

This is an electronic reprint of the original article.

Author(s): Lähderanta, Tero; Lovén, Lauri; Ruha, Leena; Leppänen, Teemu; Launonen, Ilkka; Riekk, Jukka; Sillanpää, Mikko J.

Title: Capacitated spatial clustering with multiple constraints and attributes

Year: 2024

Version: Publisher's version

License: CC BY

Please cite the original version:

Lähderanta, T., Lovén, L., Ruha, L., Leppänen, T., Launonen, I., Riekk, J. & J. Sillanpää, Mikko J. (2024). Capacitated spatial clustering with multiple constraints and attributes. *Engineering Applications of Artificial Intelligence*, 127, Part A, 107182.
<https://doi.org/10.1016/j.engappai.2023.107182>



Capacitated spatial clustering with multiple constraints and attributes

Tero Lähderanta^{a,*}, Lauri Lovén^{b,c}, Leena Ruha^d, Teemu Leppänen^e, Ilkka Launonen^a,
Jukka Riekkilä^b, Mikko J. Sillanpää^a

^a Research Unit of Mathematical Sciences, University of Oulu, P.O. BOX 8000, Oulu, FI-90014, Finland

^b Center for Ubiquitous Computing, University of Oulu, Oulu, Finland

^c Distributed Systems Group, TU Wien, Vienna, Austria

^d Natural Resources Institute Finland, Oulu, Finland

^e Oulu University of Applied Sciences, Oulu, Finland

ARTICLE INFO

Dataset link: <http://sguangwang.com/TelecomDataset.html>

Keywords:

Distributed networks
Constrained programming
Dual clustering
Location-allocation

ABSTRACT

Capacitated spatial clustering, a type of unsupervised machine learning method, is often used to tackle problems in compressing data, classification, logistic optimization and infrastructure optimization. Depending on the application at hand, a multitude of extensions to the clustering problem may be necessary. In this article, we propose a number of novel extensions to PACK, a recent capacitated partitional spatial clustering method which uses an optimization algorithm that is based on linear programming tasks. These extensions relate to the relocation and location preference of cluster centers, outliers, and non-spatial attributes, and they can be considered jointly. In the context of edge server placement, these improve the spatial location of servers while considering, for example, application placement on the servers in response to spatial application usage patterns. We demonstrate the usefulness of an extended version of PACK with an example with simulated data, as well as a real world example in edge server placement for a city region with various different setups. These setups are evaluated with summary statistics about spatial proximity and attribute similarity. As a result, the similarity of the clusters was improved by 53% at best while simultaneously the proximity degraded only by 18%. The extensions provide valuable means for including non-spatial information in the cluster analysis, and to attain better overall proximity and similarity.

1. Introduction

Clustering, one of the most important tasks and techniques in data mining and unsupervised machine learning, refers to the unsupervised classification of patterns into groups. Its primary goals include pre-processing, compressing and classifying the data, and gaining further insight into it (Celebi, 2014; Patel and Thakral, 2016; Grubésic et al., 2014).

This work focuses on partitional clustering, also referred to as non-hierarchical clustering. Partitional clustering aims to partition a data set into non-overlapping subsets such that each data point is in exactly one subset and each cluster can be represented by a single point, referred to as a prototype of cluster (Jin and Han, 2010; Xiao and Yu, 2012). Partitional clustering is especially useful in applications which use the partitions for further analyses. As an example, direct marketing campaigns often start with segmenting customers into groups (see further examples in the work of Banerjee and Ghosh, 2006).

Further, clustering often involves a spatial dimension with geographic information related to the studied phenomenon. Such a setting,

referred to as spatial clustering, requires an appropriate and meaningful treatment of space, spatial relationships, and the attributes of locations (Grubésic et al., 2014). We focus in particular on partitional spatial clustering, where points of interest are partitioned into disjoint clusters. These points contain geographical information of an event and thus they can be seen as spatial point events in the region of interest (Gatrell et al., 1996). Partitional spatial clustering is used, e.g. for the spatial analysis of Internet of Things (IoT) sensor data (Lee and Lee, 2015) or clustering of web user sessions (Sisodia et al., 2016; Sisodia, 2017). Partitioning sensor data into local clusters can help in finding local features of the observed phenomena, and in distributing the computational burden of the data analytics, especially in the case local or edge-based computing capacity is available (Yi et al., 2015; Xu et al., 2017; Lovén et al., 2019, 2021). Traditional partitional clustering algorithms, such as K-means (Zhao et al., 2018) or CLARANS (Ng and Han, 2002), and more recent methods such as nonnegative matrix factorization (Peng et al., 2021b) and subspace clustering methods (Peng

* Corresponding author.

E-mail address: tero.lahderanta@oulu.fi (T. Lähderanta).

et al., 2021a), have been successfully applied to such partitioning problems in the field of spatial analysis. However, these general methods do not consider the special features and constraints that often appear in partitioning problems, especially in location-allocation (Hale and Moberg, 2003).

In this article, we present a novel capacitated partitioning clustering approach that jointly considers constraints related to outliers, non-spatial attributes and the locations of the cluster centers. The main focus of this method is to partition the data into a set of clusters based on some application-specific metric and constraints, rather than provide a density-based approach to clustering. An essential part of the method is its connection to location-allocation problems, which are a set of spatial optimization problems (Hale and Moberg, 2003). The term capacitated clustering refers to a cluster analysis where the sizes of the clusters, that is, the number of data points or the total weight value in a cluster, are either constrained or penalized (Mulvey and Beck, 1984). Constraints for center placement correspond to a case where the location of some cluster heads are known. To elaborate in more detail, instead of knowing the location of some centers exactly, there may be more or less vague prior information about their locations. On the other hand, if some center locations are fixed, it may be possible to relocate such centers under certain circumstances. Contrary to related works, we consider all these constraints jointly with capacitated clustering.

Outlier points, on the other hand, may distort solutions to the clustering problem. Thus it may be beneficial to allow some points to be assigned as outliers, i.e. not assigned to any cluster. However, to our knowledge no previous work considers outliers jointly with capacitated clustering.

Finally, the spatial distribution of points might not be the only determinative factor in a clustering problem. One example is found in customer segmentation where customers are divided into clusters not only based on their spatial location, but also the total number of their purchases and their geodemographic lifestyle class (Brimicombe, 2007). Furthermore, these types of non-spatial attributes can take into account the possible temporal aspects of the spatial point events (Ansari et al., 2020). Again, to the best of our knowledge, no related work considers such non-spatial attributes in conjunction with capacitated spatial clustering or the other extensions we presented above.

We demonstrate the feasibility and importance of the joint consideration of the above extensions with a simulated data experiment as well as with an experiment in edge computing. Edge computing is recognized as a paradigm shift from centralized, cloud-based processing towards distributed computing near the sources of the data (Satyanarayanan, 2017). A key part of the upcoming 5G and beyond-5G mobile networks among others, edge computing promises more bandwidth, lower latency, and improved privacy through virtualization of cloud-based applications into the network infrastructure, close to the end user devices. However, distributing application components across the communication networks introduces challenges in the management of the computing resources across the potentially massive-scale edge environment where individual resource nodes may appear or disappear at arbitrary times.

In the first experiment, we simulate a data set related to a typical spatial clustering problem. The experiment considers non-spatial attributes, outliers and capacity limits simultaneously. It provides a controlled setup to introduce the above mentioned extension to the clustering analysis as well as to illustrate the balancing between spatial and non-spatial attributes.

Our second experiment considers edge server placement (Lähderanta et al., 2021; Lovén et al., 2020). While topical in its direct connection to edge computing, edge server placement also provides a suitable experimental setup for evaluating the multi-objective nature of our proposed method. Indeed, edge server placement requires simultaneously fulfilling a number of requirements. First, edge servers need to be placed into close proximity of the users and access points, thus minimizing latencies. Second, the allocation of user workloads

should consider the capacities of edge servers, ideally ensuring a balanced workload division between the servers. Third, placement may require the consideration of preferred locations (e.g. airports or shopping centres), outliers (e.g. access points whose workload can be readily uploaded to cloud), and non-spatial attributes (e.g. application usage profile or the type of user equipment).

We use the Shanghai Telecom data set (Guo et al., 2019; Wang et al., 2019; Xu et al., 2019) in our server placement evaluation. The data set contains over 7.2 million sessions from mobile phones to 2732 base stations in the city of Shanghai over a six-month period. In the field of server placement these extensions have not yet been widely considered, since only a few papers address preferred location or outlier extensions (Leyva-Pupo et al., 2019; Wang et al., 2019).

We also provide an extensive, open-source toolkit for conducting the proposed analyses.¹ We refer to the proposed approach with acronym PACK (PlAcement with Capacitated K-family²). We have proposed earlier versions of PACK, in our previous articles, with less features, for placing edge servers in the city of Oulu (Lähderanta et al., 2021; Lovén et al., 2020) and maternity hospitals in Finland (Huotari et al., 2020). However, compared with these earlier versions, we make a number of important enhancements. In short, our contributions are as follows:

1. We consider novel extensions for center placement in spatial clustering, namely relocating fixed centers and a preference for certain locations over the others.
2. We consider novel extensions for capacitated spatial clustering, namely outliers that might distort the clustering structure, and non-spatial attributes of the points of interest.
3. We examine the effects of the above extensions and their combinations in edge computing server placement with a real-world data set of a mobile network in the city of Shanghai, China.

The rest of the article is organized as follows. In Section 2 we take a detailed look on the existing literature on capacitated clustering problems. In Section 3 we present the PACK method and its extensions. In Section 5 we demonstrate PACK with an experiment in edge server placement with multiple setups. In Section 6 we highlight and discuss the main findings of our evaluations.

2. Related work

We consider related work along the axes of center placement, balanced and constrained clustering, outliers, and non-spatial attributes.

Center placement. Sometimes the locations of cluster centers need to be constrained, often to coincide with some of the data points. Such location-constrained algorithms are referred to as *actual point-prototype-based clustering algorithms* (Xiao and Yu, 2012). One of the most popular of such algorithms is the *partitioning around medoids* (PAM) algorithm (Kaufman, 1990).

Further, Rahman and Smith (2000), among others, consider clustering where some centers have fixed locations, such as the existing health service infrastructure in a city. In location-allocation literature relocating fixed centers that result in a sub-optimal partition is referred to as a facility location relocation problem (see e.g. the work of Farahani et al., 2009). In practice, the relocation of fixed centers can be seen as moving existing facilities to a more suitable location, which depends on the cost of the relocation.

However, we could neither find a study observing that fixed centers may result in a sub-optimal partition, nor one offering ways to relocate some of the fixed centers to produce better results.

Finally, we assigned a location preference to some demand points, indicating they should have a center close to them (Lähderanta et al.,

¹ <https://github.com/terolahderanta/rpack>

² Here, k-family refers to the family of different variants of K-mean, i.e. K-mean, K-medoid, and K-median.

2021). However, we did not yet consider that location preference can also be introduced as a soft constraint for cluster centers. Similarly, Leyva-Pupo et al. (2019) addressed location preference by assigning point-wise requirements for the latency from certain points to the cluster center.

Capacity constraints. Clustering is often used for dissecting data into separate groupings for further processing or analysis. Such cases often control sizes of clusters, such that too large or too small clusters are undesirable. These types of clustering methods can be divided into two categories called balance constrained and balance driven approaches (Malinen and Fränti, 2014). Here we investigate the related work on balance constrained clustering.

In balance constrained clustering, the balance between cluster sizes is of first importance, and the minimization of distances is only a secondary criterion. This is typically achieved by introducing an additional hard constraint to the capacity of a cluster. Several approaches have been proposed (Banerjee and Ghosh, 2006; Malinen and Fränti, 2014; Elliott, 2011). Hu et al. (2018) constrained only the minimum size of clusters, and both Zhu et al. (2010) and Ganganath et al. (2014) allowed predefined size constraints to vary between clusters. Li et al. (2018) and Liu et al. (2017) studied balance-driven clustering with a soft constraint that penalizes large deviations in cluster sizes, and Gupta (2017) recently provided survey on balanced data clustering.

Hard capacity constraints have also been proposed in the clustering framework by Mulvey and Beck (1984), who assumed for the weights and coefficients that $w_i = 1$ and $a_i = 1$ for all i , respectively, and referred to the problem as the *capacitated clustering problem* (CCP). Later, Negreiros and Palhano (2006) used squared Euclidean distances in a continuous setting and referred to the problem as *capacitated centered clustering problem* (CCCP). Another possibility for combining balance and capacity constraints is to apply both a lower and an upper bound for the capacity (Borgwardt et al., 2017). Grossi et al. (2016) considered how hard capacity constraints can be modeled together with several other constraints and optimization criteria, such as the selection of the number of clusters. However, no previous studies have considered capacitated spatial clustering with conjunction with the extensions presented in Section 3.

Constrained clustering. In this work we impose constraints on cluster sizes in the form of capacity limits along with constraints on the placement of cluster centers. However, constrained clustering in general (see e.g. Basu et al., 2008) is a broader topic that has merited studies of their own.

For example, Dao et al. (2017) proposed a constrained programming model which allows the integration of different kinds of instance-level or cluster-level user constraints with a given optimization criterion of choice, such as minimizing the maximal diameter of clusters, maximizing the minimal split between clusters, minimizing the within-cluster sum of dissimilarities, or minimizing the within-cluster sum of squares. The authors then also considered the case of bi-criterion optimization constrained clustering with a Pareto optimal solution.

Recently, Nghiem et al. (2020) also studied post-processing of constrained or unconstrained clustering algorithm results as a combinatorial optimization problem while enforcing all constraints a-posteriori and introducing new ones related to cluster overlap, neighborhoods, property-cardinalities and attribute levels. This integration of existent and new constraints was implemented in an integer linear program and demonstrated with large datasets such as MNIST.

Outliers. If a point does not seem to belong to any cluster, a forced assignment may distort information and the interpretation of the cluster it is assigned to (Tseng, 2007). In location-allocation problems, this could produce unnatural partitions that are unwanted and sub-optimal in terms of minimizing the distances. It may thus be necessary to allow a point not to be a member of any center. In partitional clustering, such points that are left unassigned are referred to as outliers. Further, a partition where every point is assigned is referred to as complete

clustering, and a partition where some points may be left unassigned is referred to as partial clustering (Steinbach et al., 2019).

In this work, we define an outlier as a remote data point such that the distance to a cluster center, measured as a weighted combination of spatial and non-spatial distances, is greater than a predefined threshold value. Outliers could be removed prior to the clustering by using some outlier detection method. However, the clustering structure could be highly dependent on the outliers, and on the other hand, the determination of the outliers depends on the clustering structure. The identification of the outliers should thus not be a separate step, but an integral part of the clustering process (Liu et al., 2019).

Outliers can be identified by determining their number prior to analysis and then iteratively assigning the most remote points as outliers. Such an approach has been proposed by, e.g., Chawla and Gionis (2013). Whang et al. (2015) allowed overlap between clusters by combining outlier analysis with a K-means algorithm where a fixed number of points is assigned to at least one cluster and a fixed number is not assigned at all. Cygan and Kociumaka (2014) considered a capacitated k-center problem with outliers, where the algorithm chooses the number of outliers via dynamic programming.

A third alternative is to formulate the problem as a bi-criterion optimization task, where a fixed penalty is paid for each outlier point (Tseng, 2007). Thus, the outliers can be assigned to a $k + 1$ 'th cluster by adding a penalty to the loss function. Tseng (2007) further gives this bi-criterion approach a model-based interpretation, pointing out that the penalty term corresponds to assuming that outlier points emerge from a homogeneous Poisson process.

Alternatively, outliers can be identified by keeping track of the distances of points to their nearest center. Olukanmi and Twala (2017) determined the related threshold value iteratively. Wang et al. (2007) determined an upper bound for the distance to each center, which is used in the decision of number of centers to be placed.

None of the related work, however, consider outliers in conjunction with spatial capacitated clustering.

Non-spatial attributes. In many real world scenarios it is necessary to consider also non-spatial or non-geographical attributes. Such a two domain approach is referred to as *dual clustering* (Lin et al., 2005).

Dual clustering methods can be divided into two subcategories: the hybrid distance measure approach, and the separate clustering operations approach (Zhu et al., 2020). In the first approach, the spatial and non-spatial attributes are included with a distance measure which consists of both a spatial and a non-spatial similarity measure component, each weighted depending on the problem in question. Lin et al. (2005) further developed the Interleaved clustering-classification algorithm with a hybrid-distance measure, with weight $w \in [0, 1]$ for the spatial attributes and weight $1 - w$ for the non-spatial attributes. Zhang et al. (2007) aim to obtain non-overlapping clusters by embedding a penalized spatial distance measure (PSD). They compare PSD to the standard 2-dimensional Euclidean distance with only spatial attributes and to the 3-dimensional Euclidean distance with both spatial and non-spatial attributes. Again, however, we did not find any related work that considers non-spatial attributes jointly with capacitated clustering.

PACK. We have proposed earlier versions of our PACK approach (see Section 3) with less extensions (Lähderanta et al., 2021; Lovén et al., 2020). The newest version of PACK includes many of the approaches presented in the related work and, most importantly, PACK allows the use of multiple extensions simultaneously, a feature which none of the previous studies has implemented. A detailed comparison of server placement algorithms with respect to their properties has been made by Lähderanta et al. (2021). To highlight the novelty of this article, we summarize the extensions introduced in the different versions in Table 1.

3. The PACK method

We propose a capacitated spatial clustering method which introduces a number of novel extensions, while covering a wide range of extensions earlier considered only separately. We refer to the proposed approach as PACK (PlAcement with Capacitated K-family).

Table 1

The extensions included in different versions of the PACK algorithm. Extensions newly introduced in each version are highlighted with green color.

Feature		PACK 1	PACK 2	PACK 3
Center	Fixed centers	–	✓	✓
	Released centers	–	–	✓
Location preference	Point-based	✓	✓	✓
	Center-based	–	–	✓
Non-spatial attributes		–	–	✓
Capacity limits		✓	✓	✓
Membership	Hard	✓	✓	✓
	Fractional	✓	✓	✓
	Overlapping	✓	✓	✓
Outliers		–	–	✓

PACK 1: Lähderanta et al. (2021).

PACK 2: Lovén et al. (2020).

PACK 3: This paper.

3.1. Problem formulation

Table 2 lists the symbols used. Our aim is to minimize distance $d(\cdot, \cdot)$ between all candidate center locations c_j^* and data point locations x_i^* , while taking account the total weights of the points w_i' and the additional penalties regarding outliers and released centers. As a result, we obtain the memberships y_{ij} from each data point i to each center j . PACK minimizes the following objective function

$$\underset{c_j^*, y_{ij}, A}{\operatorname{argmin}} \underbrace{\sum_{i=1}^n \sum_{j=1}^k w_i' d(x_i^*, c_j^*) y_{ij}}_{\text{distance}} + \underbrace{\lambda_o \sum_{i=1}^n w_i' y_{ik+1}}_{\text{outliers}} + \underbrace{\lambda_f t}_{\text{released centers}} \quad (1)$$

with the following constraints:

- | | |
|-------------------------|--|
| 1. Center locations | $c_j \in \{p_1, p_2, \dots, p_s\}, \quad \forall j,$ |
| 2. Fixed centers | $c_l = f_l, \quad l \in \{1, \dots, m\} \setminus A,$ |
| 3. Membership | $y_{ij} \in \{0, 1\}, \quad \forall i, j$ |
| 4. Membership | $\sum_{j=1}^k y_{ij} = 1 \quad \forall i,$ |
| 5. Capacity constraints | $L \leq \sum_{i=1}^n a_i y_{ij} \leq U \quad \forall j.$ |

The values of the input parameters are first scaled via the so-called z-scores (see Vesanto, 2001).

Weights and location preference. PACK allows weighted data points and enables the inclusion of preferred cluster center locations. These preferred locations are incorporated into the problem by adding a parameter $\gamma_i > 0$ to the weight of the preferred points w_i i.e. the total weight $w_i' = w_i + \gamma_i$. The larger the parameter γ_i , the more strongly the point i attracts a center (Ackerman et al., 2012).

Fixed and released centers. Applying constraint 2, $0 \leq m \leq k$ centers can be assigned to have predetermined fixed locations f_1, \dots, f_m , meaning that the locations of m centers are decided before the optimization. Further, we propose a relaxation where a fixed center at a preassigned location can be released and relocated if a penalty λ_f is paid. Thus, if t fixed centers are released, the cost is $\lambda_f t$. A center is released if the release decreases the value of the objective function by more than λ_f .

Distance metrics. PACK is agnostic towards the distance metric used. The objective function (1) admits an arbitrary distance measure d . For example standard Euclidean distance and squared Euclidean distance can be used.

Distance function d takes four arguments: $\{x_i, \theta_i\}$ and $\{c_j, \theta_j\}$, which correspond to the spatial coordinates and the non-spatial attributes of point i and center j . Furthermore, in a dual clustering scenario a hybrid distance measure can be applied in PACK. This is typically done by dividing the distance measure into a sum of two distances: $d(x_i^*, c_j^*) = d(\{x_i, \theta_i\}, \{c_j, \theta_j\}) = \lambda_d d_1(x_i, c_j) + (1 - \lambda_d) d_2(\theta_i, \theta_j)$, where λ_d is the weight given to the spatial distance measure d_1 and $(1 - \lambda_d)$ is the weight given to the non-spatial distance measure d_2 .

Algorithm 1 PACK-algorithm

Input: $x_i^*, w_i', k, N_{init}, \lambda_o, \lambda_f, f_j, j = 1, \dots, m$

Output: locations of the centers c_j^* and allocations of points to centers $y_{ij}^*, j = 1, \dots, k + 1$

```

1: for  $l = 1$  to  $N_{init}$  do
2:   Generate initial value for  $c_j^*, j = 1, 2, \dots, k$  using K-means++
3:   repeat
4:     Allocation-step: minimize (1) with respect to  $y_{ij}$  with branch-
       and-cut algorithm
5:     Location-step: minimize (1) with respect to  $c_j^*$  by computing
       the medoid
6:      $S =$  the value of the objective function
7:   until  $c_j^*$  does not change
8:   if  $S < S_{min}$  or  $l = 1$  then
9:      $S_{min} \leftarrow S$ 
10:     $c_j' \leftarrow c_j^*$ 
11:     $y_{ij}' \leftarrow y_{ij}$ 
12:   end if
13: end for
14: return  $c_j', y_{ij}'$ 

```

Capacity. Both lower and upper capacity constraints can be applied to control the amount of weight assigned to a center. Depending on the upper and lower limits, this constraint can be used for both balancing the data and constraining only either too large or too small clusters.

Further, PACK allows the capacity constraint and the loss function to have different constants (a_i and w_i , respectively). This allows, for example, using a location preference for the objective function but not for the capacity limits.

Outliers. PACK identifies outliers by applying the bi-criterion approach (Tseng, 2007; Charikar et al., 2001). Instead of assigning a constant outlier penalty λ_o for each point, we use a penalty that is proportional to the weight of the point, $\lambda_o w_i'$. As a result, the capacity limits permitting, a point is assigned as an outlier if $d(x_i^*, c_j^*) > \lambda_o$. In other words point i is assigned as an outlier if the outlier penalty is greater than the cost of allocation, increasing the value of the objective function by a minimum of $\lambda_o w_i'$. Hence, the selection depends only on the total distance to the clusters and not on the weight of the point. Note that this distance measure is a weighted combination of the spatial and non-spatial distances.

3.2. The block coordinate descent algorithm

PACK employs a block coordinate descent algorithm (Tseng, 2001) for minimizing function (1) with a fixed number of centers (Algorithm 1). The optimization steps in Algorithm 1 are then linear, which enables the use of a wide variety of mixed integer linear programming (MILP) optimization methods, providing faster and solvable computation even with a high number of points. The parameters λ_o, λ_f and f_j are considered as inputs to the algorithm. In this work we assume that the values of the parameters are decided by the experts in the application in question. However, one option is to use for example a cross-validation technique to determine those values, see Arlot and Celisse (2010). On each iteration, a block coordinate descent algorithm minimizes the objective function with respect to a block of variables while holding other blocks of variables fixed at the values obtained in the previous iteration step (Wright, 2015). The two main steps, i.e., the blocks, for the algorithm are the *allocation-step* (line 4), where the points are assigned to the centers, and the *location-step* (line 5), where the centers are relocated based on the points assigned to them. The steps are iterated until convergence is reached.

Table 2

The symbols utilized in PACK.

	Symbol	Range	Description
Inputs	$x_i, i = 1, \dots, n$	\mathbb{R}^2	the spatial coordinates of point i
	$\theta_i, i = 1, \dots, n$	\mathbb{R}^Q	non-spatial attributes of point i
	$x_i^* = \{x_i, \theta_i\}, i = 1, \dots, n$	\mathbb{R}^{2+Q}	the location and the non-spatial attributes of point i
	Q	\mathbb{N}	dimension of non-spatial attributes
	w_i	$[0, \infty[$	weight of point i
	$d(\cdot, \cdot)$	$[0, \infty[$	distance between two locations
	γ_i	$[0, \infty[$	location preference of point i
	$w'_i = \gamma_i + w_i$	$[0, \infty[$	weight of point i , corrected by its location preference
	$p_h, h = 1, \dots, s$	\mathbb{R}^2	a possible location for a center
	m	\mathbb{N}	the number of centers with fixed location
	$f_o, o = 1, \dots, m$	\mathbb{R}^2	location of a fixed center
	λ_d	$[0, 1]$	trade-off between spatial and non-spatial distance measures
	λ_o	$[0, \infty[$	the penalty parameter of outliers
	λ_f	$[0, \infty[$	the penalty parameter for releasing a fixed center
	L	$[0, \infty[$	the lower capacity limit
	U	$[L, \infty[$	the upper capacity limit
	a_i	$[0, \infty[$	the weight used in the capacity constraints (typically $a_i = w_i$)
Outputs	A	$\{\mathbb{R}^2\}$	set of released centers
	$t = A $	\mathbb{N}	number of released centers
	$c_j, j = 1, \dots, k$	\mathbb{R}^2	the spatial coordinates of center j
	$c_j^* = \{c_j, \theta_j\}, j = 1, \dots, k$	\mathbb{R}^{2+Q}	the location and the non-spatial attributes of center j
	y_{ij}	$[0, 1]$	membership of point i to center j

Further, as block-coordinate descent finds the local minima close to the initial values, PACK uses a number of initial values to find the global minimum. The initial values are obtained using the K-means ++-algorithm (Arthur and Vassilvitskii, 2007) that spreads the initial locations of centers improving both the speed and the accuracy of the K-means method.

Allocation step. The allocation step minimizes the objective function (1) with respect to y_{ij} . Constraints 3, 4 and 5 are applied, and the locations of the centers c_j^* are assumed to be fixed. If no capacity constraints are applied, this step corresponds to assigning each point to the nearest facility measured in terms of the chosen distance metric. In addition a point is assigned as an outlier if its distance to the nearest center exceeds the limit λ_o . On the other hand, if capacity constraints are applied, this step is an NP-hard integer programming task (Papadimitriou, 1981). If the constraint 3 is changed to $y_{ij} \in [0, 1]$, an approach referred to as fractional membership by Lähderanta et al. (2021), the task is solved in polynomial time (Khachiyan, 1980).

Generally, the integer programming problem in the allocation step can be solved with the branch-and-cut method, which is a general method for a variety of MILP problems that can guarantee the global optimum of an individual allocation-step (Mitchell, 2002). However, other optimization methods are also applicable.

Location step. Location step minimizes the objective function (1) with respect to c_j^* s, while keeping the allocations y_{ij} fixed. In other words, each center is separately relocated given the points assigned to it. This step is omitted for fixed centers. In continuous space with the squared Euclidean distance metric, c_j^* is the weighted mean of the points allocated to cluster j . In the discrete setting, c_j^* is the point which minimizes the sum of pairwise distances among points assigned to c_j^* .

If the release of fixed centers is allowed, a center is released and relocated if the sum of distances to the assigned points is reduced by more than λ_f . Similarly, a previously released fixed center can be returned to its original location if the increase in the sum of distances is lower than the release penalty λ_f .

Implementation. The algorithm is implemented as an open source R package called *tpack* (Lähderanta et al., 2019) available on GitHub.

The allocation step is done with a branch-and-cut algorithm (Mitchell, 2002) on Gurobi (Gurobi Optimization, 2018), a fast optimizer package with R bindings freely available for academic use. If Gurobi is not available, PACK uses the lpSolve-package for R (Berkelaar et al., 2015) in the allocation step.

4. Evaluation with simulated data

In this section, we introduce the properties of PACK extensions with a controlled setup. First we generate multiple random data sets, where the original clustering is known and consequently we can analyze the effectiveness of the algorithm with the selected parameters.

4.1. Data generation

Data points are simulated from a mixture of 10 gamma distributions with varying shape (0 to 15) and scale (0 to 100) parameters. The spatial coordinates of these data points are within a $[0, 1] \times [0, 1]$ space. The weights of the data points are simulated from a uniform distribution such that the weights are between 1 and 100. Furthermore, we generate 20 data points that are considered as outliers by sampling uniformly from the region of data points. For each data set, we generate unique parameters. In total, we simulate 100 data sets each with 520 observations.

Additionally, three non-spatial attributes are simulated for each point by first dividing the data points into four regions, generating the border lines from uniform distribution, see Fig. 1. Secondly, the non-spatial attributes for each data point are simulated from a multivariate normal distribution with different parameters based on the point's section, see Table 3. We refer to these attributes as the Red, Green and Blue. Please note that even though the parameters of the distributions of non-spatial attributes depend on the location of the point, the distributions of non-spatial values for each region overlap with each other, reducing the spatial dependence (see Fig. 2). Examples of dividing the data and the distribution of non-spatial attributes can

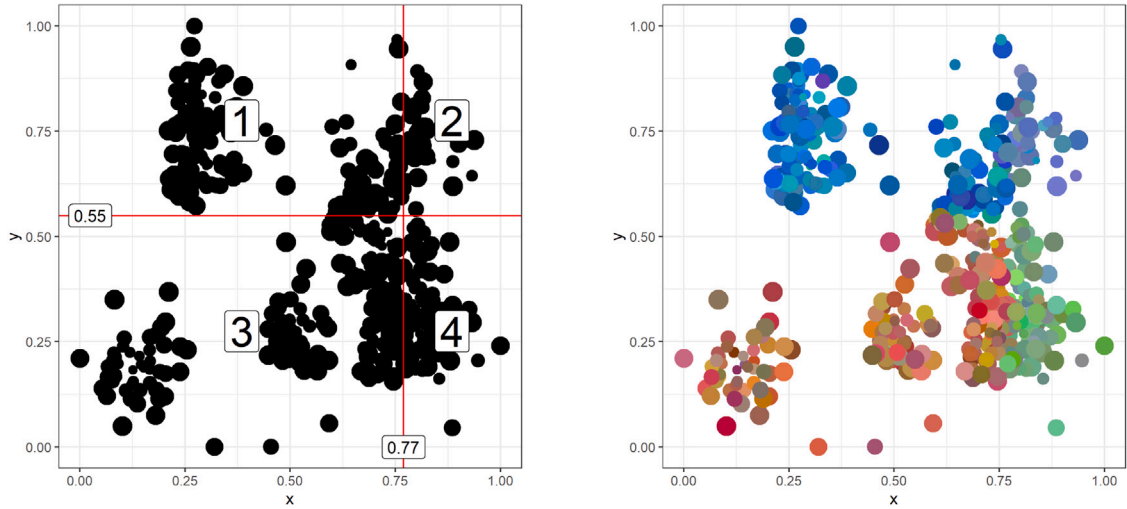


Fig. 1. Example of a single simulated data set. Points correspond to data points originated from cluster structure and the outlier points. Size of the point reflects the weight of the data point. **Left panel:** The data points are divided into four regions by the randomly generated two red lines from a uniform distribution between 0 and 1. These regions have a unique distribution for the non-spatial attributes. **Right panel:** The values of the three non-spatial attributes are coded into single RGB colors (i.e., with red, green, and blue channels), showing the difference of values depending on the regions.

Table 3

Parameters for the multivariate normal distributions, used in the simulation of non-spatial attributes ([Red, Green, Blue] respectively).

Region	Mean	Covariance
1	$\begin{bmatrix} 1 & 5 & 7 \end{bmatrix}^T$	$\text{diag}(1, 2, 2)$
2	$\begin{bmatrix} 4 & 5 & 6 \end{bmatrix}^T$	$\text{diag}(1, \frac{1}{2}, 1)$
3	$\begin{bmatrix} 8 & 4 & 1 \end{bmatrix}^T$	$\text{diag}(2, 2, 2)$
4	$\begin{bmatrix} 5 & 7 & 3 \end{bmatrix}^T$	$\text{diag}(1, 1, 2)$

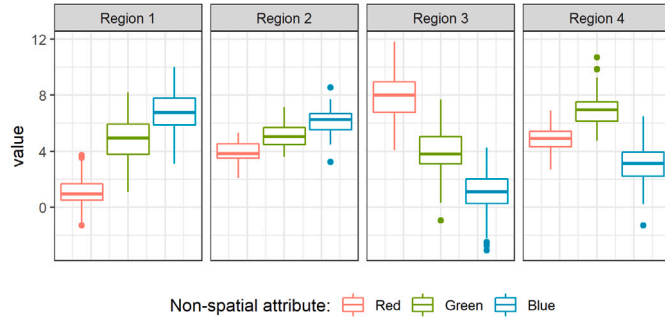


Fig. 2. Example of a single simulated data set. Boxplots correspond to the distribution of non-spatial attributes (Red, Green and Blue) in each region.

be seen in Figs. 1 and 2. The simulation codes with toy examples in R can be found on GitHub³ (Lähderanta et al., 2019).

4.2. Analysis with a single data set

In this section, we perform the clustering on a single data set generated with the simulation procedure in Section 4.1. The simulated data set is presented in Fig. 1 and the distribution of the non-spatial attributes is presented in Fig. 2. In the clustering, we assume the number of clusters to be known, such that $k = 10$, and we utilize upper and lower size limits for each cluster, such that the sizes must be between $[\sum_i w'_i - 100k, \sum_i w'_i + 100k]$. We utilize standard Euclidean distance as the distance function for both spatial and non-spatial attributes as they represent practical distance, for example in kilometers in many

location-allocation problems. Furthermore, we demonstrate clustering results with various values for the other clustering parameters:

- λ_d is set to 0.1, 0.2, ..., 0.9, 1.0
- λ_o is set to 0.05, 0.1, 1

The summary statistics of the clustering are shown in Fig. 3 and the plotted clusters are shown in Fig. 5. As the distances are scaled into the interval $[0, 1]$, the outliers are decided based on the maximum distance in the data set, such that for example when $\lambda_o = 0.05$, then a data point is marked as an outlier if the distance to its nearest cluster center is greater than 5% of the maximum distance. In the data set, we can observe that the mean distance to its center improves as we increase the value of λ_d , that is the spatial distance is weighted more. Contrarily, the average standard deviations of the non-spatial attributes within clusters raise linearly. Furthermore, as can be seen in Fig. 5, with the lower values for λ_d , the clusters follow the generated region borders more closely (dashed lines), and with $\lambda_d = 0.2$ the overlap between clusters is evident. With this example data set, no clear conclusions can be made with the elbow method when choosing the optimal balance between mean distance and the deviations of non-spatial attributes. One candidate value for the optimal value of λ_d would be $\lambda_d = 0.7$, as the decrease in mean distance is minimal with greater values.

The impact of the outlier parameter λ_o can be seen in Fig. 3, as the mean distance is generally lower with smaller values of λ_o . This is illustrated in the clusterings in Fig. 5, where the number of outlier points (marked as crosses) increases as we decrease the value of λ_o , thus the mean distance from a non-outlier data point to its center is lower. Furthermore, in the clusterings with a low value of λ_d , the found outliers can be located spatially near the cluster center, see Fig. 5, clustering with $\lambda_o = 0.2$ and $\lambda_d = 0.2$. This is due to the fact that the outlier detection relies on the total distance between points, including both spatial and non-spatial distance measures. Similarly, in Fig. 4 illustrates that the similarity of clustering to the original clustering increases when the value of λ_d increases. For further reading on Jaccard index, see for example (Tan et al., 2016).

4.3. Analysis with 100 simulated data sets

In this section, we analyze the 100 simulated data sets with different clustering parameter, similarly to the previous section.

The main summary statistics of each clustering with the 100 data sets are presented in Fig. 6. Naturally, as the λ_d decreases, that is

³ <https://github.com/terolahderanta/rpack>

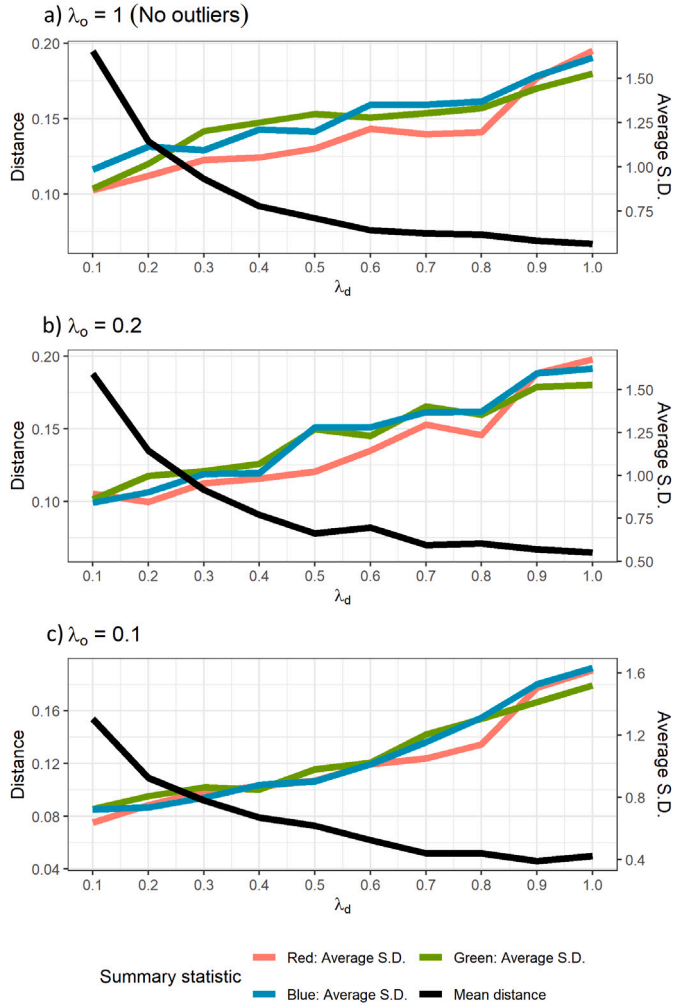


Fig. 3. The summary of clustering results with the single simulated data set, showing the average standard deviation (S.D.) of each non-spatial attribute (Red, Green and Blue) within a cluster and the mean spatial distance from each point to its cluster center. The clusterings have varying values for λ_d and λ_o .

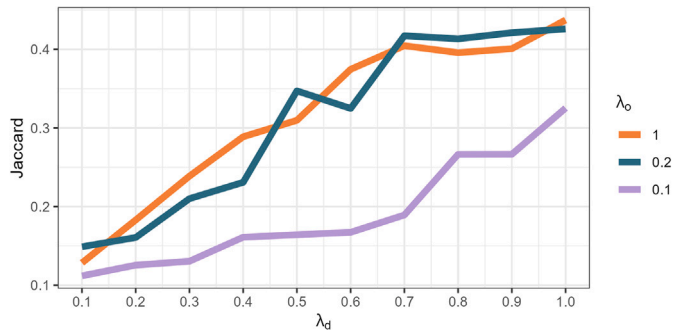


Fig. 4. The Jaccard index of each clustering with single simulated data set. The clusterings have varying values for λ_d (x axis) and λ_o (color).

the non-spatial attributes are weighted more, the mean distance is increased. At the same time, the average standard deviation of a non-spatial attribute on a cluster decreases. This is an indication of similarity of non-spatial attributes in a cluster. Furthermore, the impact of the additional outlier penalty is subtle, providing small improvements to the mean distance (Fig. 6).

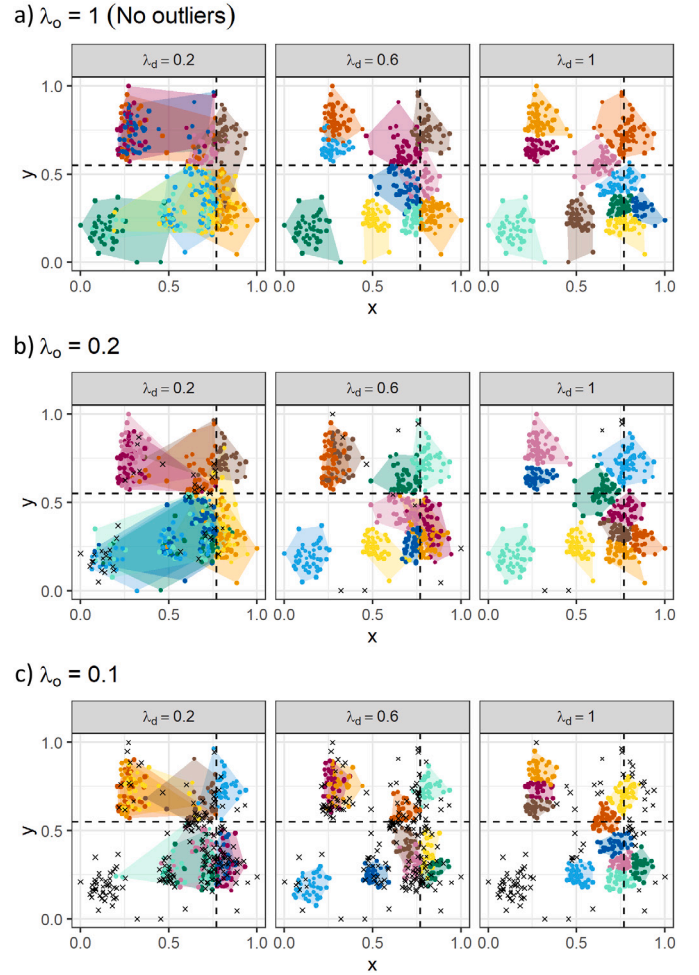


Fig. 5. The capacitated clustering results with the single simulated data set. Image (a) represents scenario without outliers, (b) with some outliers and (c) with a high number of outliers. Colors represent the cluster areas and the data points in that cluster while crosses denote found outliers. Dashed lines represent the generated non-spatial region borders. Here, we display three different values of λ_d and three values for λ_o .

5. Evaluation with edge server placement

We demonstrate the feasibility and the practical benefits of PACK with an experiment focusing on the placement of edge servers in the city center of Shanghai, China (Guo et al., 2019; Wang et al., 2019; Xu et al., 2019). The Shanghai Telecom data set contains the locations of 2732 base stations in the Shanghai region and the mobile user connections to those base stations in a six-month period. Spatial distribution of the base stations and their workloads are shown in Fig. 7. We demonstrate the use of capacity constraints, non-spatial attributes, fixed server location, releasing fixed servers, location preference and outliers.

The experiment optimizes the telecommunications infrastructure in Shanghai, placing *edge computing servers* (ES) to reduce the communication latency experienced by the mobile phone users. Edge computing (Shi et al., 2016) refers to computing infrastructure that facilitates data processing for user applications directly at the points of interest, at one hop distance from the users' equipment. Deploying edge infrastructure, characteristics such as backbone network topology, wireless network coverage and users' applications requirements, largely dictate the physical placement options for the servers (Lähderanta et al., 2021).

ES placement is an *offline* problem, as opposed to *online* problems such as offloading tasks to cloud or migrating virtual machines between

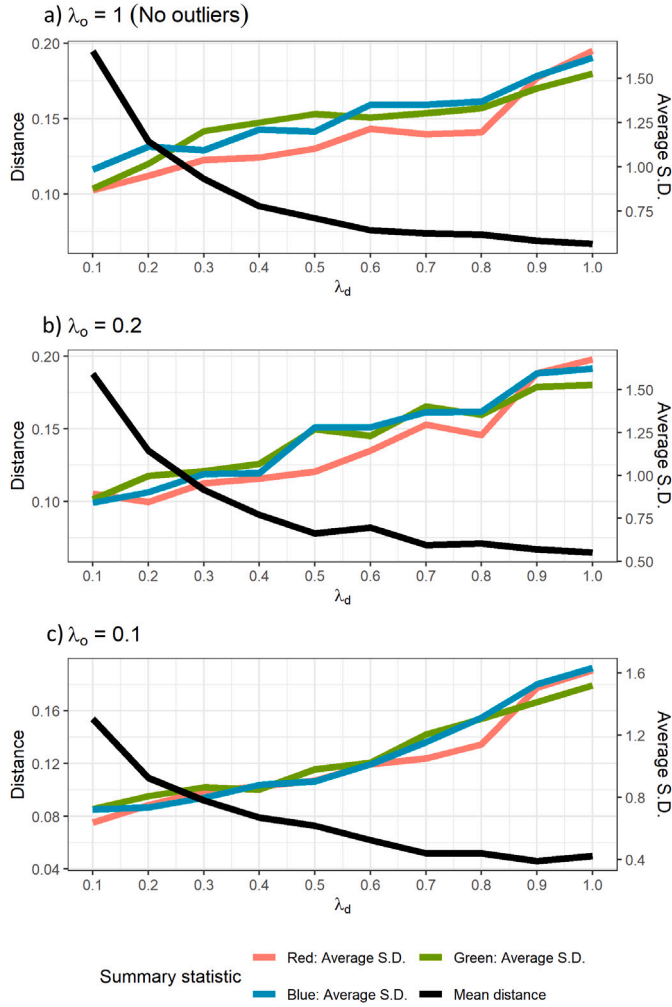


Fig. 6. The summary of clustering results with the 100 simulated data sets, showing the average standard deviation (S.D.) of each non-spatial attribute (Red, Green and Blue) within a cluster and the mean spatial distance from each point to its cluster center. The clusterings have (a) no outliers, (b) some outliers and (c) a high number of outliers with varying values for λ_d .

ES's. The goal of ES placement is to deploy a number of edge computing servers in the city region such that the latencies (i.e. distances) between the servers and the base stations assigned to each server are minimized. Each base station is assigned to exactly one ES, and the workload (i.e., weight) of the base station, that is, its maximum number of concurrent users in the recorded data, is allocated to that server.

Each ES, represented by a cluster center, has a limited computing capacity for the workload it can handle, represented by the sum of the weights of the base station locations in the corresponding cluster. Further, each edge server can only be placed at the same location as one of the base stations. Since the topology of the backbone network is unknown, we use geospatial distances to approximate the latencies between the base stations, as discussed in Lähderanta et al. (2021).

ES placement provides an ideal setting for studying the PACK extensions, as these extensions have clear real-world interpretations. We explore the effect of extensions in two scenarios:

1. Adding the non-spatial attribute *average session length* to the placement problem as an additional parameter and the inclusion of outliers.
2. Joint consideration of (1) fixed server locations, with ten servers already deployed or having preferred locations, and (2) including the average session length in the placement.

In the evaluations, we assume a budget for the number of servers that could be placed to be between 35 and 45 servers. To determine the optimal number of servers in terms of server costs and minimizing latency, baseline clustering was performed with varying values for k and the value $k = 38$ was decided with the elbow method (Yuan and Yang, 2019). Furthermore, we apply capacity limits for each server to guarantee a reasonable distribution of workload in the scenarios.

We use the average session duration as a proxy to end-user application usage profiles, which were unavailable in the open data set. Bohmer et al. list distinctly varying usage durations for different mobile applications, with e.g. an average of 68.11 s for news applications, and 114.25 s for game usage (Böhmer et al., 2011). Clustering similar application usage patterns for individual edge servers helps in, for example, caching and pre-loading of popular content to that ES, greatly reducing user latency and communication burden on the core network.

Each base station has spatial coordinates in longitudes and latitudes x_i , and an average session length θ_i , which we then combine into the vector $\{x_i, \theta_i\}$. We study the impact of average session duration by implementing a hybrid distance function d which is a sum of the spatial and non-spatial components: $d(\{x_i, \theta_i\}, \{c_j, \theta_j\}) = \lambda_d d_1(x_i, c_j) + (1 - \lambda_d) d_2(\theta_i, \theta_j)$. In our evaluations, we assign d_1 as the squared Euclidean distance function, and d_2 as the squared difference between average session lengths. Squared distance is chosen as it contributes towards better worst-case latency in the deployment (Lähderanta et al., 2021). In the evaluations, we test four different weights λ_d : 0.9, 0.99, 0.999 and 1. $\lambda_d = 1$ corresponds to a placement without any non-spatial attributes (Lähderanta et al., 2021). We scale the distances to the range [0, 1].

Each clustering is evaluated with regard to the spatial proximity between ESs (i.e., cluster centers) and base stations, and the session length similarity among the ESs. In the evaluations, proximity is defined as the mean of the squared Euclidean distance between an ES and its base stations, and similarity is defined as the standard deviation of session duration on the base stations. We explore the means of weighted distances between servers and base stations, and the standard deviations of session lengths in each server.

The whole list of setups from both scenarios studied in this paper can be seen on Table 4.

5.1. Scenario 1: Clustering with non-spatial attribute and outlier feature

In this section we analyze the impact of four different weights λ_d for the spatial distance measure and the impact of outliers. In the scenario we have in total eight different setups for the placement, corresponding to varying intensities of the weight value λ_d for dual clustering, and the consideration of outliers (see Table 5). Furthermore, we compare these setups to standard K-means and K-medoid methods (Rajarajeswari and Ravindran, 2015) with three dimensions (spatial location and the non-spatial attribute).

In ES placement, outlier base stations may not be served by an ES, but instead they can be connected straight to cloud. While this increases the communication latency of those outlier significantly, it improves the overall quality of service of the system. For the outlier penalty, we set $\lambda_o = 0.05$. As discussed in Section 3.1, this corresponds to a distance threshold to the nearest cluster center. In practice, if the distance from a base station to its ES exceeds 5% of the maximum distance in the data set, then the base station is marked as an outlier. Note that in the algorithm the distances are scaled to [0, 1].

In Fig. 8 we have selected four setups for closer inspection. The impact of the weight value λ_d for dual clustering is clearly seen as spatially overlapping server regions: As we increase the weight value of the non-spatial distance measure, that is, the value of λ_d decreases, the resulting server regions overlap with each other more. This can be seen in the bottom left panel of Fig. 8, where for example the pink and purple regions on the south-east have isolated points in each other regions.

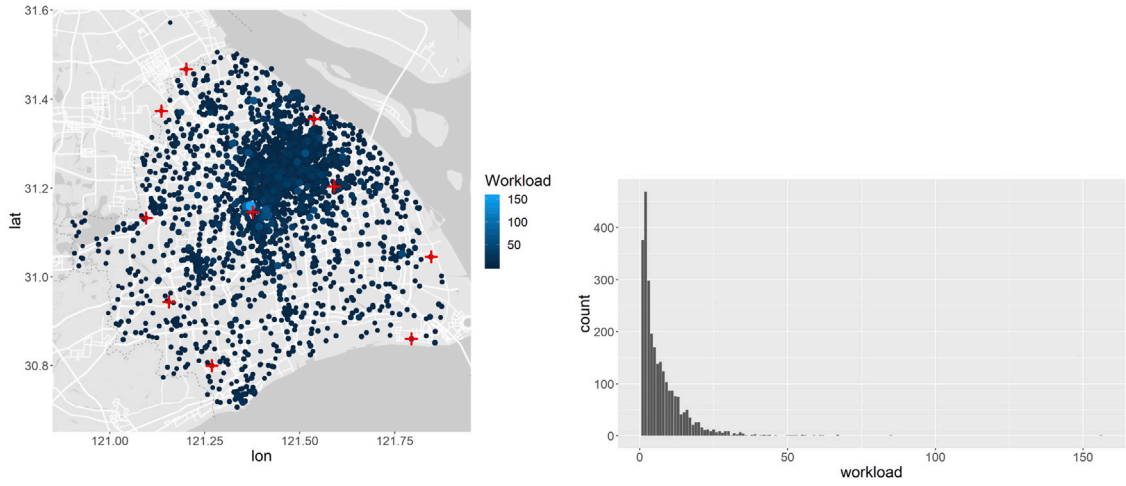


Fig. 7. Base stations in Shanghai area. Left panel: the locations of base stations, color representing the workload and the red crosses the 10 fixed edge server locations. Right panel: the workload distribution of the base stations.

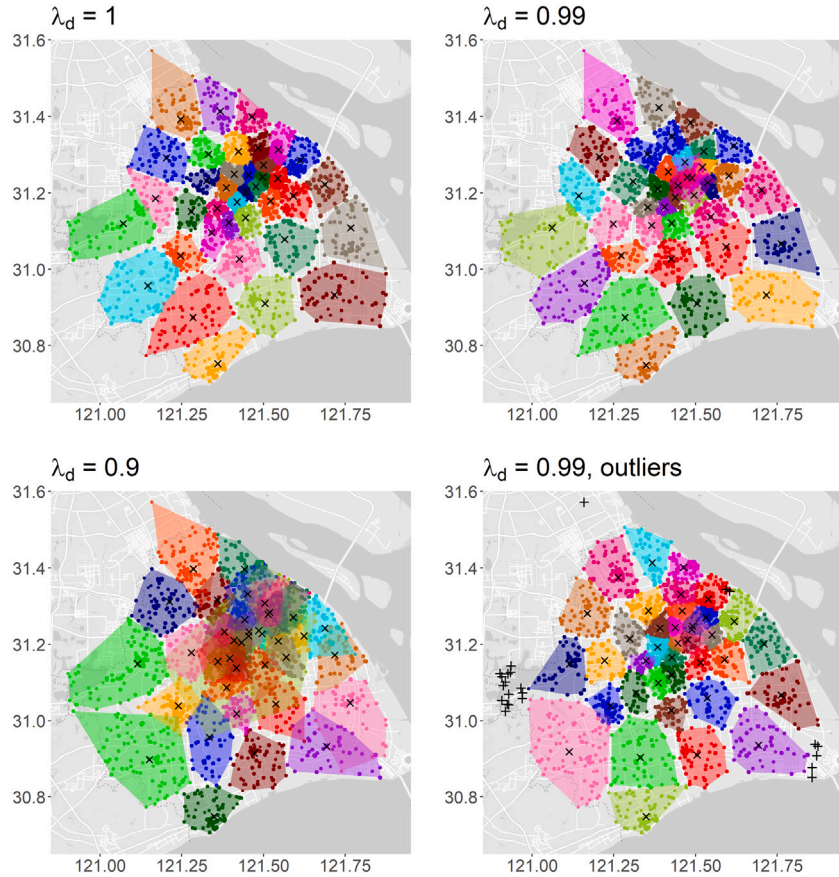


Fig. 8. The optimal placement of 38 edge servers with selected four setups. Colors represent a set of base stations that are covered by a single edge server, the edge servers are marked as crosses and outliers as plus-signs. The lower the λ_d , the more the regions overlap spatially.

Meanwhile, without outliers, the mean of the server-wise session duration standard deviations is cut down by 45% when compared to the placement without non-spatial attributes (Table 5). On the other hand, spatial proximity worsens by 35%. With outliers, the reduction in the standard deviation is as high as 53%, while proximity degrades only by 18%. This observation is further detailed in Fig. 9, where a trend in the standard deviations can be seen. When λ_d is decreased, the session lengths in each cluster are more similar.

Including outliers improves the placement in each setup when compared to not allowing outliers. Mean distances to server are slightly

smaller in each setup, and the standard deviation of the session length has decreased. With $\lambda_d = 0.99$, both proximity and similarity are improved, by 2% and 13% respectively (Table 5).

When comparing to standard clustering methods, K-means and K-medoid, we observe that these approaches do not provide reasonable results in the server placement scenario. In the K-medoid setup, the spatial proximity somewhat improves and session similarity is only 1% worse, workload balance is significantly worse (45%). As the K-means do not constraint the locations of the servers, session similarity is much better than the baseline setup (61%). However, a price is paid for the

Table 4
Placement setups for evaluations in Shanghai, China.

Setup	Centers	λ_d	Outliers	Explanation
BASE	–	1	No	Clustering with capacity constraints but without non-spatial attributes or outliers (Lähderanta et al., 2021).
DUAL1	–	0.999	No	Dual clustering (i.e. considering both spatial and non-spatial attributes) with minimal weight on session duration.
DUAL2	–	0.99	No	Dual clustering with medium weight on session duration.
DUAL3	–	0.9	No	Dual clustering with high weight on session duration.
OUT	–	1	Yes	Outliers, No dual clustering.
DUAL1-OUT	–	0.999	Yes	Outliers, dual clustering with minimal weight on session duration.
DUAL2-OUT	–	0.99	Yes	Outliers, dual clustering with medium weight on session duration.
DUAL3-OUT	–	0.9	Yes	Outliers, dual clustering with high weight on session duration.
KMEANS	–	1	No	K-means method with three dimensions.
KMEDOID	–	1	No	K-medoid method with three dimensions.
FIXED	Fixed	1	No	Same as BASE, but 10 edge servers have fixed locations (Lovén et al., 2020).
FIXED-DUAL	Fixed	0.9	No	As above, but with dual clustering with medium weight on session duration.
FIXED-DUAL-OUT	Fixed	0.9	Yes	As above, but with outliers and dual clustering with medium weight on session duration.
REL	Release	1	No	10 fixed servers can be released with given cost λ_d on the objective function.
REL-DUAL	Release	0.9	No	As above, but with dual clustering with medium weight on session duration.
REL-DUAL-OUT	Release	0.9	Yes	As above, but with outliers and dual clustering with medium weight on session duration.
PREF	Preference	1	No	A location preference $\gamma_i = 200$ for the ten fixed server location candidates. This value is larger than the largest weight (i.e. 156) in the data.
PREF-DUAL	Preference	0.9	No	As above, but with dual clustering with medium weight on session duration.
PREF-DUAL-OUT	Preference	0.9	Yes	As above, but with outliers and dual clustering with medium weight on session duration.

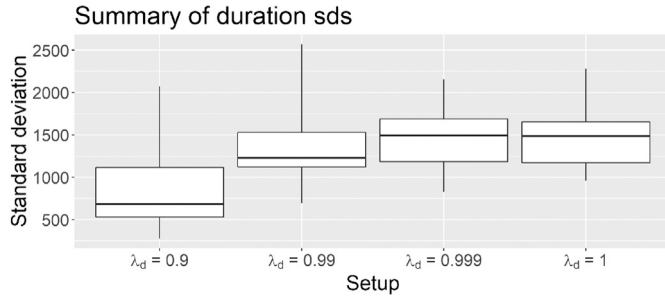


Fig. 9. Distribution of standard deviations in each setup without outliers.

improvement, as the spatial proximity and workload balance suffer greatly (53% and 191%, respectively).

5.2. Scenario 2: Placement with fixed server locations

In this scenario, we place a total of 38 edge servers in which session length is taken into account, with ten of the server locations fixed in four alternative ways, corresponding to the following setups:

1. No fixed cluster center locations. This setup corresponds to the baseline placement case studied by Lähderanta et al. (2021) and is identical to the setup in previous section with $\lambda_d = 1$.

2. The locations of the ten edge servers are fixed. This corresponds to the deployment of 28 servers, with ten servers already deployed. This setup corresponds to the method studied by Lovén et al. (2020).
3. There is a release cost ($\lambda_f = 0.02$) to the ten servers such that if the benefit (i.e., reduction in the objective function (1)) of relocating any of those servers is greater than the release cost, then that server is relocated.
4. A location preference is set for the ten fixed server location candidates. Those locations are considered more important and attract the servers more than the other locations. Location preference in edge server placement would correspond to a situation where, for example, mobile connectivity in a certain area would be above that of the surrounding areas and especially good at the center of that area. Such an area, and especially its center, would thus be a preferred location for an edge server. We set the value of the additional weight to $\gamma_i = 200$ for each preferred server location $i = 1, \dots, 10$. This value refers to the number of users in a base station and is larger than the largest weight value (i.e. 156) in the data set.

Furthermore, we incorporate the session length attribute and the possibility of outliers in the placement. In this scenario a weight value for spatial attributes $\lambda_d = 0.99$ is used with each setup mentioned above, so that in total of nine setups are evaluated.

Table 5

Summary statistics for Scenarios. Improvements are marked in green, while degradations are in red, compared to baseline. See Table 4 for setup explanations.

	Setup	Spatial proximity (distance to ES)		Session similarity (SD. of session duration)		Workload balance (SD. of ES workloads)		Outliers (% of total)
		Mean	Diff.	SD	Diff.	SD	Diff.	
Scenario 1	BASE	3.31	(baseline)	1474	(baseline)	131.10	(baseline)	–
	DUAL1	3.29	–1%	1476	0%	129.20	–1%	–
	DUAL2	3.36	+2%	1334	–9%	138.60	+5%	–
	DUAL3	4.02	+34%	809	–45%	129.30	–1%	–
	OUT	3.25	–2%	1500	+2%	131.00	0	1.8
	DUAL1-OUT	3.24	–2%	1474	0%	123.30	–6%	0.8
	DUAL2-OUT	3.24	–2%	1280	–13%	130.40	–1%	1.9
	DUAL3-OUT	3.89	+18%	698	–53%	123.60	–6%	4.7
	KMEANS	5.07	+53%	576	–61%	380.90	+191%	–
Scenario 2	KMEDOID	3.24	–2%	1485	+1%	249.20	+47%	–
	FIXED	3.70	(baseline)	1480	(baseline)	132.30	(baseline)	–
	FIXED-DUAL	3.79	+2%	1228	–17%	128.90	–3%	–
	FIXED-DUAL-OUT	3.77	+2%	1287	–13%	132.40	0	2.0
	REL	3.46	–6%	1485	0%	131.70	0	–
	REL-DUAL	3.55	–4%	1280	–14%	137.20	+4%	–
	REL-DUAL-OUT	3.52	–6%	1310	–11%	131.90	0	1.6
	BASE	3.31	(baseline)	1474	(baseline)	131.10	(baseline)	–
	PREF	3.28	–1%	1483	+1%	127.50	–3%	–
	PREF-DUAL	3.36	+2%	1273	–14%	133.10	+2%	–
	PREF-DUAL-OUT	3.36	+2%	1274	–14%	121.90	–8%	1.3

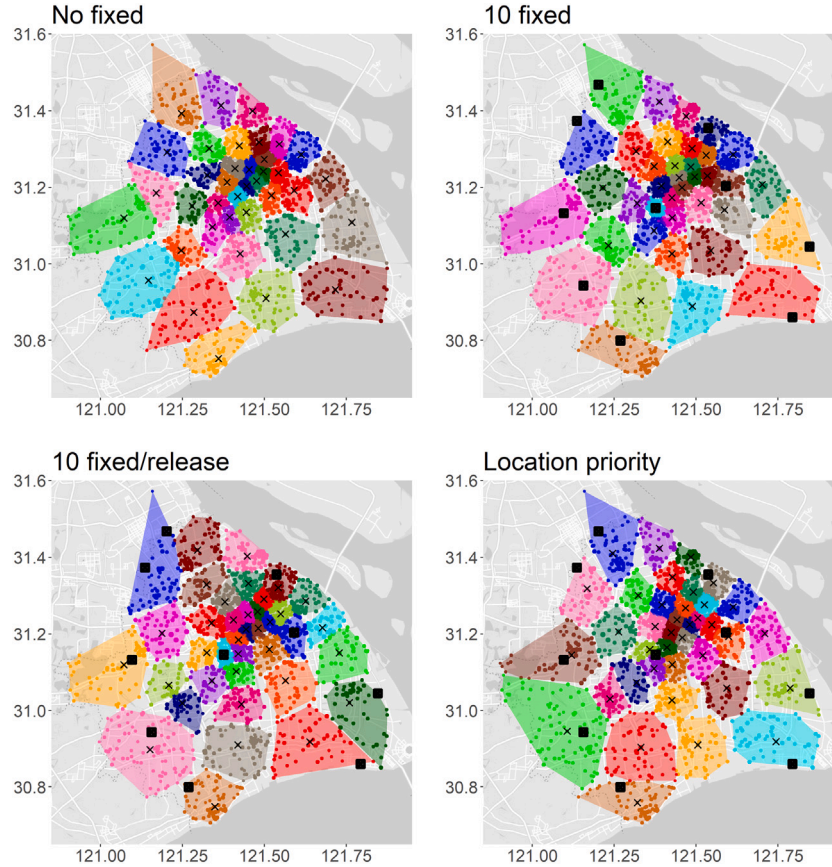


Fig. 10. The optimal placement of 38 edge servers with selected four setups. Colors represent a set of base stations that are covered by a single edge server, the edge servers are marked as crosses and the fixed servers as squares.

Fig. 10 compares four selected setups without non-spatial attributes. Overall, each setup produces deployment of servers with some important differences. First, the benefit of relocating the fixed servers is clear: some of the fixed servers in the south-west and in the south-east are located on the edge of the map which overall worsens the average distance to the center, that is, latency to the server.

Table 5 further details the changes in proximity and similarity. Compared with a baseline method (Lovén et al., 2020), fixed servers with the non-spatial attributes indeed improves similarity by 17% while slightly (–2%) degrading in proximity. Just allowing the releasing of fixed servers, on the other hand, only improves on proximity (+6%). Whereas, jointly considering both the non-spatial attributes as well as

the releasing of fixed servers improves proximity by 4% and similarity by 14%. On the other hand, the center-based location preference, compared to a baseline method (Lähderanta et al., 2021), affects proximity only by little. Including the non-spatial attributes, however, improves similarity by 14% while reducing proximity just slightly (−2%).

The inclusion of outliers provided a minor improvement to the spatial proximity as well as to the workload balance, when compared to the setups without outlier detection (Table 5). However, session similarity worsened in all of these setups.

Overall, average latency is decreased as we allow the servers to be relocated. The setup with location preference can be seen as a “relaxed” version of the setup with release cost on pre-determined center locations, as it pulls the servers closer to the predetermined locations rather than forcing the servers to be located exactly at the predetermined locations. Furthermore, the mean and median distances in setups with location preference are much smaller than in the other setups. When it comes to setups with average session length, the non-spatial attribute improved session similarity as expected, while still maintaining reasonable spatial proximity in each setup.

All four setups produce clusters which appear feasible, considering domain knowledge. The differences in the resulting clusters are mainly due to the locations of the predetermined spots: forcing the servers to the fixed locations can cause worse average latency in the deployment, which can be solved by allowing servers at the fixed locations to relocate.

6. Discussion and conclusion

In this article, we introduced new extensions to the PACK method for capacitated clustering. The new extensions included support for multidimensional attributes for points of interest, outliers, releasing fixed centers, and a location preference for cluster centers. The PACK method allows the joint consideration of all these extensions.

We first introduced the extensions with a controlled data set, where the original simulated clustering was known. The simulation considered a scenario with weighted data points, outliers and non-spatial attributes. In the experiment, the emphasis was on demonstrating the effect of penalty parameters λ_d and λ_o , which are related to the impact of non-spatial attributes and outliers, respectively. With a single simulated data set, we were able to illustrate the balance between spatial and non-spatial attributes in the data set. While the final decision on the values of clustering parameters depends on the application at hand, we argued that with this simulated data set, the optimal value for λ_d could be either 0.7 or 0.8. With 100 simulated data sets, we were able to generalize the impact of non-spatial attributes, decreasing the average standard deviation of the attributes linearly. On the contrary, the mean distances from points to the center naturally increased as we put more weight on the non-spatial attributes, as was expected. This trade-off between non-spatial attributes and distances is a key part in dual clustering.

To evaluate PACK with edge server placement application, we demonstrated the use of non-spatial attributes together with outliers and pre-determined server locations in a real-life data set with in total of 2732 base stations with mobile user connections. We used a non-spatial attribute, the average session duration, as a proxy for user application usage profiles. As a result, the average standard deviation of the user session durations on edge servers was reduced by 53% at best. On the edge servers such an improvement would allow specialization and a more efficient use of available computational resources, allowing edge servers to cater for location-specific application usage and user profiles. Regarding the infrastructure, such insights simplifies the selection of physical server hardware, possibly reducing deployment budgets and maintenance efforts.

Furthermore, the inclusion of outliers impacted the placement of servers in a positive way, decreasing both the mean distances between servers and base stations and the standard deviations of session lengths

in each server. Service level agreements allowing, accepting that certain base stations always offload user workload to cloud instead of an edge server improves the quality of service of the remaining network, as measured by latency. The outlier detection method presented here could consider spatial data with nonspatial attributes, however for a complicated spatial point event data there is a lack of such detection method. Moreover, we proposed an extension to the placement algorithm which allowed the release of fixed centers. Compared with the location preference extension, which favored the neighborhoods of the given points, the proposed releasing of fixed centers emphasized only the given points, either locating a center exactly on them or allowing a free placement for the center. Such functionality gives edge operators more flexibility in placement, allowing them, for example, to give a number of potential hot-spots an increased chance to gain an edge server, with the final placement depending on the overall quality of service.

Furthermore, we provided a comparison to the standard clustering methods of K-means and K-medoid. Even though these methods performed well in terms of spatial proximity and/or session similarity, they are unsuitable for edge server placement because the methods do not consider workload balance, resulting in an uneven distribution of workload between edge servers. Furthermore, the execution times of these classic algorithms are far less than those of the proposed methods.

Considering the size of the data sets, the experiments presented here were performed in reasonable time. Overall, if scalability is a concern, fractional membership can be used instead of hard membership (Lähderanta et al., 2021). Moreover, parallel computing can be used as the algorithm uses different starting values that are independent of each other.

In conclusion, we proposed the PACK method for capacitated spatial clustering, with number of extensions related cluster center locations, outliers and non-spatial attributes. We demonstrated these extensions with an example with simulated and controlled data and with an edge server placement problem, where two different scenarios were examined. We concluded that the inclusion of non-spatial attributes allows a greater control between spatial proximity and attribute similarity. Furthermore, in the edge server placement problem a better overall proximity and similarity was achieved when outlier points were detected and discarded from the cluster analysis. Control over the cluster centers provided a way to apply preference to certain locations without affecting the overall clustering quality. We argue that the inclusion of the novel extensions and the flexibility of the PACK method, combined with the easy-to-use open-source R software package *rpac*, provides a versatile toolbox for spatial clustering, especially in relation to edge server placement.

Funding

This research is supported by the Research Council of Finland 6Genesis Flagship (grants 318927, 346208); the ECSEL JU FRACTAL (grant 877056), receiving support from the EU Horizon 2020 programme and Spain, Italy, Austria, Germany, France, Finland, Switzerland; the Infotech Oulu research institute; the Future Makers program of the Jane and Aatos Erkko Foundation and the Technology Industries of Finland Centennial Foundation; by the University of Oulu and the Research Council of Finland PROFIS funding for mathematics and AI: data insight for high-dimensional dynamics (grant 326291); the Neural pub/sub research project, funded by Business Finland with diary number 8754/31/2022; and by the personal grant for Lauri Lovén on Edge-native AI research by the Tauno Tönnings foundation.

Code availability

rpac source code and the simulation experiments are freely available at <https://github.com/terolahderanta/rpac>.

CRediT authorship contribution statement

Tero Lähderanta: Conceptualization, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Lauri Lovén:** Conceptualization, Software, Data curation, Visualization, Writing – original draft, Writing – review & editing. **Leena Ruha:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing. **Teemu Leppänen:** Conceptualization, Writing – original draft, Writing – review & editing. **Ilkka Launonen:** Methodology, Writing – review & editing. **Jukka Riekk:** Supervision, Project administration, Funding acquisition. **Mikko J. Sillanpää:** Conceptualization, Supervision, Project administration, Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of data and material

The data set of the experiments is available at <http://sguangwang.com/TelecomDataset.html>.

References

- Ackerman, M., Ben-David, S., Brânzei, S., Loker, D., 2012. Weighted clustering. In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence. AAAI '12*, AAAI Press, pp. 858–863.
- Ansari, M.Y., Ahmad, A., Khan, S.S., Bhushan, G., Mainuddin, 2020. Spatiotemporal clustering: A review. *Artif. Intell. Rev.* 53 (4), 2381–2423. <http://dx.doi.org/10.1007/s10462-019-09736-1>.
- Arlot, S., Celisse, A., 2010. A survey of cross-validation procedures for model selection. *Stat. Surv.* 4 (none), 40–79. <http://dx.doi.org/10.1214/09-SS054>.
- Arthur, D., Vassilvitskii, S., 2007. K-means++: The advantages of careful seeding. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '07*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 1027–1035.
- Banerjee, A., Ghosh, J., 2006. Scalable clustering algorithms with balancing constraints. *Data Min. Knowl. Discov.* 13, 365–395.
- Basu, S., Davidson, I., Wagstaff, K., 2008. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. CRC Press.
- Berkelaar, M., et al., 2015. IpSolve: Interface to Lp_solve v. 5.5 to Solve Linear/Integer Programs. URL <https://CRAN.R-project.org/package=lpSolve>, R package version 5.6.13.
- Böhmer, M., Hecht, B., Schöning, J., Krüger, A., Bauer, G., 2011. Falling asleep with Angry Birds, Facebook and Kindle: A large scale study on mobile application usage. In: *Mobile HCI 2011 - 13th International Conference on Human-Computer Interaction with Mobile Devices and Services*. pp. 47–56. <http://dx.doi.org/10.1145/2037373.2037383>.
- Borgwardt, S., Brieden, A., Gritzmann, P., 2017. An LP-based k-means algorithm for balancing weighted point sets. *European J. Oper. Res.* 263 (2), 349–355. <http://dx.doi.org/10.1016/j.ejor.2017.04.054>.
- Brimicombe, A.J., 2007. A dual approach to cluster discovery in point event data sets. *Comput. Environ. Urban Syst.* 31 (1), 4–18. <http://dx.doi.org/10.1016/j.compenvurbsys.2005.07.004>, Extracting Information from Spatial Datasets.
- Celebi, M.E., 2014. *Partitioning Clustering Algorithms*. Springer.
- Charikar, M., Khuller, S., Mount, D.M., Narasimhan, G., 2001. Algorithms for facility location problems with outliers. In: *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics*, pp. 642–651.
- Chawla, S., Gionis, A., 2013. K-means-: A unified approach to clustering and outlier detection. In: *Proceedings of the 2013 SIAM International Conference on Data Mining. SIAM*, pp. 189–197.
- Cygan, M., Kociumaka, T., 2014. Constant factor approximation for capacitated k-center with outliers. In: Mayr, E.W., Portier, N. (Eds.), *31st International Symposium on Theoretical Aspects of Computer Science. STACS 2014*, In: *Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 25, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 251–262. <http://dx.doi.org/10.4230/LIPIcs.STACS.2014.251>.
- Dao, T.-B.-H., Duong, K.-C., Vrain, C., 2017. Constrained clustering by constraint programming. *Artificial Intelligence* 244, 70–94.
- Elliott, M.R., 2011. A simple method to generate equal-sized homogenous strata or clusters for population-based sampling. *Ann. Epidemiol.* 21 (4), 290–296.
- Farahani, R., Drezner, Z., Asgari, N., 2009. Single facility location and relocation problem with time dependent weights and discrete planning horizon. *Ann. Oper. Res.* 167 (1), 353–368.
- Ganganath, N., Cheng, C., Tse, C.K., 2014. Data clustering with cluster size constraints using a modified K-means algorithm. In: *2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. pp. 158–161. <http://dx.doi.org/10.1109/CyberC.2014.36>.
- Gatrell, A.C., Bailey, T.C., Diggle, P.J., Rowlingson, B.S., 1996. Spatial point pattern analysis and its application in geographical epidemiology. *Trans. Inst. Brit. Geogr.* 21 (1), 256–274. URL <http://www.jstor.org/stable/622936>.
- Grossi, V., Guns, T., Monreale, A., Nanni, M., Nijssen, S., 2016. Partition-based clustering using constraint optimization. In: *Data Mining and Constraint Programming: Foundations of a Cross-Disciplinary Approach*. Springer International Publishing, Cham, pp. 282–299. http://dx.doi.org/10.1007/978-3-319-50137-6_11.
- Grubestic, T.H., Wei, R., Murray, A.T., 2014. Spatial clustering overview and comparison: Accuracy, sensitivity, and computational expense. *Ann. Assoc. Am. Geogr.* 104 (6), 1134–1156. <http://dx.doi.org/10.1080/00045608.2014.958389>.
- Guo, Y., Wang, S., Zhou, A., Xu, J., Yuan, J., Hsu, C.-H., 2019. User allocation-aware edge cloud placement in mobile edge computing. *Softw. - Pract. Exp.* 50 (5), 489–502. <http://dx.doi.org/10.1002/spe.2685>.
- Gupta, S., 2017. A survey on balanced data clustering algorithms. *Int. J. Women Res. Eng. Sci. Manag.* 2 (9), 2611–2614.
- Gurobi Optimization, L., 2018. Gurobi optimizer reference manual. URL <http://www.gurobi.com>.
- Hale, T.S., Moberg, C.R., 2003. Location science research: A review. *Ann. Oper. Res.* 123 (1–4), 21–35.
- Hu, C.W., Li, H., Qutub, A.A., 2018. Shrinkage clustering: A fast and size-constrained clustering algorithm for biomedical applications. *BMC Bioinformatics* 19 (1), 19. <http://dx.doi.org/10.1186/s12859-018-2022-8>.
- Huotari, T., Rusanen, J., Keistinen, T., Lähderanta, T., Ruha, L., Sillanpää, M., Antikainen, H., 2020. Effect of centralization on geographic accessibility of maternity hospitals in Finland. *BMC Health Serv. Res.* 20, 337. <http://dx.doi.org/10.1186/s12913-020-05222-5>.
- Jin, X., Han, J., 2010. Partitional clustering. In: *Encyclopedia of Machine Learning*. Springer US, Boston, MA, p. 766. http://dx.doi.org/10.1007/978-0-387-30164-8_631.
- Kaufman, L., 1990. Partitioning around medoids (program PAM). In: *Finding Groups in Data*. John Wiley & Sons, Ltd, pp. 68–125. <http://dx.doi.org/10.1002/9780470316801.ch2>.
- Khachiyan, L., 1980. Polynomial algorithms in linear programming. *USSR Comput. Math. Math. Phys.* 20 (1), 53–72.
- Lähderanta, T., Leppänen, T., Ruha, L., Lovén, L., Harjula, E., Ylianttilä, M., Riekk, J., Sillanpää, M.J., 2021. Edge computing server placement with capacitated location allocation. *J. Parallel Distrib. Comput.* 153, 130–149. <http://dx.doi.org/10.1016/j.jpdc.2021.03.007>.
- Lähderanta, T., Lovén, L., Ruha, L., 2019. rpack. GitHub repository, GitHub, <https://github.com/terolahderanta/rpack>.
- Lee, I., Lee, K., 2015. The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Bus. Horizons* 58 (4), 431–440.
- Leyva-Pupo, I., Santoyo-González, A., Cervelló-Pastor, C., 2019. A framework for the joint placement of edge service infrastructure and user plane functions for 5G. *Sensors* 19 (18), 3975.
- Li, Z., Nie, F., Chang, X., Ma, Z., Yang, Y., 2018. Balanced clustering via exclusive lasso: A pragmatic approach. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. pp. 3596–3603.
- Lin, C., Liu, K., Chen, M., 2005. Dual clustering: Integrating data clustering over optimization and constraint domains. *IEEE Trans. Knowl. Data Eng.* 17 (5), 628–637. <http://dx.doi.org/10.1109/TKDE.2005.75>.
- Liu, H., Han, J., Nie, F., Li, X., 2017. Balanced clustering with least square regression. In: *Thirty-First AAAI Conference on Artificial Intelligence*. pp. 2231–2237.
- Liu, H., Li, J., Wu, Y., Fu, Y., 2019. Clustering with outlier removal. *IEEE Trans. Knowl. Data Eng.* 1.
- Lovén, L., Lähderanta, T., Ruha, L., Leppänen, T., Peltonen, E., Riekk, J., Sillanpää, M.J., 2020. Scaling up an edge server deployment. In: *2020 IEEE International Conference on Pervasive Computing and Communications Workshops. PerCom Workshops*, IEEE, Austin, TX, US, pp. 1–7.
- Lovén, L., Lähderanta, T., Ruha, L., Peltonen, E., Launonen, I., Sillanpää, M.J., Riekk, J., Pirttikangas, S., 2021. EDISON: An edge-native method and architecture for distributed interpolation. *Sensors* 21 (7), <http://dx.doi.org/10.3390/s21072279>.
- Lovén, L., Peltonen, E., Pandya, A., Leppänen, T., Gilman, E., Pirttikangas, S., Riekk, J., 2019. Towards EDISON: An edge-native approach to distributed interpolation of environmental data. In: *28th International Conference on Computer Communications and Networks (ICCCN2019)*, 1st Edge of Things Workshop 2019. EoT2019, IEEE, Valencia, Spain, pp. 1–6.
- Malinen, M.I., Fränti, P., 2014. Balanced K-means for clustering. In: *Fränti, P., Brown, G., Loog, M., Escolano, F., Pelillo, M. (Eds.), Structural, Syntactic, and Statistical Pattern Recognition. Springer Berlin Heidelberg, Berlin, Heidelberg*, pp. 32–41.
- Mitchell, J.E., 2002. Branch-and-cut algorithms for combinatorial optimization problems. In: *Handbook of applied optimization*, Vol. 1. Oxford, UK, pp. 65–77.

- Mulvey, J.M., Beck, M.P., 1984. Solving capacitated clustering problems. *European J. Oper. Res.* 18 (3), 339–348. [http://dx.doi.org/10.1016/0377-2217\(84\)90155-3](http://dx.doi.org/10.1016/0377-2217(84)90155-3).
- Negreiros, M., Palhano, A., 2006. The capacitated centred clustering problem. *Comput. Oper. Res.* 33 (6), 1639–1663.
- Ng, R., Han, J., 2002. CLARANS: A method for clustering objects for spatial data mining. *IEEE Trans. Knowl. Data Eng.* 14 (5), 1003–1016. <http://dx.doi.org/10.1109/TKDE.2002.1033770>.
- Nghiem, N.-V.-D., Vrain, C., Dao, T.-B.-H., Davidson, I., 2020. Constrained clustering via post-processing. In: *International Conference on Discovery Science*. Springer, pp. 53–67.
- Olukanmi, P.O., Twala, B., 2017. K-means-sharp: Modified centroid update for outlier-robust k-means clustering. In: *2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics. PRASA-RobMech, IEEE*, pp. 14–19.
- Papadimitriou, C.H., 1981. On the complexity of integer programming. *J. ACM* 28 (4), 765–768.
- Patel, K.A., Thakral, P., 2016. The best clustering algorithms in data mining. In: *2016 International Conference on Communication and Signal Processing. ICCSP, IEEE*, pp. 2042–2046.
- Peng, C., Zhang, Q., Kang, Z., Chen, C., Cheng, Q., 2021a. Kernel two-dimensional ridge regression for subspace clustering. *Pattern Recognit.* 113, 107749. <http://dx.doi.org/10.1016/j.patcog.2020.107749>.
- Peng, C., Zhang, Z., Kang, Z., Chen, C., Cheng, Q., 2021b. Nonnegative matrix factorization with local similarity learning. *Inform. Sci.* 562, 325–346. <http://dx.doi.org/10.1016/j.ins.2021.01.087>.
- Rahman, S., Smith, D.K., 2000. Use of location-allocation models in health service development planning in developing nations. *European J. Oper. Res.* 123 (3), 437–452. [http://dx.doi.org/10.1016/S0377-2217\(99\)00289-1](http://dx.doi.org/10.1016/S0377-2217(99)00289-1).
- Rajarajeswari, A., Ravindran, R.M., 2015. A comparative study of k-means k-medoid and enhanced k-medoid algorithms. *Int. J. Adv. Found. Res. Comput. (IJAFRC)* 2 (8), 7–10.
- Satyanarayanan, M., 2017. The emergence of edge computing. *Computer* 50 (1), 30–39.
- Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L., 2016. Edge computing: Vision and challenges. *IEEE Internet Things J.* 3 (5), 637–646.
- Sisodia, D.S., 2017. A subtractive medoids selection based fuzzy relational clustering of augmented web user sessions. *J. Intell. Fuzzy Systems* 33 (4), 2259–2268.
- Sisodia, D.S., Verma, S., Vyas, O.P., 2016. A discounted fuzzy relational clustering of web users' using intuitive augmented sessions dissimilarity metric. *IEEE Access* 4, 6883–6893. <http://dx.doi.org/10.1109/ACCESS.2016.2611682>.
- Steinbach, M., Kumar, V., Tan, P., 2019. Cluster analysis: Basic concepts and algorithms. In: *Introduction to Data Mining, 2nd Edn.*. Pearson Addison Wesley.
- Tan, P.-N., Steinbach, M., Kumar, V., 2016. *Introduction to Data Mining*. Pearson Education India.
- Tseng, P., 2001. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.* 109, 475–494.
- Tseng, G.C., 2007. Penalized and weighted K-means for clustering with scattered objects and prior information in high-throughput biological data. *Bioinformatics* 23 (17), 2247–2255. <http://dx.doi.org/10.1093/bioinformatics/btm320>.
- Vesanto, J., 2001. Importance of individual variables in the k-means algorithm. In: Cheung, D., Williams, G.J., Li, Q. (Eds.), *Advances in Knowledge Discovery and Data Mining*. In: *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp. 513–518. http://dx.doi.org/10.1007/3-540-45357-1_54.
- Wang, S., Guo, Y., Zhang, N., Yang, P., Zhou, A., Shen, X.S., 2019. Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach. *IEEE Trans. Mob. Comput.* 1.
- Wang, C.-M., Hsu, C.-C., Liu, P., Chen, H.-M., Wu, J.-J., 2007. Optimizing server placement in hierarchical grid environments. *J. Supercomput.* 42 (3), 267–282.
- Whang, J.J., Dhillon, I.S., Gleich, D.F., 2015. Non-exhaustive, overlapping k-means. In: *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, pp. 936–944.
- Wright, S.J., 2015. Coordinate descent algorithms. *Math. Program.* 151 (1), 3–34.
- Xiao, Y., Yu, J., 2012. Partitive clustering (K-means family). *Wiley Interdiscip. Rev.: Data Min. Knowl. Discov.* 2 (3), 209–225.
- Xu, L., Collier, R., O'Hare, G.M.P., 2017. A survey of clustering techniques in WSNs and consideration of the challenges of applying such to 5G IoT scenarios. *IEEE Internet Things J.* 4 (5), 1229–1249.
- Xu, J., Wang, S., Bhargava, B.K., Yang, F., 2019. A blockchain-enabled trustless crowd-intelligence ecosystem on mobile edge computing. *IEEE Trans. Ind. Inform.* 15 (6), 3538–3547.
- Yi, S., Li, C., Li, Q., 2015. A survey of fog computing: Concepts, applications and issues. In: *Proceedings of the 2015 Workshop on Mobile Big Data*. pp. 37–42.
- Yuan, C., Yang, H., 2019. Research on K-value selection method of K-means clustering algorithm. *J. Multidiscip. Sci. J.* 2 (2), 226–235.
- Zhang, B., Yin, W.J., Xie, M., Dong, J., 2007. Geo-spatial clustering with non-spatial attributes and geographic non-overlapping constraint: A penalized spatial distance measure. In: Zhou, Z.-H., Li, H., Yang, Q. (Eds.), *Advances in Knowledge Discovery and Data Mining*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1072–1079.
- Zhao, W.-L., Deng, C.-H., Ngo, C.-W., 2018. K-means: A revisit. *Neurocomputing* 291, 195–206. <http://dx.doi.org/10.1016/j.neucom.2018.02.072>.
- Zhu, S., Wang, D., Li, T., 2010. Data clustering with size constraints. *Knowl.-Based Syst.* 23 (8), 883–889. <http://dx.doi.org/10.1016/j.knosys.2010.06.003>.
- Zhu, J., Zheng, J., Di, S., Wang, S., Yang, J., 2020. A dual spatial clustering method in the presence of heterogeneity and noise. *Trans. GIS* 24 (6), 1799–1826. <http://dx.doi.org/10.1111/tgis.12687>, [arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.12687](https://onlinelibrary.wiley.com/doi/pdf/10.1111/tgis.12687).