69	1.32	1.26	1.29	1.67	1.75	1.59	1.98	1.70	1.79
	Std	**2.63**	2.42	4.66	4.44	4.55	5.89	6.14	5.59	6.98	5.98	6.29
CONN	Mean	410.78	404.16	**380.52**	382.48	385.19	383.30	386.55	385.80	390.34	386.74	391.53
	± CI 95 %	4.74	4.33	**3.50**	3.99	5.24	4.46	4.13	4.30	5.46	3.62	4.45
	Std	16.69	15.22	**12.33**	14.05	18.45	15.70	14.52	15.12	19.22	12.73	15.65

Table 7. Summary of the parameter tuning.

CDI:	$\theta = 0.97$,	$\alpha = 1$,	$\gamma = 70$,	$M = 100$,	$e_w = 10$
DEV:	$\theta = 0.95$,	$\alpha = 0$,	$\gamma = 60$,	$M = 10$,	$e_w = 11$
CONN:	$\theta = 1$,	$\alpha = 1$,	$\gamma = 50$,	$M = 100$,	$e_w = 20$

5.2 LPS-ACO Algorithm Application

To assess the performance of our LPS-ACO algorithm, we compare our results with two well-known classical clustering algorithms and two recently developed heuristic algorithms. The former two algorithms are the classical k-means [15] algorithm and the hierarchical agglomerative clustering single-linkage [16]. These two classical algorithms were selected because of their intrinsic nature of optimizing the DEV (in the case of k-means) and CONN (in the case of single-linkage) metrics.

The first heuristic algorithm was the Electrical Pattern Ant Colony Clustering (EPACC) [1][3]. This algorithm uses some principles of the ACO algorithm to guide the construction phase. Nevertheless, the main drawback of this algorithm is the lack of a pheromone trail that captures the information related to the metric under optimization. This makes the algorithm less flexible to move from one metric to another.

The second heuristic used for comparison was created by a slight modification of the EPACC algorithm. To explain this, it is important to recall that the original EPACC algorithm [1] uses a roulette wheel selection process for each assignation of p_i to c_i. The roulette wheel selection implies that a random number is generated in each assignation. In our EPACC modification, we use a principle called stochastic universal sampling (SUS) for the creation of the random number. Our algorithm called EPACC-SUS is identical to the original EPACC, with the only difference that a unique random number is created for all the assignations of p_i to c_j, rather that a particular random number for each p_i.

[3] The set of parameters used for the EPACC algorithm were those reported as the best set of parameters in [1].

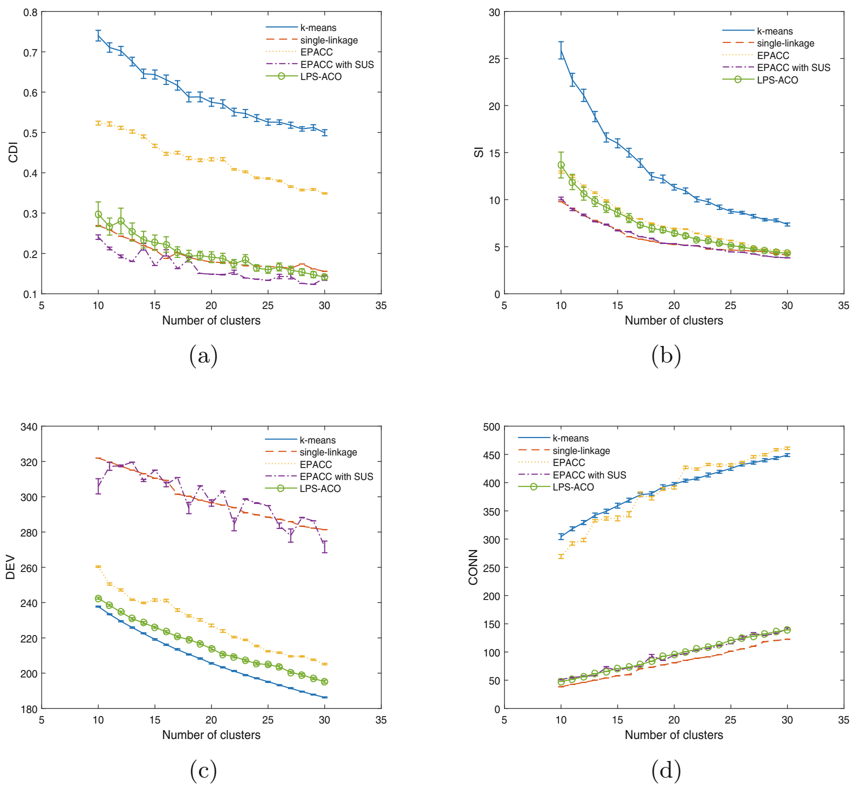

Fig. 3. Comparison of algorithms using different metrics. (a) CDI metric, (b) SI metric, (c) DEV metric, (d) CONN metric.

This simple modification results in an exceptional performance related to CDI and CONN metrics as shown in the results.

The algorithms were run 50 times varying the number of clusters K from 10 to 30. The results correspond to the mean value of those 50 test. The 95 % CI were also calculated and plotted for each point of the results.

Figure 3 shows the comparison of the tested algorithms under different scenarios. The first thing to remark is that k-means present a poor performance related to CDI, SI and CONN metrics (as shown in Fig. 3(a), (b), (d)). This poor performance could mislead the user to the conclusion that k-means is a bad algorithm for the LPS clustering problem. However, it can be observed in Fig. 3(c) that k-means present the best performance among all the algorithms. The opposite case occurs with the single-linkage and the EPACC-SUS algorithms. Both of them present the best performance related to CDI, SI and CONN metrics (as shown in Fig. 3(a), (b), (d)) but the worse performance related to DEV metric (shown in Fig. 3(c)).

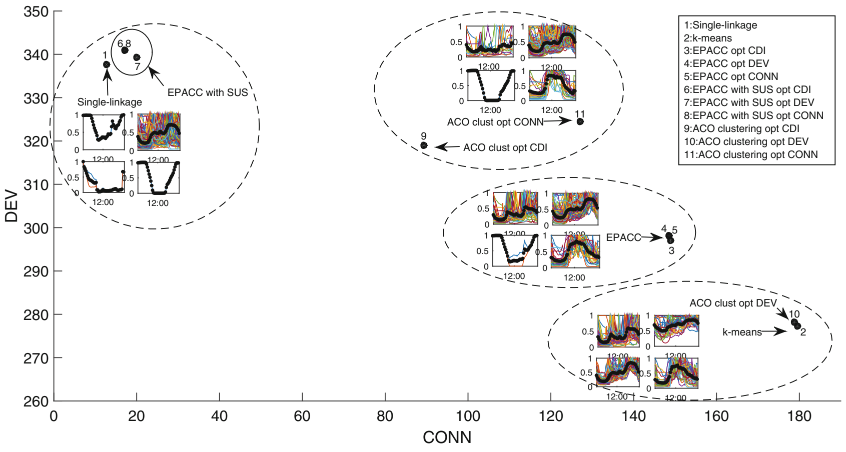

Fig. 4. Comparison of algorithms based on DEV and CONN metrics

This particular behavior when optimizing a single metric open a question on the validity of the chosen metric to measure the quality of the LPS clustering. For instance, observing just Fig. 3(a), (d), k-means and EPACC could be considered as bad algorithms for the LPS clustering problem. That conclusion completely changes by observing Fig. 3(c), in which those two algorithms present a good performance (in fact, k-means is the best algorithms when DEV metric is used to evaluate the performance of the algorithms) related to DEV metric. Overall, our LPS-ACO algorithm present a good performance for all the analyzed metrics without being the best in any case.

To go deep in the analysis of a proper LPS clustering, we developed an experiment using the same data set P but decreasing the number of clusters to $K = 4$ for an easy visualization of the patterns in each cluster. Again, each algorithm was run 50 times, and the mean values were stored. Moreover, EPACC, EPACC-SUS, and LPS-ACO algorithms were run three times each changing the optimized metric (i.e., CDI, DEV and CONN). SI metric was not taken into account since its behavior is similar to CDI.

Figure 4 shows the mean values of the 50 runs of each algorithm related to DEV and CONN metrics. Also, some graphical clusterings (corresponding to one of the solutions found) are shown in some areas of the plot to observe the kind of solutions generated by each algorithm. The first thing to recall is that single-linkage and k-means are on the opposite sides of the plots and represent the best solution for CONN and DEV metrics respectively (notice that this is a minimization problem). Moreover, it can be observed (on the top left of the figure) that single-linkage generates solutions that classified single patterns as a group and let one unique cluster with the rest of the patterns. This kind of classification will optimize CDI, SI or CONN metric, but the validity of such classification can be questioned when we analyze graphically the formed clusters. On the opposite side, we can visualize the kind of solutions provided by an algorithm like k-means on the bottom right of the figure. These solutions seem wrong from the perspective of metrics like CDI or CONN, but forms compact and well-defined clusters optimizing DEV metric. Nevertheless, such algorithms might fail in the identification of atypical patterns.

After those observations, it is logical to think that the correct solution (from the perspective of external knowledge) can be situated in the Pareto front between this two optimal solutions (represented by metrics DEV and CONN). For that reason, an algorithm that can generate optimal solutions considering the trade-off between these two metrics, such as our LPS-ACO algorithm, should be able to access to more proper clusterings solutions for the LPS problem.

6 Conclusion and Future Work

In this paper, we propose the application of Ant Colony Optimization (ACO) to the electrical LPS clustering problem. We analyze the performance of the algorithm searching for the optimization of different metrics proposed in the literature. As in most heuristics, ACO performance depends on good parameter settings. We present a careful analysis of the impact of ACO parameters in the quality of the solutions. Additionally, we compare our LPS-ACO algorithm with some classical clustering algorithms, such as k-means and single-linkage, and other heuristics, such as EPACC and EPACC-SUS. The results have shown that the selection of a proper metric to validate the performance of an algorithm plays a key role determining the effectiveness of a clustering algorithm. For instance, depending on the chosen metric for evaluating the algorithms, we can wrongly rank a clustering algorithm as good or bad if the selected metric is not suitable for the problem under analysis. For that reason, it is important to consider different metrics to measure the performance of an algorithm. Overall, the results obtained by our LPS-ACO algorithm are promising and promote further study to improve the ACO performance. As future work, we will examine the design of a proper metric that reflects good partitions in the particular domain of the electrical LPS clustering problem. We are also interested in the design of a multi-objective ACO approach that allows us to move in all the Pareto front or to determine the proper number of cluster automatically for the electrical LPS problem.

References

1. Chicco, G., Ionel, O.M., Porumb, R.: Electrical load pattern grouping based on centroid model with ant colony clustering. IEEE Trans. Power Syst. **28**(2), 1706–1715 (2013)
2. Das, S., Abraham, A., Konar, A.: Automatic clustering using an improved differential evolution algorithm. IEEE Trans. Syst. Man Cybern. Part A: Syst. Hum. **38**(1), 218–237 (2008)
3. Chicco, G., Napoli, R., Piglione, F.: Comparisons among clustering techniques for electricity customer classification. IEEE Trans. Power Syst. **21**(2), 933–940 (2006)

4. Figueiredo, V., Rodrigues, F., Vale, Z., Gouveia, J.: An electric energy consumer characterization framework based on data mining techniques. IEEE Trans. Power Syst. **20**(2), 596–602 (2005)
5. Chicco, G., Ionel, O.M., Porumb, R.: Formation of load pattern clusters exploiting ant colony clustering principles. In: IEEE EUROCON, pp. 1460–1467 (2013)
6. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. Part B: Cybern. **26**(1), 29–41 (1996)
7. Gerbec, D., Gasperic, S., Smon, I., Gubina, F.: Allocation of the load profiles to consumers using probabilistic neural networks. IEEE Trans. Power Syst. **20**(2), 548–555 (2005)
8. Chaouch, M.: Clustering-based improvement of nonparametric functional time series forecasting: application to intra-day household-level load curves. IEEE Trans. Smart Grid **5**(1), 411–419 (2014)
9. Handl, J., Knowles, J.: An evolutionary approach to multiobjective clustering. IEEE Trans. Evol. Comput. **11**(1), 56–76 (2007)
10. Li, M., Ming-ming, S.: An improved ant colony clustering algorithm based on dynamic neighborhood. In: IEEE International Conference on Intelligent Computing and Intelligent Systems, vol. 1, pp. 730–734 (2010)
11. Dunn, J.C.: Well separated clusters and optimal fuzzy partitions. J. Cybern. **4**, 95–104 (1974)
12. Davies, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. **1**(2), 224–227 (1979)
13. Chou, C.H., Su, M.C., Lai, E.: A new cluster validity measure and its application to image compression. Pattern Anal. Appl. **7**(2), 205–220 (2004)
14. Blum, C., Dorigo, M.: The hyper-cube framework for ant colony optimization. IEEE Trans. Syst. Man Cybern. Part B Cybern. **34**(2), 1161–1172 (2004)
15. Lloyd, S.: Least squares quantization in pcm. IEEE Trans. Inf. Theory **28**(2), 129–137 (1982)
16. Day, W.H., Edelsbrunner, H.: Efficient algorithms for agglomerative hierarchical clustering methods. J. Classif. **1**(1), 7–24 (1984)