Aggregating and visualizing urban heat demand using graph theory. A case study from Hamburg, Germany.

Dochev I., Seller H., Peters I.
Technical Urban Infrastructure Systems Group Prof. Irene Peters

18.1.2019
preprint version
Please quote this working paper when using the methods or results herein

Abstract

Many cities in Germany currently prepare 'heat demand cadastres' – thematic maps, depicting building heat demand – as a basis for decision support and policy design for reducing the climate impact of space heating. The prevailing trend towards open data goes into the direction of making these cadastres public, so that a variety of actors can use them and contribute to overall efforts for climate friendly heating. However, making such data public may violate the legal requirement of protecting private data. We present a way of tackling this problem with an aggregation approach for representing building heat demand. Using an algorithm based on graph theory, we form building groups that contain multiple buildings and do not allow the tracing of information on energetic characteristics and behavior to individuals. We also present how additional constraints can be introduced, for example, aggregating with respect to plot boundaries. Then we discuss how the building groups can be visualized in a map by presenting a way of generating custom-made geometry that represents each group. In the end we show a possible way of visualizing the groups with both specific heat demand (in kWh/m^2*a) and total heat demand (in kWh/a) in a single visualization. This allows the analysis of more complex questions involving energy efficiency and heat supply.

Keywords: urban heat demand, private data protection, aggregation, graph theory, visualization

1. Introduction

In the effort to increase energy efficiency and reduce climate relevant emissions, many cities and regions nowadays develop so-called "Urban Building Energy Models" (UBEMs) (Reinhart and Cerezo 2016). These models are used for analysing building stocks and informing policy makers about potentials for reducing energy use and CO_2 emissions and introducing renewable energy sources. With the use of GIS, many of these models become spatial models which can be visualized through thematic heat demand maps, or "heat demand cadastres". Space is of the essence in heat planning because heat transport is associated with losses and costs, while renewable energy sources are in many cases local. In addition, the need for more cooperation and coordination between public and private actors in urban planning as defined for example in the Copenhagen Charter (Danish Ministry of the Environment 2002) has caused spatial data to be made increasingly public. There are currently many examples of municipal and regional authorities that operate geoportals allowing open access to numerous spatial datasets - natural environment, built environment, technical and transport infrastructure and many more. This trend has also reached the energy sector with the introduction of publicly accessible energy-relevant datasets both on the supply and the demand side – e.g., solar or geothermal energy potentials but also building energy demand maps. However, energy consumption and demand, respectively, reflect personal behaviour as well as the condition of property such as buildings (which to a large extent are privately owned). Therefore, a potential conflict arises between the need for open data and the need for personal data protection. Different countries and authorities go about this issue in different ways. We concentrate on the case of Germany, where data protection requirements in this context are relatively strict and present a serious obstacle to open-access energy demand data.

2. Specific Context

The GEWISS Project Hamburg (GEWISS Project 2018) is the context in which this paper originates. The city-state of Hamburg with a population of 1.8 million inhabitants and a building stock of 300 000 buildings has released, in 2017, a publicly available thematic map depicting building heat demand – the so-called heat demand cadastre (*Wärmekataster*). This dataset, however, should not allow the user to identify characteristics of individual persons. To this end, building data can only be released if aggregated to a certain level that ensures data protection. This is operationalized by defining a minimum amount of dwelling units per aggregated building group for residential buildings. For non-residential buildings, data protection requirements are more difficult to put into numbers, since the object of protection is less clear – the individuals working or the company owning a building? Similar ambiguity can be observed for rented apartments - is the object of protection the behavior of the resident or the quality of the building, given that the latter is owned by a different individual or company?

For the purpose of presenting a building heat demand aggregation approach we use a requirement given to us by the Hamburg Ministry of Environment and Energy (*Behörde für Umwelt und Energie*). It states that building groups have to include a minimum of five units, where a unit in a residential building is the dwelling unit, while each non-residential building is comprised of a single unit. We use this as a working hypothesis to develop the aggregation approach, while at the same time providing the flexibility to adjust for alternative formulations of requirements and/or minimum counts.

3. Defining the problem

The problem at hand can be approached with the development of an aggregation algorithm that groups buildings in space based on a minimum amount of units in each individual group. This is the minimum requirement. However, we find it useful to add some further requirements.

The purpose of heat demand mapping is to provide a basis for planning. For strategic energy planning at the city level, the size of the aggregation units may be less important. Not so at the local level. There, the level of aggregation is quite relevant. Take for example energy planning for neighbourhoods, where public authorities together with the private sector need to analyse demands and potentials in order to devise concrete measures for individual objects in the neighbourhood. Therefore, the aggregation approach will lead to most useful results if the unit count of the created groups is as close to the minimum required as possible. This will allow a finely-grained structure of building groups, which affords more flexibility when analysing the local context.

Moreover, data at a coarser spatial level easily run into the so-called “ecological fallacy” problem. Also known as “modifiable areal unit problem” (Openshaw 1984), this concerns the masking and distortion of spatial phenomena at an aggregated level due to averaging-out effects. This is an additional reason why we choose an aggregation approach that minimizes the group unit count.

In addition, heat demand maps are a way of analysing potentials for district heating, which is considered a key technology for a sustainable heat supply in some urban contexts. Since district heating infrastructure tends to follow the street network of a city, we aim at aggregating in such a way that respects the street layout of a city and therefore increases usefulness for district heating planning.

A further requirement that we introduce is that of spatial non-overlapping of groups. The reasons for this are map usefulness and ease of orientation. Having a building of one group in-between buildings of another group would impair the usability of the map.

Finally, building aggregation in thematic maps can be split into two distinct tasks – defining building groups and defining the geometry to represent these groups, the latter being not a trivial issue at all that has to be addressed appropriately.

What is required can then be defined as an aggregation algorithm that (i) groups buildings to satisfy a minimum unit count, (ii) optimizes to reduce unit counts to as close as possible to the minimum, (iii) produces groups that correspond to the street layout, (iv) produces spatially non-overlapping groups and (v) generates a geometry representing each individual group.

4. General approaches

Before analysing more advanced or custom-made solutions to the problem, we look at the simpler aggregation approaches, commonly found in thematic maps.

4.1. Using existing spatial units

The easiest way of approaching the aggregation task is to use existing spatial units, which also have a defined geometry. These include units like census tracts, postal code areas[1], “neighbourhoods” (our translation for the unit called *Stadtteile,* an established administrative unit that is usually comprised of several census tracts), or urban blocks (*Baublöcke*, another established administrative unit that denotes areas in-between the streets of a city). Although “neighbourhood” (*Stadtteil*) as a spatial unit can come in various shapes and sizes, it usually includes many buildings and the buildings in one neighbourhood would most probably satisfy the first condition (a minimum of five units). Neighbourhoods are usually bound by the main street network which satisfies condition (iii). Using existing neighbourhood boundaries to create a polygon and thus represent heat demand would also satisfy the need for a geometry representation. The same generally applies to census tracts and postal code areas. However all of these units are predefined which lowers the flexibility of the aggregation

[1] Geographic data representing postal codes is generally available

and does not satisfy condition (ii) – optimizing for building groups with a unit count as close as possible to the minimum.

The urban block is a relatively well-known spatial unit which comes closest to what is needed for the purpose at hand. In the specific case of Hamburg, the city is divided into ~100 neighbourhoods, ~1 000 census tracts and ~8 000 urban blocks. With ~300 000 buildings, the average number of buildings in each neighbourhood is 3 000, in each census tract 300 and in urban block 37.5. The amount of units (as defined for data protection) will be correspondingly higher, depending upon the type of buildings in each neighbourhood. The urban block is the smallest pre-defined spatial unit and it comes close to satisfying all requirements for aggregation. The minimum of five units per building group will in most cases be met and would require only a small amount of corrections, since the average is 37.5 buildings per urban block. The size of urban blocks, although varying, does not vary so much as to produce too many groups of less than five units.

Additionally, urban blocks per definition follow the street layout and have a known geometry already made available by the public authorities of Hamburg[2]. A further argument for their use is the fact that in some cases (including the specific case of Hamburg) some statistical data is available at this level, making the urban blocks also statistical units. Having energetic data at an aggregation level which is the same as, or corresponds to, a statistical unit allows for multi-sectoral analysis which is advantageous (for example analysing connections between heat demand and socio-demographic data).

Still, the urban block, although a small spatial unit, only partially satisfies requirements (i) and (ii). Some urban blocks might have less than the required amount of units and would need to be merged with neighbouring ones or removed from the dataset. With a building count per urban block of 37.5 on average the urban block is still too large for condition (ii). However, although we do not use the urban block as it is, we use it as part of our aggregation approach.

4.2. Raster cells

An alternative to existing spatial units is the use of a raster grid. By superimposing a raster grid of a certain size over the building stock of a city, each building is allocated to a specific raster cell, thus forming a group with the others in the same cell. Each group can then be represented by the geometry of the raster cell – a square. The first problem with this approach is the dependency on the raster grid position. Shifting the grid around in the Cartesian plane would change the content of each each cell. In other words, the allocation of a building to a cell is arbitrary and depends upon the initial position of the raster. A further problem is that it does not satisfy the requirement for having as few units as possible. The cell size has to be such that every cell has the minimum amount of units, which leads to large cells in less densely built-up areas. This could be tackled by starting at a certain cell size satisfying the minimum requirement and iteratively subdividing each cell until the smallest possible unit that satisfies data protection is found. In this way almost all conditions are met, except the street layout condition (iii). Superimposing a raster grid over the city means fitting the urban fabric and layout to a

[2] Even without an available geometry, the task of extracting the areas in-between the street network of a city is a relatively straightforward GIS task.

regular grid, which might be acceptable for locations with street layouts following an orthogonal grid[3]. For other cities however, this would mean an aggregation unit that does not follow the street layout and hence violates the respective requirement.

5. Methodology

The methodology for defining the building groups and generating the geometry representation of the groups are discussed separately in the following sections.

5.1. Number of units per building

Adhering to the “five plus” rule defined in Section 2 is not a straightforward task, since the digital cadastre of Hamburg (*ALKIS*) does not contain number of dwelling units per building. Therefore we need to estimate this value. Usually, one would approach the problem by trying to estimate as closely as possible, having some underestimations and some overestimations, but approximating with an average. In our case however, since the strict requirement is for a minimum unit count, we aim at generally underestimating in most cases. With this purposeful underestimation we try to err on the side of caution, avoiding the cases where our estimation is too high and a building group is presented as having more than five units, when in reality it has less.

Two variables found in the *ALKIS* can be used for the estimation – the floor count and the gross floor area of a building (resulting from the multiplication of the number of floors and the area of the building footprint). We use the floor count assuming that there is at least one dwelling per floor. Additionally, through observation we found that some residential buildings in the cadastre have non-residential uses in the ground floor, although the building use is given as only residential. To avoid overestimating the dwelling units as a consequence, we consider every residential building with up to three floors as having a dwelling count of one. Buildings with between three and five floors we consider as having three dwelling units and above five floors the dwelling units are equal to the floor count. Non-residential buildings have one unit per building. These rules are summarized in Table 1. It is obvious that this leads quite an underestimation for most of the buildings, but it is a needed precaution in order to minimize the risk for breaching data protection requirements.

Building use	Floors	Estimated units
Residential and mixed-use with residential	1 to 3	1
	3 to 5	3
	>5	number of floors
Non-residential	any	1
Any building use without heat demand	any	not considered for aggregation

Table 1. Overview of building unit classification.

[3] Known in urban planning as a “Hippodamus” or “Manhattan” layout – a modern day example is Manhattan in New York.

5.2. Defining the building groups

In this section, we explain the method used to assign a group ID to each building in the *ALKIS*, thus defining which buildings are parts of the same group.

Since the urban block satisfies many of the defined requirements we approach the aggregation problem by starting at this spatial unit. We then subdivide the urban block into building groups with number of units as close as possible to the minimum requirement. With the initial grouping according to the urban blocks all buildings receive an ID corresponding to the ID of the urban block they reside in (urban block IDs are six-digit numbers for example “100000”). The further subdivision within the block is done by appending an additional integer to the urban block ID – “100000_1”, “100000_2” etc. The problem lies in generating the additional IDs. In order to produce spatially clustered groups and avoid overlaps, we need to describe the spatial relationships between buildings. This means numerically describing the positions of each building and the corresponding distances to the other buildings.

In order to come up with a robust method that performs well in different urban settings – single-family or row-house house neighbourhoods, denser city centres, large apartment complexes, commercial areas etc. – two approaches are possible. The first approach would be to classify urban space into such categories (single-family, city-centre etc.) and adapt the spatial grouping rules based on the category. The problem would be that there would always be exceptions and urban spaces that fall into no category. For this reason we use another approach - we analyse the problem in the general case of a random spatial constellation. By having an algorithm that solves the general case of a random input we try to make the desired output independent of peculiarities of the input.

5.2.1. Graph Theory – Minimal Spanning Tree

The problem then translates to analysing the spatial relationships (distances) between a random number of points (an urban block can potentially have any number of buildings) randomly distributed in space. In order to illustrate the approach we first represent each building with its centroid point and then show how this can be expanded to a constellation of arbitrary geometries (building footprints). Also, for illustration purposes in this section we consider each building to have a single unit.

The problem definition already points towards graph theory. A random spatial constellation *S,* consisting of 20 (*n*) points is presented in Figure 2 (left). These points represent a random urban block with a random positioning of 20 buildings.

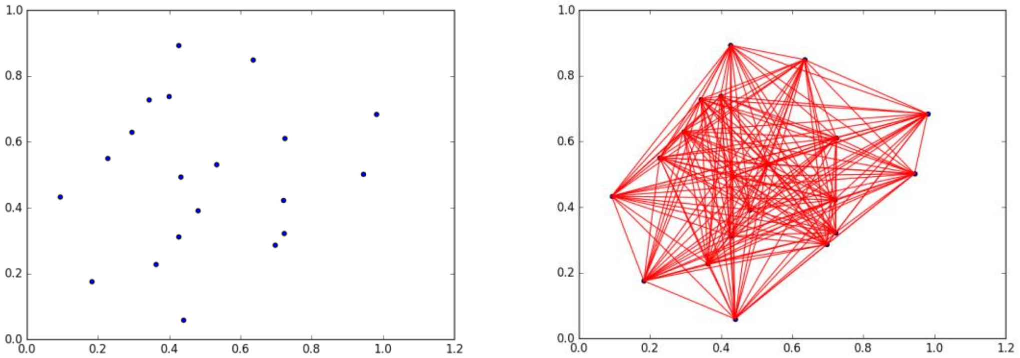

Figure 2. A random set of points S and a complete graph N.

Describing the distances between the points in *S* can be represented by a complete graph G, consisting of *n* nodes and *n***n*-1 (380) edges, each edge representing the Cartesian distance between two points (Figure 2 right). For the purpose of grouping buildings spatially, however, describing distances between every set of two points is not enough. What is needed is a description of which buildings are closest to which other building/s. A graphical representation of this is a minimal subtree that spans every point of *S* – a minimum spanning tree (MST) (Figure 3).

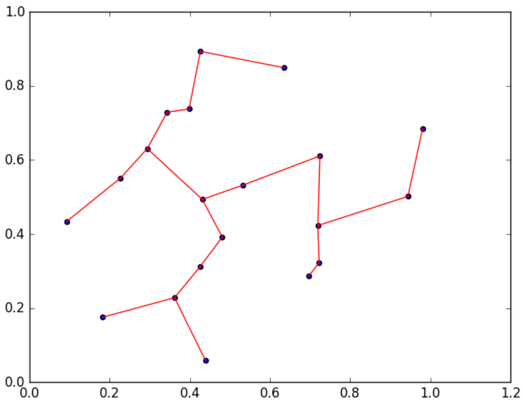

Figure 3. A minimum spanning tree of S.

A MST[4] is such a subtree of G, that spans *S* and has a total edge length[5] that is minimal compared to all other subtrees of G that span *S*. The MST is a unique subgraph of G if the edge lengths are unique[6] (Kruskal 1956). Describing the distances between points (buildings) in this way is advantageous since it allows the clustering of points based on proximity. Since the MST is a minimum tree, its longest edge connects the two subtrees of the MST that are furthest apart. Therefore one can obtain two sets S_1

[4] In the spatial context this can be referred to as a “shortest spanning tree”, since edge weights represent lengths.

[5] A general term here is ‘weight’. Since we represent distances in space we use ‘length’.

[6] Modern digital cadastres have usually high precision in coordinates, up to centimetres in some cases. Therefore the case of two buildings in a digital cadastre being exactly the same distance away from another building is highly unlikely. In this rare case there would be two MSTs. The one that is used is arbitrary and depends on the order of the input buildings. This is not an important issue, since if two buildings are indeed exactly the same distance away from another one, both possible MSTs would be appropriate for the aggregation.

and S_2 that are as far away from each other as possible by disconnecting the MST at the longest edge. S_1 and S_2 are then the node sets of two connected components of the MST. Repeating the process by disconnecting the next largest edge of the MST will then produce three components which are furthest away from each other as possible. Continuing the process will eventually lead to the disconnection of the whole tree and the production of *n* connected components with a single node in each. Since the purpose is to create groups of a certain amount of points in each (in the concrete case five) the splitting process has to stop and insure that no edge that is removed leads to a connected component being smaller than five points. For this we use an algorithm that:

- creates a complete graph out of a set of point coordinates
- creates a minimum spanning tree using Kruskal's algorithm (Kruskal 1956)
- sorts all edges according to their length
- takes the first (longest) edge and splits the MST by disconnecting the two nodes of the edge
- measures the number of points in each connected component
- if none of the connected components have a node count lower than the required (five) proceeds to the next longest edge
- if a connected component has a node count lower than five, the edge is restored and the algorithm moves to the next edge.
- the algorithm ends after it has iterated over all the edges of the MST.

Figure 3. illustrates this process with the random set of points presented above. The first three longest edges are numbered i, ii, iii. The first edge is removed however it leads to a component consisting of only two points (marked with a black ellipse). The edge is therefore restored. The next two edges are then removed. The node counts thereafter are six, six and seven and no further splitting is possible.

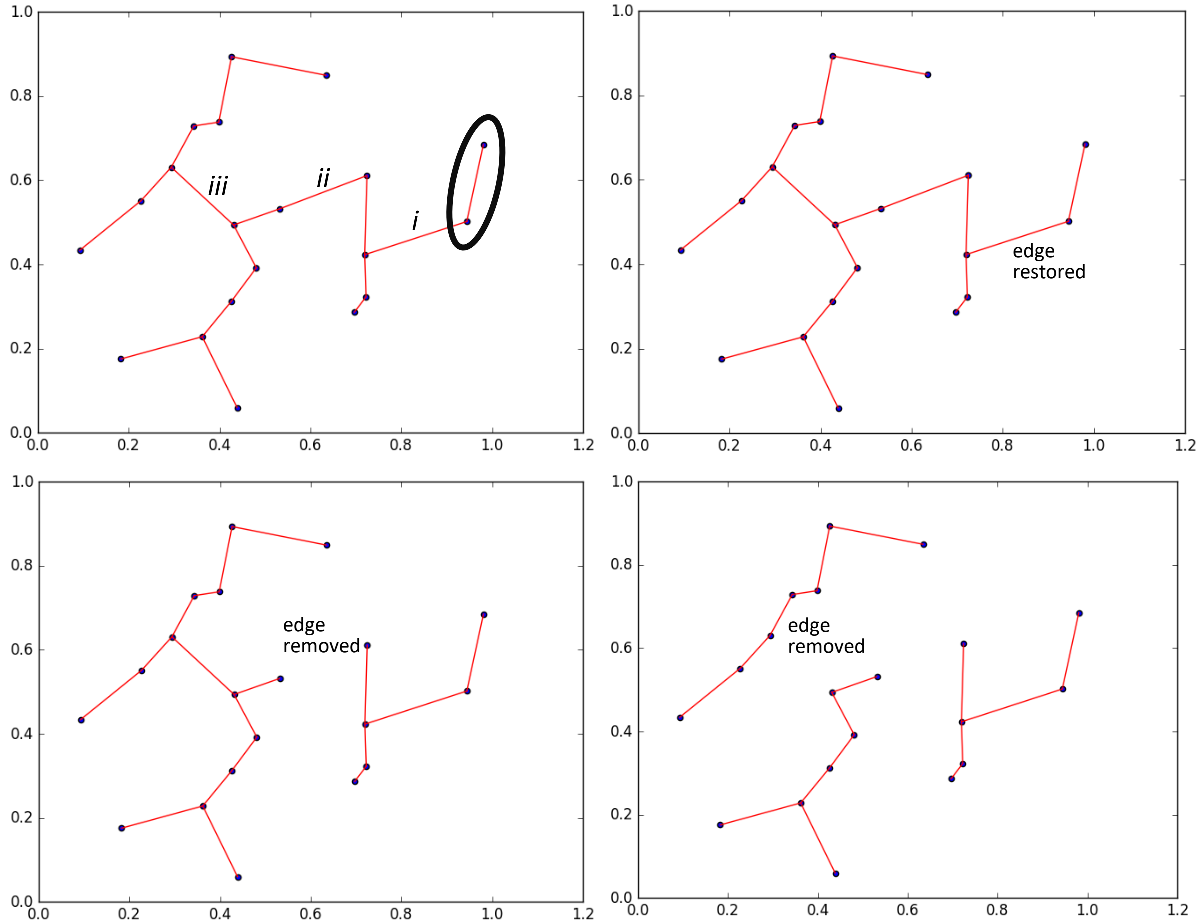

Figure 3. Splitting the MST while restoring edges that lead to too small connected components.

The three connected components in the end represent the three groups into which the points are divided. Each component is labeled with an arbitrary ID – 1, 2 or 3. If we assume that this random urban block had an ID of '100000' (as in section 5.2.), each point (representing a building) receives an ID of '100000_1', '100000_2', '100000_3' respectively, corresponing to the labels of the connected components. Thus each building is uniquely labeled with the urban block and group ID it resides in.

The labelling order is not of importance. It does not make any difference, for example, if the labeling proceeds from left to right or from up to bottom or in an arbitrary fashion. It is only important that three distinct IDs are created and each point is labeled with the ID of its connected component.

It is clear that the 20 initial points can, theoretically, be divided in four groups of five. At this point however we make a compromise between the size of the groups and their spatial layout. Looking for an optimal split in exactly four groups may lead to some groups having large distances between one or more than one of the group members. Apart from the visual unintuitiveness, this may also be impractical when it comes to the planning of district heating grids, where shortest grid lengths are generally sought for. The defined algorithm will produce groups of between 5 and 9 points per group in the general case which we consider a reasonable compromise between size and spatial constellation in view of the purpose of the aggregation.

5.2.2. Plot reweighting

Although for the purpose of strategic or regional energy planning a single building might not be of great importance, building complexes of similar use are – for example large prefabricated apartment blocks, hospital complexes or school campuses. Attempting to cluster only spatially does no take into accout such use- or ownership-dependant groupings of buildings. This leads to some building groups containing buildings that are part of other well-defined complexes, due to the spatial constellation within the urban block. Since ownership data is not available in this case we attempt to represent this with the plot structure of the urban block. We introduce a weighting of the lenghts when constructing the connected graph *G* spanning edges between all pairs of buildings. The distances between buildings within a single plot are multiplied (re-weighted) with a factor of 0.05. In this way edges between buildings in the same plot are always very short and are much further down the ordered list of edges to be removed. This results in the tendency that the buildings within the same plot are usually part of the same group, since edges between buildings of different plots are taken as "artificially" much longer after the reweighting.

This modification of the purely spatial logic allows for the algorithm to tend to preserve ownership-specific building constellations which is advantageous for planning. However, note that representing ownership, although only indirectly with the plot structure, might be a violation of data protection requirements depending upon the rules for data protection. For the purpose of our paper, only the number of units (dwelling units or otherwise) is considered. An example of the plot reweighting is presented in section 5.2.4.

5.2.3. Maximum distance within a group

Another adjustment to the basic splitting logic is the introduction of a maximum distance between the buildings in a group. There are situations in urban space, where a single building is in a highly isolated location even within an urban block - a small hut within a park complex, or a small workshop in agricultural land on the outskirts of a city. Although the distance to the closest neighbour building might be great, the basic algorithm will not divide the group at this edge, since this will produce a connected component (the small building) which is too small for data protection. However, for planning purposes, especially of heating grids, having such buildings within a group, lying at such a distance from its neighbours, represents a problem. It is unlikely that a long pipeline will be built to supply such buildings – unless this single far-away building is rather large and has a high total demand. Then it becomes more important for heat planning. In order to mirror these arguments into the algorithm and into the groups it produces we formulate the following rules:

If a building that does not fulfil the minimum data protection requirement is located on an edge longer than a certain maximum distance, the edge is removed and the building is marked with an ID 'Anonymized'. This building is not part of any building group and is filtered out. The maximum distance is a function of the areas of this building and the other building on the edge and a given distance factor. In this way, the maximum allowed distance is different depending upon the size of the buildings. The larger the buildings, the longer the distance allowed between them. The formula for the distance function is defined as:

$$M = m * 2(\sqrt{\frac{a_1}{\pi}} + \sqrt{\frac{a_2}{\pi}})$$

where:

M = the allowed distance between the two buildings
m = the distance factor, a setting for the algorithm
a_1 and a_2 = the floor areas of the two buildings

The logic behind the formula is to represent the two building geometries as circles with areas equaling the areas of the buildings. The allowed distance between them is then a function of the diameters of the two circles. The larger the diameters, the larger the building areas and therefore the larger allowed distance between them. An example of the effects of this is presented in the next section. The distance factor is an adjustable parameter to the algorithm.

5.2.4. A real-world example

The following section exemplifies the basic algorithm, the plot reweighting and the maximum distance, presented in sections 5.2.1., 5.2.2. and 5.2.3. respectively. An existing constellation of buildings and plots all in the same urban block is taken as an example (Figure. 4). As opposed to centroid points, in this example we take building polygons as the nodes of the complete graph. The generated MST takes into account the factor multiplication in the plot reweighting (0.05 * distance for buildings within the same plot). Notice the edge marked with (3). It is obviously longer than an edge located at (2) and should not be part of the MST. However since (3) connects buildings within the same plot its length is decreased with the factor 0.05 and is shorter than (2). Therefore the MST includes edge (3) not edge (2).

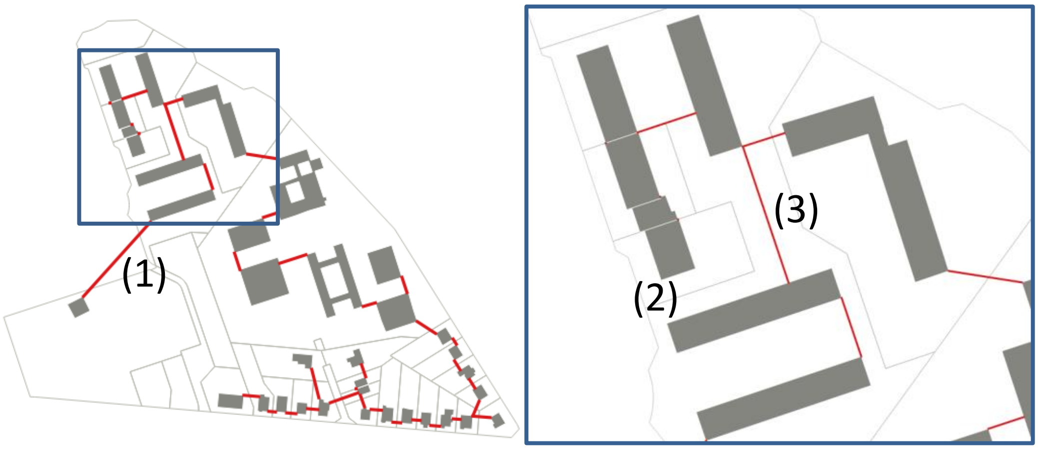

Figure 4. Example of the plot reweighting.

The algorithm then proceeds to split the MST into connected components representing building groups. The longest edge of the MST (edge (1) in Figure 4) is obviously the first candidate for disconnection. In the general case, the edge would be restored, since it will result in a single-building group that has less than five units. However, in this case the maximum distance rule is applied. Based on the length of the edge and the areas of the two connected buildings, the edge is removed and the isolated building is considered as “Anonymized”. Note that single-building groups are also allowed if a single building has more than five units, depending upon the rules defined in section 5.1. Therefore edge (1) could also be removed if the isolated building itself has more than five units and therefore constitutes a “proper” group. In that case however, the isolated building will receive a group ID and be defined as a building group.

Splitting the remaining edges leads to Figure 5.

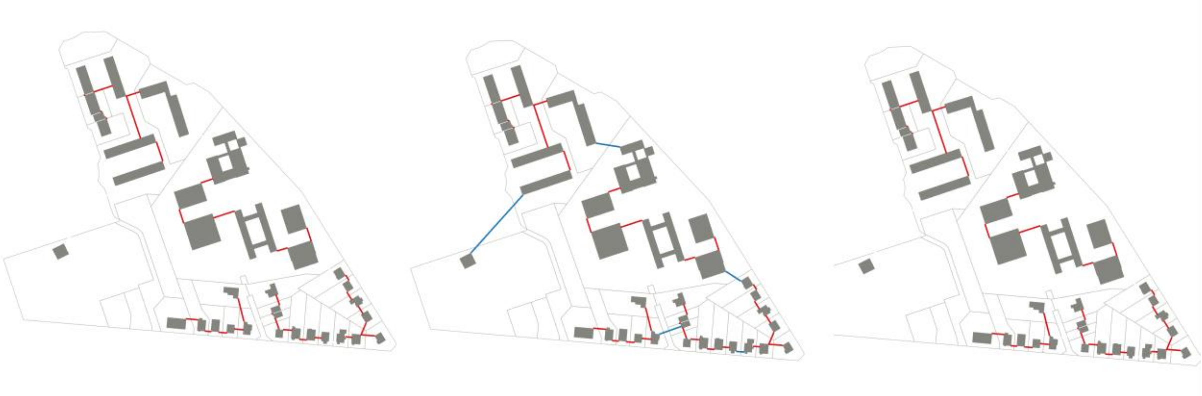

Figure 5. Splitting the urban block.

5.3. Creating Geometry Representation

5.3.1. The Representation Problem

After all buildings in the urban block receive a group ID (or are anonymized) the question remains how to present their heat demand. For specific heat demand (i.e. per m²) at the group level we sum all floor areas [m²] and total heat demands [kWh/a] and then divide them producing one value - the weighted average of the specific heat demand [kWh/m²*a] for the group. One way of spatially representing this value is to use the existing building geometries and symbolize each with the colour that reflects the specific heat demand of the building group (Figure 6.). The problem with this approach is rooted in the general principle that objects (buildings) should not be used to visually represent the characteristics of other objects (building groups).

Figure 6. Building groups represented with building geometries.

Figure 6 is an example for this. From the viewpoint of the map user it is difficult to understand that the colours refer to values for the groups and not for the individual buildings. This can be explicitly written in the legend, however it is not visible. Moreover, when groups have similar values one cannot distinguish which buildings are in which group (for example between 710005_4 and 710005_2). Labelling each group (Figure 3. left) does not help to overcome this. Labelling each building (Figure 3. right) does, but it overloads the map with content.

An alternative to the building geometry is to use the plot geometry. This is a viable way for plots like the ones in Figure 3. Plot geometries however come in various shapes and sizes and using them as basis fails in areas where there is a single building in a large plot.

It is for these reasons that we generate a custom-made group geometry that corresponds to each building group.

5.3.2. Convex Hulls, Concave Hulls and Aggregate Polygons

The task of finding a geometry that represents each building group can be formulated as finding a polygon in 2D space that includes all the vertex points of the building geometries and has a form that fits the general outline of the underlying building geometries. This can be interpreted as a "convex hull". A convex hull is a *convex* polygon with a minimal area that includes all points of a given set. It has inner angles of less than 180°. A *concave* polygon is a polygon that can have inner angles of more than 180°. Figure 7 presents the difference between the two (McConnell 2006, pp. 130–131).

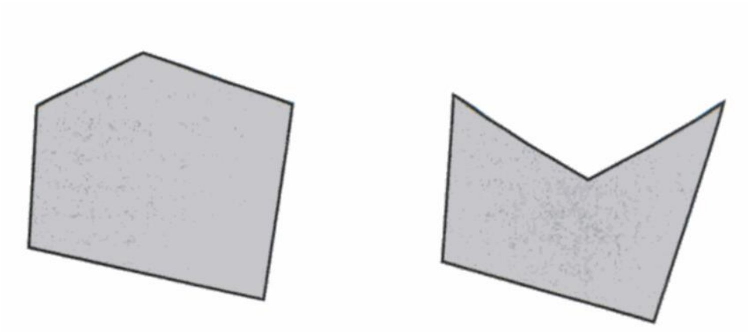

Figure 7. A convex (left) and a concave (right) polygon (McConnell 2006, p. 131).

A convex hull can be used for the task of representing the building groups. However in order to preserve their general shape, in many cases a concave hull is more appropriate, (see (Duckham et al. 2008), who also argue that for preserving the shape of a set points, convex hulls are insufficient). Generating a convex hull for a set of points is a relatively well known GIS operation with algorithms being a standard part of many GIS and spatial analysis libraries. Concave hulls are more difficult to compute and have been the object of research for some decades – with for example α-shape (Edelsbrunner et al. 1983) or χ-shape (Duckham et al. 2008) algorithms.

The α-shape or χ-shape algorithms are mainly employed in the field of image recognition which is a more complicated task than our task of representing building groups. For this reason we use a simpler approximation of a concave hull using a combination of two polygon buffers. This approach is referred to as "Aggregate Polygons" available as SQL code at Github (Düster 2011). Although we do not use the code itself, we adopt its approach.

In essence the method buffers each building geometry outwards at a given distance and dissolves the overlapping polygons to produce a single buffer. Then a second buffer is generated, but with negative distance, which means it buffers inwards from the previous buffer. In the process the areas in-between buildings become parts of the buffer area. The orthogonality of the geometry representation stems from the buffer options[7], see (Santilli 2009).

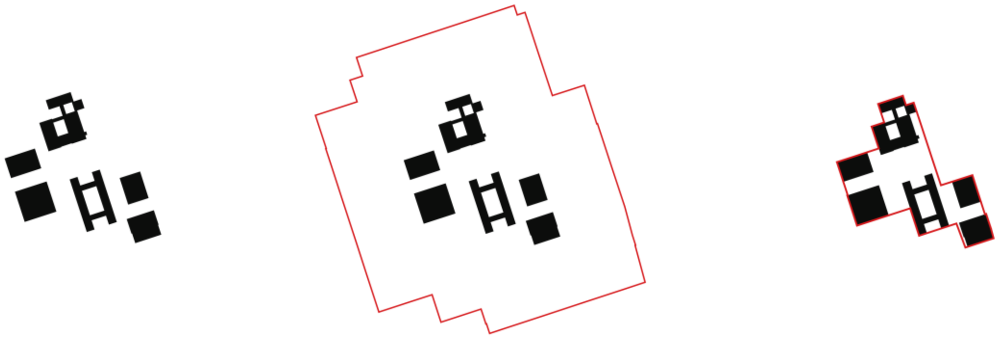

Figure 8. Geometry representation with two buffers. See (Düster 2011)

The success of the buffering depends on the buffer distance. Depending upon the spatial constellation and orientation of buildings, different distances would be most appropriate. For this, we use a trial and error approach. We start with half the length of the longest edge of the MST for the building group. Since all buildings are buffered with half the distance of the longest edge, even the two most distanced ones should have buffers that at least touch.

However, the application of the inward buffer results in some constellations whose geometry is not contiguous. To correct for this, we multiply the distance by 2. If the geometry is still not contiguous, we generate a convex hull, which is by definition contiguous.

5.4. Visualization

After the building groups are defined and a geometry representation is produced, what remains is to visualize the heat demand characteristics of the building groups in a way that is useful for planning. There are two general numeric heat demand characteristics that are used in the practice – specific heat demand in kWh/m²*a and total heat demand in kWh/a (or MWh/a). The former can be interpreted as a measure of energy efficiency, the latter of energy quantity needed per year. There are other characteristics that could be of interest – for example heat loads, heating system types, heating system temperatures or refurbishment state. We however concentrate on the specific and total heat demands, as the most widely depicted in heat demand maps. It has to be noted that heat demand can also take the form of useful energy (delivered from radiators or showers and taps), final energy (which includes transmission losses in a building and conversion losses of boiler, heat transfer station or similar), but also primary energy (contained in the raw fuels used). Which exactly is to be used depends on the specific case. For this paper we use final energy. Since all of these are measured usually in kWh it does not influence the properties of the visualization which of these are chosen.

In thematic maps, polygon characteristics are often symbolized with a coloured filling of the entire polygon. Since our polygon representation of the building groups includes spaces in-between

[7] We use a 'metre limit' of 2.5 meters, as in (Düster 2011). The options for the buffer generation are part of the buffer class of the GEOS library (Santilli 2009).

buildings, symbolizing the entire polygon makes small buildings placed at some distance from each other disappear within the large polygon representing the group. Filling the entire polygon with colour (Figure 9 left) can mislead the map user into thinking the group has a large total heat demand, even if the colours depict specific heat demand and this is presented in the legend. The purpose of the polygon representation of the building group is to represent the buildings and designate which buildings are in which groups. The size of the polygon representation however is not in any way a function of any heat demand related characteristics[8] and it (the size) should not influence the map user. In order to avoid giving too much visual significance to the size of the polygon representation, we use only the outline of the polygon (Figure 9 right).

Figure 9. Visualising aggregated final energy.

Note that the thickness (lineweight) of the outline in Figure 9 (right) is different for the groups. This is because we introduce another aspect that aims at increasing the usefulness of the map. In general, thematic heat demand maps depict either specific heat demand or total heat demand. We propose combining the two in one visualization. We colour the outline of the polygon representation based on specific heat demand and adjust its thickness as function of the total heat demand.

This has the advantage that it allows a more integrated analysis of heat demand in space. For example, it allows the quick visual localization of large 'heat sinks'[9] with low or high energy efficiency. This is advantageous for planning since it points towards appropriate strategies. For example a large heat sink with low specific heat demand is likely a target for renewable heat supply, while a large heat sink with high specific heat demand is likely first a target for an increase of efficiency.

Lastly, we discuss the colour scheme and size of the symbolized classes. Positive characteristics are usually associated with green colours, while negative ones with red. In the concrete case we propose

[8] Buildings with very large footprint areas will have a large polygon representation, but this applies also to small buildings with large distances in-between.
[9] Large heat consumers – large multistory residential buildings, but more often hospitals, schools or commercial complexes

using an already relatively known colour scheme in Germany – the green-yellow-red colour gradient of the energy certificates according to the Energy Efficiency Ordinance in Germany (*EnEV*) (Figure 10.).

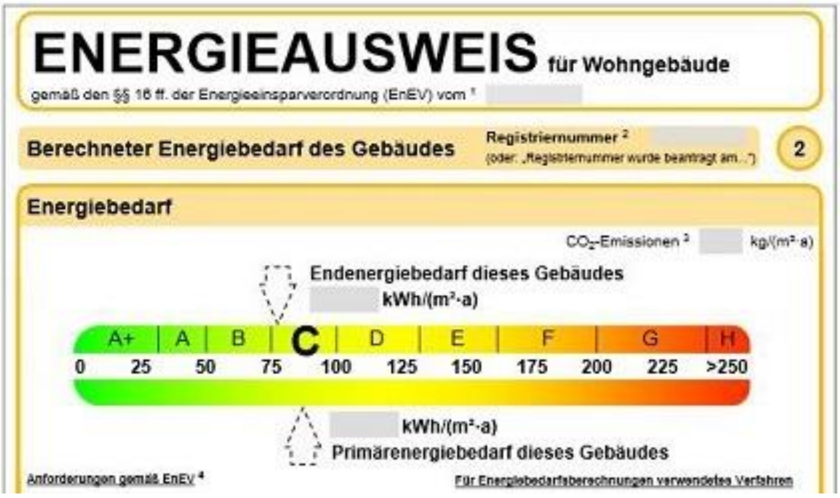

Figure 10. Specific heat demand colour scheme of energy certificates in Germany. Source: (Federal Ministry for Economic Affairs and Energy 2013)

We start with this colour scheme and adjust the yellow (middle) tone so that it is closer to white. This allows more contrast between the yellow and orange tones. In order to retain easier recognition we decrease the number of classes and classify into five classes with 50 kWh/m^2*a bins (Figure 12). The exact colour scheme is also dependant on the background map. In this paper we use an orthophoto as background map (Figure 12). The darkness of the orthophoto increases the contrast with the adjusted colour scheme. Having a lighter background means that the middle tones (yellow-white) would have to be again adjusted. An important point is adjusting for colour blindness, but this would require changing the basic green-yellow-red palette. This adjustment is beyond the scope of this paper.

5.5. Software used

The presented algorithm was written in the Python language using the Numpy (Oliphant 2006), Scipy (Jones et al. 2001) and shapely (Gillies et al. 2018) libraries and the PyQGIS library of the open-source GIS software QGIS (QGIS Development Team 2018). The visualizations were made again with QGIS. The code as well as a test dataset is available on Github (https://github.com/ivandochev/aggregating-urban-heat-demand). The script is designed to run within QGIS 2.18 as an algorithm from the processing toolbox.

6. Results

We applied the described methods to a dataset of 300 000 buildings (residential and non-residential) in the city of Hamburg. The algorithm produced 40 000 building groups. The group size for the majority of the groups is between five and ten units (Figure 11), which means the groups are relatively small and close to the minimum required (five units). This was one of the conditions defined in Section 3.

We use the geometry representation and the visualization approach to produce Figure 12. We use a digital orthophoto (LGV Hamburg 2016) as background map.

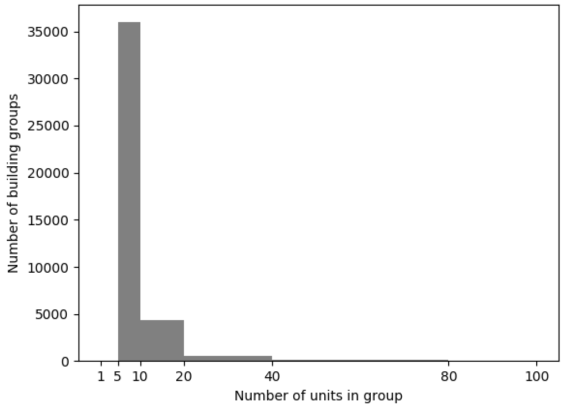

Figure 11. Frequency of group size (measured in building units), according to the rules described in section 5.1.

Figure 12. Visualization of the aggregated heat demand (building groups). Background map: (LGV Hamburg 2016)

7. Conclusion and outreach

The presented algorithm was designed to meet certain requirements – foremost that building groups have no less than five dwelling units, with non-residential buildings each constituting one single unit. These are requirements specific to the German context and Hamburg in particular. However, by adjusting the way the number of units per building is estimated, the algorithm can accommodate a large number of different requirements. Moreover, we presented how additional constraints can be introduced – for example, buildings in a single plot being always in the same building group.

We also discussed the issue of representing the defined building groups by creating a new geometry. This is an important part of the aggregation problem, since the lack of a representation geometry can lead to difficulties in map usage.

With the increase in big data, data mining and open data, protecting privacy in public maps and datasets is gaining importance. Despite rising concerns about data-use violations, we believe energy policy should be based on quantifiable characteristics. The difficult but important task is to find the balance between protecting privacy and retaining data usability. This paper is an effort in this direction. There is of course room for improvement. The defined rules for the number of units per building are highly simplified and generalized. They can be improved if different strategies for different building types are formulated. For example, data on publicly owned buildings may be considered as not requiring the same extent of data protection as privately owned buildings. The municipality, being the owner, can agree to make this data public. To take account of this circumstance, our algorithm could be modified to assign a value of five units to publicly owned buildings. Then each public building will have enough units to constitute a building group on its own and be practically non-aggregated, although technically being defined as a building group.

On the visualization side, large amount of spatial data nowadays include the third dimension. Representing the heat demand of building groups in a 3D visualization is a further challenge to be explored.

Publication bibliography

Danish Ministry of the Environment (2002): Copenhagen Charter 2002. A Statement on the Occasion of the Danish Presidency. European Cities in a Global Era – Urban Identities and Regional Development. Available online at http://naturstyrelsen.dk/media/nst/attachments/92328/era_charter_uk.pdf, checked on 4/11/2018.

Duckham, Matt; Kulik, Lars; Worboys, Mike; Galton, Antony (2008): Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. In *Pattern Recognition* 41 (10), pp. 3224–3236. DOI: 10.1016/j.patcog.2008.03.023.

Düster, Horst (2011): aggregatePolygons. Version 1.0. github: Düster, Horst. Available online at https://github.com/hdus/pgtools/blob/master/aggregatepolygons/aggregatePolygons.sql, checked on 3/26/2018.

Edelsbrunner, H.; Kirkpatrick, D.; Seidel, R. (1983): On the shape of a set of points in the plane. In *IEEE Trans. Inform. Theory* 29 (4), pp. 551–559. DOI: 10.1109/TIT.1983.1056714.

Federal Ministry for Economic Affairs and Energy (2013): New energy certificates for residential buildings. Available online at https://www.bmwi.de/Redaktion/DE/Infografiken/energieausweis.html, checked on 4/10/2018.

GEWISS Project (2018): GEWISS Geographisches Wärmeinformations- und Simulationssystem [Geographical Heat Information and Simulation System]. Hamburg. Available online at http://gewiss.haw-hamburg.de/, updated on 5/16/2018.

Gillies, Sean; Tonnhofer, Oliver; Arnott, Joshua; Wasserman, Jacob; Bierbaum, Aron; Adair, Allan et al. (2018): Shapely. Manipulation and analysis of geometric objects in the Cartesian plane. Version 1.2.18. Available online at https://shapely.readthedocs.io/en/latest/.

Jones, Eric; Oliphant, Travis; Peterson, Pearu; et al. (2001): SciPy: Open Source Scientific Tools for Python. Available online at http://www.scipy.org/.

Kruskal, J. B. (1956): On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proceedings of the American Mathematical Society* (7), pp. 48–50. DOI: 10.1090/S0002-9939-1956-0078686-7.

LGV Hamburg (2016): Digital Orthophotos 20cm. Hamburg Geoportal: Landesbetrieb Geoinformation und Vermessung. Available online at https://www.geoportal-hamburg.de/Geoportal/geo-online/, checked on 4/10/2018.

McConnell, Jeffrey J. (2006): Computer graphics. Theory into practice. Boston, London: Jones and Bartlett Publishers.

Oliphant, Travis E (2006): A guide to NumPy. USA: Trelgol Publishing. Available online at http://www.numpy.org/.

Openshaw, Stan (1984): The modifiable areal unit problem. Norwich: Geo Books (Concepts and techniques in modern geography, 0306-6142, no.38).

QGIS Development Team (2018): QGIS Geographic Information System. Version 2.18.17: Open Source Geospatial Foundation Project. Available online at http://qgis.osgeo.org.

Reinhart, Christoph F.; Cerezo, Carlos Davila (2016): Urban building energy modeling – A review of a nascent field. In *Building and Environment* 97, pp. 196–202. DOI: 10.1016/j.buildenv.2015.12.001.

Santilli, Sandro (2009): BufferParameters: GEOS Geometry Engine Open Source. Available online at https://svn.osgeo.org/geos/trunk/src/operation/buffer/BufferParameters.cpp, checked on 3/27/2018.