

UiO : **University of Oslo**

Daniel Rygh Bakkelund

# **Order Preserving Hierarchical Clustering**

**Thesis submitted for the degree of Philosophiae Doctor**

Department of Informatics  
Faculty of Mathematics and Natural Sciences

DataScience@UiO

SIRIUS Centre for Scalable Data Access



**2022**

© Daniel Rygh Bakkeland, 2022

*Series of dissertations submitted to the  
Faculty of Mathematics and Natural Sciences, University of Oslo  
No. 2532*

ISSN 1501-7710

All rights reserved. No part of this publication may be reproduced or transmitted, in any form or by any means, without permission.

Cover: Hanne Baadsgaard Utigard.

Print production: Representralen, University of Oslo.

*To Pauline:*

*Why is a raven like a writing desk?*

*The Mad Hatter; Lewis Carroll*

*"Alice's Adventures in Wonderland" (1865)*



# Acknowledgements

This pamphlet describes the fruits of my academic labours over the last five years. I say labourious, but I could say joyful, illuminating, fantastic, maturing, inspiring, educating, awesome, contemplative and many other words, of which none can possibly fully describe the utter privilege. The privilege of getting up every morning, knowing that you shall immerse yourself deeply into the things you love the most. And that you shall do the same thing tomorrow, and the day after, and the day after. Sure, it has been unnerving at times, but without that, this journey would not have been half as grand.

This is therefore, in a sense, a sad day; marking the end of an era of my life.

First and foremost, I would like to thank my advisors, Henrik Forssell and Gudmund Hermansen, who have followed me for the last half of my PhD. They have complemented each other perfectly; Henrik's scrutinising eye and attention to detail on one side, and Gudmund's ever returning questions regarding "why do you do that?" and "what do you really mean with this?" on the other. I hope, and believe, this has led to a style of exposition that is both rigorous, correct and easily understandable.

My thanks to Evgenij Thorstensen and Martin Giese, who were my advisors for the first half of my PhD. Sometimes, a detour is required to understand where you are going.

A special thanks goes to the DataScience@UiO research cluster set up by Ingrid Glad and Arild Waaler, and to all the good conversations we had at our gatherings. My thanks to Arnaldo Frigessi, for his continuous encouragement and support; and to Carlo Mannino, with which I had in particular one meeting that was pivotal to the course of my research.

My thanks to TechnipFMC, my employer outside the university, for allowing me the opportunity of leave over these years, and for supporting my efforts in pursuing the industry problem that inspired this PhD in the first place. An example to be followed.

And finally, of course, there is my family; my darling Rigmor and my fantastic daughter Pauline, who have endured me all this time: Without your everlasting patience, persisting support and unconditional love, I could never have completed this project.

• Daniel Rygh Bakkelund

Oslo, June 2022

---

I *could* also thank my PhD colleagues Lars Tveito and Sigurd Kittilsen for their relentless efforts to get me off track by pointing at every mathematical riddle and unsolved math problem there is. Luckily, I resisted most of their attempts.



# List of Papers

## Paper I

Daniel Bakkelund ‘Order preserving hierarchical agglomerative clustering’.

In: *Machine Learning*. (2021),

DOI: 10.1007/s10994-021-06125-0.

## Paper II

Daniel Bakkelund ‘An objective function for order preserving hierarchical clustering’.

*Submitted for publication.*

*Preprint available at the arXiv: 2109.04266*

## Paper III

Daniel Bakkelund ‘Machine part data with part-of relations and part dissimilarities for planted partition generation’.

In: *Data in Brief*. (2022)

DOI: 10.1016/j.dib.2022.108065.





# Contents

Acknowledgements	iii
List of Papers	v
Contents	vii
<b>1 Introduction</b>	<b>1</b>
1.1 Layout of the thesis . . . . .	2
1.2 Motivating industry problem . . . . .	2
1.3 Order relations, hierarchical clusterings and ultrametrics . . . . .	3
1.4 Order preserving flat clustering . . . . .	11
1.5 Summary of Papers . . . . .	12
1.6 Discussion . . . . .	14
References . . . . .	19
<b>Papers</b>	<b>24</b>
<b>I Order preserving hierarchical agglomerative clustering</b>	<b>25</b>
I.1 Introduction . . . . .	26
I.2 Background . . . . .	34
I.3 Optimised hierarchical clustering . . . . .	38
I.4 Order preserving clustering . . . . .	41
I.5 Partial dendrograms . . . . .	45
I.6 Hierarchical clustering of ordered sets . . . . .	49
I.7 Polynomial time approximation . . . . .	53
I.8 Demonstration of approximation efficacy on randomly generated data . . . . .	54
I.9 Demonstration on data from the parts database . . . . .	63
I.10 Summing up . . . . .	71
I.A Plots from the part database demo . . . . .	73
I.B Reference implementation . . . . .	78
References . . . . .	78
<b>II An objective function for order preserving hierarchical clustering</b>	<b>83</b>
II.1 Introduction . . . . .	83
II.2 Order preserving hierarchical clustering . . . . .	91
II.3 An objective function for trees over ordered data . . . . .	97
II.4 Properties of $g$ . . . . .	99

## Contents

---

II.5	Properties of $f = s_d + g$ . . . . .	108
II.6	Approximation . . . . .	113
II.7	Demonstration . . . . .	118
II.8	Summary and future work . . . . .	124
	References . . . . .	126
<b>III</b>	<b>Machine part data with part-of relations and part dissimilarities for planted partition generation</b>	<b>131</b>
III.1	Data specification . . . . .	132
III.2	Value of the Data . . . . .	133
III.3	Data Description . . . . .	134
III.4	Experimental design, materials and methods . . . . .	137
	References . . . . .	139
<b>A</b>	<b>Correction of Theorem I.4.6</b>	<b>141</b>

# Chapter 1

## Introduction

Clustering is the art of placing things that belong together together, and keeping things that should be kept apart apart. It is both one of the oldest, and one of the most frequently used, tools for exploratory data analysis and unsupervised classification. Indeed, some of today's most popular methods are almost identical to those described by the pioneers, such as Macqueen's  $k$ -means for flat clustering [16], or the initial model for hierarchical agglomerative clustering by Johnson [12]. And the toolbox of clustering methods is still growing, with new research being pushed to the arXiv on a weekly basis.

Over the last decades, a number of methods for clustering data organised in graphs or networks have seen the light of day, where the graph- or network structure contributes to determine the clustering. The prototypical example is clustering of community networks [8]. In community networks, vertices are individuals, and edges represent social bonds. During clustering, parts of the network with high local connectivity are collapsed to clusters. Although the original structure of the network is lost, the semantic information is maintained; the socially closely connected individuals are now gathered in clusters.

Partially ordered data is also relational, and can be represented as directed acyclic graphs or networks. The semantic meaning of a partial order varies with the context: In number theory,  $a \leq b$  can mean *a is less than or equal to b*; in a project plan or a computer program, it can mean *a must be completed before b can start*; or, as in the motivating industry problem of this thesis, where we are concerned with composite machine parts, it means that *a is a part of b*. In either case,  $a \leq b$  indicates, in some sense, that *a precedes b*, and often, when that is the case, then  $a$  and  $b$  are different.

Hence, while relations in a community network indicate individuals belonging together, partial order relations indicate that elements are different and should *not* be placed together when clustering. Moreover, we wish to transfer this information to the clustering; that is, if  $[a]$  and  $[b]$  denotes the respective clusters of  $a$  and  $b$ , then we wish for  $[a]$  to precede  $[b]$ . Concretely, we require the existence of a partial order  $\leq'$  on the set of clusters so that

$$a \leq b \Rightarrow [a] \leq' [b]. \tag{1.1}$$

This differs from the case with community networks in that we do not replace relations with clusters. Rather, we preserve the relations and transfer them to the clustering.

Methods for order preserving clustering of partially ordered sets exist to some extent, but they are usually tailored for very particular purposes and aim to satisfy domain- and application specific constraints. Also, they do not easily combine with the notions of similarity we use for clustering in data analysis. It

## 1. Introduction

---

is the aim of the present work to start closing this gap, and to make this class of methods available for more general applications.

### 1.1 Layout of the thesis

The thesis is structured as follows: Section 1.2 describes the real-world industry problem that motivated the research project. Section 1.3 recalls partial order relations and hierarchical clustering, as well as the definition of order preserving flat clustering that we find in order theory. Section 1.5 provides the abstracts of the published and submitted papers. Section 1.6 discusses the contributions of the papers in light of a comparative analysis. Some further research topics are also identified. The papers themselves are presented in the sections Paper I, Paper II and Paper III.

### 1.2 Motivating industry problem

The inspiration for this work stems from a real-world industry problem concerning a database of machine parts at TechnipFMC<sup>1</sup>, one of the leading technology providers in the energy industry. The items in the database represent blueprints for pieces of mechanical machinery, and the blueprints are linked together with part-of relations, indicating that one piece of machinery is a physical constituent of another. In the same way that a particular type of engine is a part of a particular type of car. The result is a database of machinery that consists of parts that consists of parts that consists of parts and so on. And, since an engine cannot contain itself as a sub-part [18], the structure of all the blueprints together constitute a directed acyclic graph, or, equivalently, a partially ordered set.

Over time, due to business reasons, some blueprints have been subject to copy-paste with minimal modifications. Historically, there has been no tracking of the copy-paste relations, but it is now desirable to recover these links to identify essentially equivalent machinery. The main driver for this need is a large amount of machinery in stock; typically spares that have been produced for the sake of preparedness, but that were never required. This is known as an excess inventory problem [19], and is a recognised type of problem in many industries.

One way of reducing excess inventory, is to identify equivalent machinery, so that when an existing customer comes to asks for a replacement for a previously purchased component, and if there is no such component currently in stock, then the supplier may search for equivalent machinery in stead of making new. Since the production of large steel components comes with both a fiscal and a carbon footprint, this is a double up-side compared to the alternative, namely to eventually decimate the excess inventory.

To tackle this problem, there is already a project running in the company, using machine learning to identify equivalent machinery designs. While being

---

<sup>1</sup><https://www.technipfmc.com/>

very successful, with millions of dollars saved annually, there are still challenges. One of these is that the available metadata is often very similar between a part and its sub-part, leading to parts and sub-parts often being more similar than a part and its copy.

This is where order preserving clustering comes to the rescue: If we cluster equivalent machinery, while at the same time preserving the part-of relations, we will eliminate the problem of similarities between parts and sub-parts. Moreover, the part-of relations, when transferred to the clusters, can tell us which of the clusters that contain parts that are likely to be valid sub-parts of other clusters' parts. A simplified view of the situation is depicted in Figure 1.1, where we display clustering scenarios of the four element set  $\{a, b, c, d\}$  where  $a$  is a part of  $b$  and  $c$  is a part of  $d$ . As we see from the figure, all the order preserving clusterings keep the part and sub-part apart.

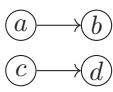
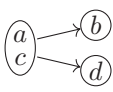
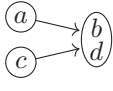
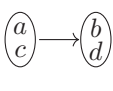
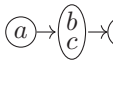
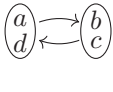
 <p>1. The original data. <math>a</math> is a part of <math>b</math> and <math>c</math> is a part of <math>d</math>.</p>	 <p>2. <math>a</math> and <math>c</math> are equivalent, and both can be sub-parts of both <math>b</math> and <math>d</math>, allowing for spare part interchange.</p>
 <p>3. <math>b</math> and <math>d</math> are equivalent, and both contain <math>a</math> and <math>c</math> as sub-parts.</p>	 <p>4. <math>a</math> and <math>c</math> are equivalent, and <math>b</math> and <math>d</math> are equivalent. It is likely that one pair is a copy of the other.</p>
 <p>5. <math>b</math> and <math>c</math> are equivalent, and both <math>a</math> and <math>b</math> are sub-parts of <math>d</math>. <math>b</math> and <math>c</math> can be interchanged as spare parts in <math>d</math>.</p>	 <p>6. Not a valid clustering since the induced part-of relations are cyclic.</p>

Figure 1.1: A selection of possible clusterings of the set  $\{a, b, c, d\}$  where  $a$  is a part of  $b$  and  $c$  is a part of  $d$ . Possible interpretations in terms of the motivating use case are given together with the clusterings. All but 6) are examples of order preserving clusterings. In 6), the part-of relations constitute a cycle, implying that the parts are proper sub-parts of themselves, which is a contradiction [18]. *Note: the figure is the same as Fig.II.1 in Paper II.*

### 1.3 Order relations, hierarchical clusterings and ultrametrics

For sets  $A$  and  $B$ , we write  $A \subset B$  to denote the usual subset relation. If  $A$  is a proper subset of  $B$ , we write  $A \subsetneq B$ .

#### 1.3.1 Order relations

This section gives a concise recollection of the required concepts of order theory. For a comprehensive introduction, see [21].

## 1. Introduction

---

A binary relation on a set  $X$  is a subset  $E \subset X \times X$ , and we say that  $x$  and  $y$  are **related under  $E$**  if  $(x, y) \in E$ . We commonly write  $xEy$  to denote this fact. We say that  $E$  is **transitive** if

$$\forall x, y, z \in X : xEy \wedge yEz \Rightarrow xEz,$$

and the **transitive closure** of  $E$  is the smallest transitive binary relation  $E'$  on  $X$  for which  $E \subset E'$ .

Now, a **partially ordered set** is a pair  $(X, \leq)$ , where  $X$  is a set, and  $\leq$  is a binary relation on  $X$  for which the following properties hold:

- po1.**  $\forall x \in X : x \leq x$  (reflexivity),
- po2.**  $\forall x, y \in X : x \leq y \wedge y \leq x \Rightarrow x = y$  (antisymmetry),
- po3.**  $\forall x, y, z \in X : x \leq y \wedge y \leq z \Rightarrow x \leq z$  (transitivity).

On the other hand, a **strictly partially ordered set** is a pair  $(X, <)$  where  $<$  is a binary relation satisfying

- spo1.**  $\forall x \in X : x \not< x$  (irreflexivity).
- spo3.**  $\forall x, y, z \in X : x < y \wedge y < z \Rightarrow x < z$  (transitivity).

Notice that spo1 and spo3 together imply

- spo2.**  $\forall x, y \in X : x < y \Rightarrow y \not< x$  (asymmetry),

We will refer to both strict and non-strict partial order relations as **orders**, unless there is any reason to distinguish the two. Also, we will stick to using the symbol  $\leq$  for orders, unless there is any risk of ambiguity.

So, if  $(X, \leq)$  is an ordered set, we say that  $x, y \in X$  are **comparable (under  $\leq$ )** if  $x \leq y$  or  $y \leq x$ ; otherwise, we say that they are **incomparable**. If all pairs of elements of  $X$  are comparable under  $\leq$ , we say that  $(X, \leq)$  is **linearly ordered**. A **chain** in  $(X, \leq)$  is a subset of  $X$  that is linearly ordered with respect to  $\leq$ . We say  $x_0 \in X$  is **the least element of  $(X, \leq)$**  if  $x_0$  is the unique element in  $X$  for which  $y \in X - \{x_0\} \Rightarrow x_0 \leq y$ , and, dually, that  $y_0 \in X$  is **the greatest element of  $(X, \leq)$**  if  $y_0$  is the unique element in  $X$  for which  $x \in X - \{y_0\} \Rightarrow x \leq y_0$ . Finally, a map  $f : (X, \leq) \rightarrow (Y, \leq')$  between ordered sets is an **order preserving map** if

$$\forall x, y \in X : x \leq y \Rightarrow f(x) \leq' f(y).$$

Looking back at the machine parts of Section 1.2, it is clear that the part-of relation is a strict partial order. In particular, a part cannot have itself as a sub-part; that is,  $x \not\leq x$ . This is the focus of Paper I, where we construct a method for hierarchical clustering of strictly partially ordered data. But, as we find in that paper, this constraint comes with some challenges. In Paper II, we therefore consider the part-of relation to be a (non-strict) partial order relation. This can be seen as a relaxation of the strict partial order, because we are now allowed  $x \leq x$ . This relaxation gives us some leeway that we can use to our avail.

### 1.3.2 Hierarchical clustering

Before we define hierarchical clustering, we need to recall flat clustering.

Recall that a **partition** of a set  $X$  is a collection of disjoint subsets  $\{A_i\}_{i=1}^n$  of  $X$  which union covers  $X$ . We refer to the elements  $A_i$  of the partition as **blocks**. In the context of this work, a **(flat) clustering of  $X$**  is a partition of  $X$ , and a **cluster** in a clustering corresponds to a block in the partition. In particular, we do not consider overlapping clusters in this thesis. We write  $[x]_{\mathcal{C}}$  to denote the cluster under  $\mathcal{C}$  containing  $x$ , possibly skipping the subscript if there is no risk of ambiguity. And the map  $q : X \rightarrow \mathcal{C}$ , defined by  $q(x) = [x]_{\mathcal{C}}$ , that sends every element to its cluster, is the **quotient map** corresponding to  $\mathcal{C}$ . We denote **the set of all (flat) clusterings over  $X$**  by  $\mathfrak{Cl}(X)$ .

We now turn to hierarchical clustering. As it turns out, hierarchical clustering fits very well together with order theory. Not too many works take on this perspective; a notable exception is that of Janowitz [10], giving a wholly order theoretic description of the concept.

Given two clusterings  $\mathcal{A}, \mathcal{B} \in \mathfrak{Cl}(X)$ , we say that  **$\mathcal{A}$  refines  $\mathcal{B}$**  if, for every cluster  $A \in \mathcal{A}$ , there exists a cluster  $B \in \mathcal{B}$  for which  $A \subset B$ . Refinement is a partial order relation on  $\mathfrak{Cl}(X)$ , and we denote this relation by  $\mathcal{A} \sqsubset \mathcal{B}$ . There are two particular clusterings that require our attention. The first is the **singleton clustering**  $S(X) = \{\{x\}\}_{x \in X}$ , where each element of  $X$  is placed in a cluster on its own. And the second is the **trivial clustering**  $\{X\}$ , where all elements of  $X$  are placed in the same cluster. Notice that  $S(X)$  refines every clustering in  $\mathfrak{Cl}(X)$ , while every clustering in  $\mathfrak{Cl}(X)$  refines  $\{X\}$ . We have already said that  $(\mathfrak{Cl}(X), \sqsubset)$  is a partially ordered set. Recall that a chain in a partially ordered set is a subset that is linearly ordered.

**Definition 1.3.1.** A **hierarchical clustering** over a set  $X$  is a chain in  $(\mathfrak{Cl}(X), \sqsubset)$  that contains both  $S(X)$  and  $\{X\}$ .

That is, a hierarchical clustering is a sequence of flat clusterings on the form

$$S(X) \sqsubset \cdots \sqsubset \{X\}.$$

In the research literature, we find that hierarchical clustering is also defined

- without requiring  $S(X)$  to be part of the hierarchical clustering [11],
- without requiring  $\{X\}$  to be part of the hierarchical clustering [Paper I],
- without requiring either  $S(X)$  or  $\{X\}$  to be part of the hierarchical clustering [3],

but the classical definition includes both.

The set  $\mathfrak{Cl}(X)$  forms a lattice under refinement, with  $S(X)$  as least element and  $\{X\}$  as greatest element. Figure 1.2 displays the Hasse diagram of this lattice for the set  $\{a, b, c\}$ , together with an indicated hierarchical clustering.

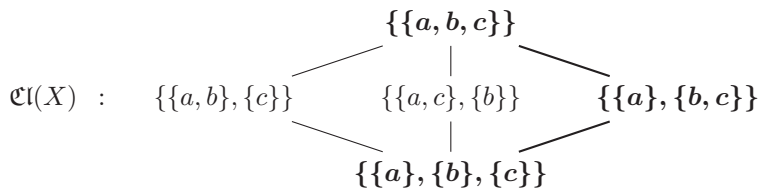


Figure 1.2: The lattice of clusterings of  $\{a, b, c\}$  arranged under refinement makes up a complete lattice. The elements in bold make up a chain in  $(\mathfrak{C}(X), \sqsubset)$  that contains both the least and greatest elements, and therefore also constitutes a hierarchical clustering of  $X$ .

### 1.3.3 Three main approaches to hierarchical clustering

It is customary to label methods for hierarchical clustering according to their generative process; that is, how the methods go forth in specifying the hierarchical clustering. The two main classes are agglomerative and divisive methods. A third important class is that of algorithm free methods, that do not specify how a hierarchical clustering should be obtained, but rather states what makes a good hierarchical clustering.

#### 1.3.3.1 Agglomerative hierarchical clustering

Agglomerative methods are the oldest methods, and also to some extent the origin, of hierarchical clustering. They are also, by far, the most popular methods for hierarchical clustering in applications. The earliest applications are found in biology, dating as far back as 1948, to a paper by the Danish biologist T. A. Sørensen [22]. The classical (and current) approach to agglomerative hierarchical clustering is usually attributed to the 1967 paper by Johnson [12]. The mathematical foundations were laid down by Nicholas Jardine and Robin Sibson in their seminal book *Mathematical Taxonomy* from 1971, and gave rise to a field of its own, carrying the very same name as their book [11].

If  $\mathbb{R}_+$  denotes the non-negative reals, the classical algorithm for agglomerative hierarchical clustering makes use of a **dissimilarity measure**, a function  $d : X \times X \rightarrow \mathbb{R}_+$  for which

- m1.**  $\forall x \in X \quad : \quad d(x, x) = 0$  (positive definitiveness),
- m2.**  $\forall x, y \in X \quad : \quad d(x, y) = d(y, x)$  (symmetry).

that provides us with a means for measuring the difference between elements of  $X$ . The algorithm goes as follows:

1. Start by generating the singleton clustering  $S(X)$ .



2. Of all clusters, find the two that are most similar according to the dissimilarity measure, and replace the two clusters by their union.
3. If we have only one cluster left, we are done. Otherwise, go to Step 2.

The sequence of generated clusters constitutes a hierarchical clustering: the process starts with the singleton partition, and since the merge in Step 2 induces a refinement, the sequence of clusters is a sequence of refinements. The process stops when we reach the trivial clustering.

Now, for this to work, we need a method for comparing *clusters* using a dissimilarity measure. The tool for this is a linkage function:

**Definition 1.3.2.**<sup>2</sup> Let  $\mathcal{P}(X)$  denote the power-set of  $X$ , and let the set of all dissimilarity measures over  $X$  be denoted by  $\mathcal{M}(X)$ . A **linkage function** is a map

$$\mathcal{L} : \mathcal{P}(X) \times \mathcal{P}(X) \times \mathcal{M}(X) \rightarrow \mathbb{R}_+$$

so that for every dissimilarity  $d \in \mathcal{M}(X)$  and every clustering  $\mathcal{C} \in \mathfrak{Cl}(X)$ , the restriction  $\mathcal{L}|_{\mathcal{C} \times \mathcal{C} \times \{d\}}$  is a dissimilarity measure on  $\mathcal{C}$ .

The classical linkage functions are defined as follows. The elements  $p$  and  $q$  are clusters in some clustering  $\mathcal{C}$ :

**Single linkage** :  $\mathcal{SL}(p, q, d) = \min_{x \in p} \min_{y \in q} d(x, y).$

**Complete linkage** :  $\mathcal{CL}(p, q, d) = \max_{x \in p} \max_{y \in q} d(x, y).$

**Average linkage** :  $\mathcal{AL}(p, q, d) = \frac{\sum_{x \in p} \sum_{y \in q} d(x, y)}{|p| \cdot |q|}.$

The output from agglomerative hierarchical clustering is often depicted in terms of a graphical dendrogram like the one in Figure 1.3.

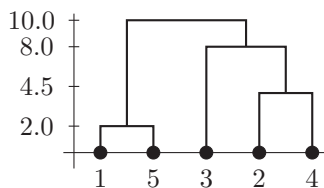


Figure 1.3: Example dendrogram from classical agglomerative hierarchical clustering over the set  $\{1, 2, 3, 4, 5\}$ . The horizontal bars indicate joined clusters, and are drawn at heights corresponding to the dissimilarity of the clusters.

In Figure 1.3, the leaves the tree correspond to the elements of  $X = \{1, 2, 3, 4, 5\}$ . A diagram like this gives rise to an **ultrametric** over  $X$ ; that is a dissimilarity measure that also satisfies

---

<sup>2</sup>Linkage functions are rarely defined formally like this, but rather just through their formulae. Definition 1.3.2 is akin to the definition provided by Carlsson and Mémoli [2].

**u3.**  $\forall x, y, z \in X : d(x, z) \leq \max\{d(x, y), d(y, z)\}$  (ultrametric inequality).

Since u3 implies the triangle inequality  $d(x, z) \leq d(x, y) + d(y, z)$ , it follows that an ultrametric is also a metric in the classical sense. The ultrametric distance between  $x$  and  $y$ , given by the graphical dendrogram, corresponds to the height in the graphical dendrogram you must go to in order to traverse from  $x$  to  $y$ . For example, for the dendrogram in Figure 1.3, the ultrametric distance between 2 and 5 is 10.0, and the distance between 2 and 3 is 8.0.

As it turns out, through this definition, there is a one-to-one correspondence between the set of dendrograms over  $X$  and the set of ultrametrics over  $X$ . That is, if  $\mathcal{D}(X)$  is the family of dendrograms over  $X$ , and if  $\mathcal{U}(X)$  is the family of ultrametrics over  $X$ , then there exists a bijective map

$$\Psi_X : \mathcal{D}(X) \longrightarrow \mathcal{U}(X) \tag{1.2}$$

between dendrograms and ultrametrics [2].

### 1.3.3.2 Divisive hierarchical clustering

Divisive methods for hierarchical clustering are of much newer origin, and the first proper contribution is that of Kaufman and Rousseeuw from 1990 [13, Ch.6]. After that, a large body of research has contributed with new methods and techniques. The idea behind divisive hierarchical clustering is very intuitive, and usually progresses according to the below algorithm.

1. Start with the trivial clustering  $\{X\}$ .
2. Find the largest cluster and split it in two.
3. If all clusters have cardinality one, we are done. Otherwise, go to Step 2.

This process is essentially the reverse of agglomerative hierarchical clustering, starting with the trivial clustering and ending up with the singleton clustering. Each invocation of Step 2 produces a refinement of the current clustering, so the result is a chain of refinements, or, equivalently, a hierarchical clustering. Notice that the process produces a hierarchical clustering that can be drawn as a binary tree, since we always split a set in two. We denote the family of binary trees over  $X$  that can be generated in this way by  $\mathfrak{B}(X)$ .

In order to make the above algorithm useful, Step 2 requires a function

$$\text{split} : \mathcal{P}(X) \rightarrow \mathcal{P}(X) \times \mathcal{P}(X),$$

that splits a cluster in two in a meaningful way. The exact implementation of the split function is what distinguishes the different methods for divisive hierarchical clustering. Some split functions use a dissimilarity measure to decide on a good split, in which case the usual idea is that different elements should be split apart, and similar elements should be kept together.

For example, the DIANA method of Kaufman and Rousseeuw picks the element in the cluster of maximum mean dissimilarity to the others, and moves along with this element all the other elements that is more similar to the new cluster than to the old [13, Ch.6].

Another common approach to splitting a cluster is that of optimising a *cut*. The idea is to find a split  $C \mapsto (A, B)$  of the cluster  $C$  that optimises some objective function; for example, the MAX-CUT [4] aims find a split  $C \mapsto (A, B)$  maximising the sum

$$\sum_{a \in A} \sum_{b \in B} d(a, b).$$

Since this way of thinking is closely related to cuts in graphs, divisive methods for hierarchical clustering have become popular for clustering of graph based data, such as community networks.

In divisive hierarchical clustering, the focus is on the splits generating the hierarchy, and graphical representations are sometimes drawn as in Figure 1.4.

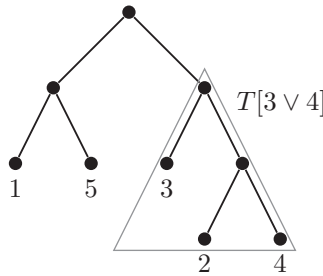


Figure 1.4: A tree  $T$  of splits of the set  $X = \{1, 2, 3, 4, 5\}$  that constitutes a hierarchical clustering of  $X$ .

*Note: The figure is the same as Fig.II.2 of Paper II.*

Not all methods for divisive hierarchical clustering come with numerical values on the splits that give rise to ultrametrics like that of the graphical dendrogram in Figure 1.3. However, there are ultrametrics that only rely on the topology of the tree, and can be computed for any hierarchical clustering. One example is the following ultrametric due to Roy and Pokutta [20]. Before we define the ultrametric, for a tree  $T \in \mathfrak{B}(X)$ , let  $T[x \vee y]$  denote the deepest internal node in the tree which rooted subtree contains both  $x$  and  $y$  as leafs; see Figure 1.4 for an example. We write  $|T[x \vee y]|$  to denote the number of leaf nodes of this subtree.

Now, given a tree  $T \in \mathfrak{B}(X)$ , the ultrametric  $u_T : X \times X \rightarrow \mathbb{R}_+$  is defined as

$$u_T(x, y) = |T[x \vee y]| - 1. \quad (1.3)$$

## 1. Introduction

---

Roy and Pokutta show that  $\mathfrak{B}(X)$  embeds into the set of ultrametrics under this construction; that is, (1.3) gives rise to an injective map  $\mathfrak{B}(X) \rightarrow \mathcal{U}(X)$  defined by  $T \mapsto u_T$ .

### 1.3.3.3 Algorithm free methods

Algorithm free methods are methods that do not provide a recipe for the generation of the hierarchical clustering. Rather, the methods specify properties that a hierarchical clustering should have, and are typically subject for optimisation. Since the number of hierarchical clusterings of a set  $X$  far exceeds  $2^{|X|}$ , many of the methods yield NP-hard optimisation problems. The solution, then, is often a heuristically based algorithm that is either agglomerative or divisive, that can provide a decent hierarchical clustering relative to the optimum.

Dasgupta's objective function for hierarchical clustering from 2016 is one example of this type of method [7]. Dasgupta makes use of a similarity measure<sup>3</sup>  $s : X \times X \rightarrow \mathbb{R}_+$ , and computes the **cost of a binary tree** to be

$$\text{cost}_s(T) = \sum_{\{\{x,y\} \subset X \mid x \neq y\}} |T[x \vee y]|s(x, y).$$

Optimising for this objective function is NP-hard, but Dasgupta provides an approximation algorithm using divisive hierarchical clustering, where the split function is a SPARSESTCUT [14]. That is, a split into subsets  $A, B$  that minimises

$$\frac{\sum_{a \in A} \sum_{b \in B} s(a, b)}{|A||B|}.$$

Now, SPARSESTCUT is also an NP-hard problem, so Dasgupta takes this one step further, and points to a polynomial time approximation of SPARSESTCUT [14], which he eventually uses in his approximation algorithm. Several publications provide further improvements over Dasgupta's model, both agglomerative [17] and divisive [20].

Another family of algorithm free methods for hierarchical clustering, is that of ultrametric fitting. Recall that every hierarchical clustering is in a one-to-one correspondence with an ultrametric (Equation (1.2)). Hence, we can replace the search for a hierarchical clustering by the search for an ultrametric.

Given a dissimilarity or metric  $d \in \mathcal{M}(X)$ , ultrametric fitting is often about solving the optimisation problem

$$\arg \min_{u \in \mathcal{U}(X)} L(d, u),$$

where  $L$  is some loss function to be minimised over  $\mathcal{U}(X)$ . A frequently used loss functions for ultrametric fitting is the  $p$ -norm, measuring the point-wise distance

---

<sup>3</sup>A similarity measure is a symmetric function  $s : X \times X \rightarrow \mathbb{R}_+$  for which pairs of similar elements yield higher function values compared to pairs of dissimilar elements. For example, given a dissimilarity measure  $d$  bounded by  $M > 0$ , the function  $s = M - d$  is a similarity measure.

between the original metric  $d$  and the ultrametric  $u$ :

$$L(d, u) = \|d - u\|_p = \sqrt[p]{\sum_{x, y \in X} |d(x, y) - u(x, y)|^p}.$$

The underlying idea is that the initial metric is a good guess, and that we should look for a hierarchical clustering (i.e. ultrametric) that is as close to that metric as possible. In ultrametric fitting, the problem of finding an optimal solution is usually NP-hard, and the contributions in the field are often algorithmically based approximations. See [5, 6] for some recent publications.

## 1.4 Order preserving flat clustering

The theory for order preserving flat clustering of partially ordered sets is previously described in [1, §3]. A parallel theory for order preserving flat clustering for strictly partially ordered sets is set forth in Section I.4 of Paper I. As it turns out, the theories are sufficiently similar for us to continue to stick to the convention of referring to both classes of order relations as orders.

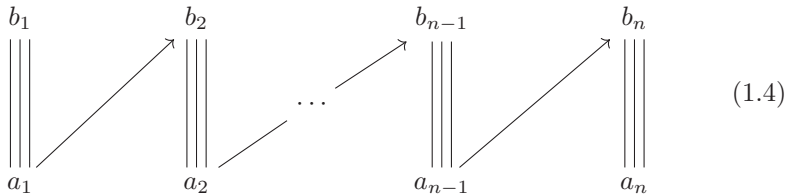
The central question to answer is how to deal with the order relation during clustering.

**Definition 1.4.1.** Let  $(X, \leq)$ , be an ordered set, and let  $\mathcal{C} = \{C_i\}_{i=1}^n$  be a clustering of  $X$ . Define a binary relation  $E$  on  $\mathcal{C}$  as follows:

$$(C_i, C_j) \in E \Leftrightarrow \exists x, y \in X : x \leq y \wedge x \in C_i \wedge y \in C_j.$$

Let  $\leq'$  denote the transitive closure of  $E$ . We refer to the relation  $\leq'$  as **the relation on  $\mathcal{C}$  induced by  $\leq$** . Or just *the induced relation* for short.

An instructive illustration of what the induced relation looks like, is that of a  **$\mathcal{C}$ -fence** [1], or just fence, for short:



Triple lines indicate that elements are in the same cluster in  $\mathcal{C}$ , and the arrows represent comparability in  $(X, \leq)$ . The fence visualises how one can traverse from  $b_1$  to  $a_n$  along arrows and through clusters in  $\mathcal{C}$ , and in that case we say that the fence **links**  $b_1$  to  $a_n$ . The induced relation  $\leq'$  has the property that  $x \leq' y$  if and only if there exists a  $\mathcal{C}$ -fence linking  $x$  to  $y$ .

There is nothing in Definition 1.4.1 suggesting that the induced relation is an order relation. The following theorem classifies precisely when this is the case.

## 1. Introduction

---

**Theorem 1.4.2.** *Let  $(X, \leq)$  be an ordered set, and let  $\mathcal{C}$  be a clustering of  $X$ . Then the following statements are equivalent:*

1. *The induced relation  $\leq'$  is an order relation on  $\mathcal{C}$ .*
2. *The quotient map  $q : (X, \leq) \rightarrow (\mathcal{C}, \leq')$  is order preserving.*

*Proof.* A proof for partial order relations can be found in [1, Thm.3.1]. For a proof in the case of strict partial orders, see [Paper I,Thm.I.4.3]. ■

Notice that Statement 2. of Theorem 1.4.2 is equivalent to

$$x \leq y \Rightarrow [x]_{\mathcal{C}} \leq' [y]_{\mathcal{C}}.$$

This is precisely what we declared to be order preserving from the semantic perspective, in the discussion preceding equation (1.1).

This brings us more or less exactly to the starting point of the research presented in this thesis. The topic of the research papers (Papers I and II) is to define order preserving hierarchical clustering, and to investigate how we can identify the order preserving hierarchical clusterings that are also good hierarchical clusterings in their own right. That is, to combine the classical ideas of hierarchical clustering with that of order preservation.

Since the topic of order preserving clustering in a data analysis context is a new field, there is little or no publicly available data formatted for this purpose. The thesis therefore also sets forth the data article in Paper III, to contribute to mending this situation.

## 1.5 Summary of Papers

### **Paper I Order preserving hierarchical agglomerative clustering**

*Author:* Daniel Bakkelund

*In:* Machine Learning (2021)

**Abstract:** Partial orders and directed acyclic graphs are commonly recurring data structures that arise naturally in numerous domains and applications and are used to represent ordered relations between entities in the domains. Examples are task dependencies in a project plan, transaction order in distributed ledgers and execution sequences of tasks in computer programs, just to mention a few. We study the problem of *order preserving hierarchical clustering* of this kind of ordered data. That is, if we have  $a < b$  in the original data and denote their respective clusters by  $[a]$  and  $[b]$ , then we shall have  $[a] < [b]$  in the produced clustering. The clustering is similarity based and uses standard linkage functions, such as single- and complete linkage, and is an extension of classical hierarchical clustering.

To achieve this, we develop a novel theory that extends classical hierarchical clustering to strictly partially ordered sets. We define the output from running classical hierarchical clustering on strictly ordered data to be *partial*

*dendrograms*; sub-trees of classical dendrograms with several connected components. We then construct an embedding of partial dendrograms over a set into the family of ultrametrics over the same set. An optimal hierarchical clustering is defined as the partial dendrogram corresponding to the ultrametric closest to the original dissimilarity measure, measured in the  $p$ -norm. Thus, the method is a combination of classical hierarchical clustering and ultrametric fitting.

A reference implementation is employed for experiments on both synthetic random data and real world data from a database of machine parts. When compared to existing methods, the experiments show that our method excels both in cluster quality and order preservation.

## **Paper II An objective function for order preserving hierarchical clustering**

*Author:* Daniel Bakkellund

*In:* Submitted to Journal of Machine Learning Research (JMLR)

**Abstract:** We present an objective function for similarity based hierarchical clustering of partially ordered data that preserves the partial order. That is, if  $(X, \leq)$  is a partially ordered set, a clustering of  $X$  is order preserving with respect to  $\leq$  if there exists a partial order  $\leq'$  on the clusters so that if  $x \leq y$  and if  $[x]$  and  $[y]$  are the clusters of  $x$  and  $y$ , then  $[x] \leq' [y]$ . The theory distinguishes itself from existing theories for clustering of ordered data in that the order relation and the similarity are combined to obtain a hierarchical clustering seeking to satisfy both. In particular, the order relation is weighted in the range  $[0, 1]$ , and if the similarity and the order relation are not aligned, then order preservation may have to yield in favor of clustering. Finding an optimal solution is NP-hard, so we provide a polynomial time approximation algorithm, with a relative performance guarantee of  $O(\log^{3/2} n)$ , based on successive applications of directed sparsest cut. We provide a demonstration on a benchmark dataset, showing that our method outperforms existing methods for order preserving hierarchical clustering with significant margin. The theory is an extension of Dasgupta's objective function for hierarchical clustering.

## **Paper III Machine part data with part-of relations and part dissimilarities for planted partition generation**

*Author:* Daniel Bakkellund

*In:* Submitted to Data in Brief

**Abstract:** Identifying relationships between entities in data is a central topic across various industries and businesses, from social networks to supply chain and heavy manufacturing industries. In this paper we present data from a database of machinery represented in terms of machine parts. The machine parts are originally organised in tree structures where the vertices are machine part types, and the edges are "part-of" relations. Hence, each tree represents a type of machinery broken down into its machine

part constituent types. The data we present is the union over these trees, making up a directed acyclic graph describing the type hierarchy of the machine parts.

The motivation for publishing the dataset is the following real-world industry problem: Each tree represents a mechanical design, and over time some designs have been copy-pasted with minor modifications. The new instances have been given new identifiers with no reference to where from they were copied. In hindsight, it is desirable to recover the copy-paste links to for interchange between essentially identical designs. However, telling which parts are copies of which other parts has turned out to be difficult. In particular, the metadata has a tendency of displaying higher similarities within a composite part than between a part and its copy. Due to non-disclosure, we cannot provide the metadata, but we provide element wise dissimilarities that are generated based on the metadata using classical methods such as Jaccard similarity on description texts, material types etc. The dissimilarities are obtained from a data science project in the company owning the data, trying to tackle the very problem of recovering the copy-paste links.

Availability of labeled data on this data set is limited, so based on our in-depth knowledge of the problem domain, we present a data synthetisation method that can generate arbitrarily large problem instances of the copy-paste problem based on the sample data, that provides a realistic representation of the real world problem. The problems are presented as planted partitions of vertices of directed acyclic graphs with vertex dissimilarities, and thus constitutes a typical classification problem along the lines of graph- or network clustering.

The type of industry data we present is usually company confidential, bound by intellectual property rights, and generally not available to scientists. We therefore publish this anonymised dataset to offer real world sample data and generated problem instances for researchers that are interested in this type of classification problems, and on which theories and algorithms can be tested.

The data and the problem generation methodology are backed by a Python implementation, providing both data access and an API for parameterised problem generation. The data is also available as raw files.

## 1.6 Discussion

The work presented in this thesis discusses the topic of order preserving hierarchical clustering of partially ordered sets equipped with a measure of similarity or dissimilarity. Concretely, we have worked our way around the difficulties of, on one side, generating a hierarchical clustering based on the similarity or dissimilarity, while simultaneously identifying clusterings that do not violate the partial order relation.



The *modus operandi* has been to first establish a mathematical interpretation of order preserving hierarchical clustering; then, to develop a method for deciding what constitutes a *good* order preserving hierarchical clustering in the presence of a measure of similarity or dissimilarity; and finally, to demonstrate the efficacy of the theory on synthetic and real world data. For both of the presented theories (Papers I and II), we can see from the demonstrations that taking the partial order into account has a pronounced effect on the quality of the generated hierarchical clustering.

The thesis also sets forth the data article in Paper III, based on real world data from the mentioned industry problem. The motivation is to make available data, suited for testing methods and algorithms for order preserving (hierarchical) clustering in a data analysis context. The paper presents both real and synthetic data, and a framework for benchmarking methods for order preserving (hierarchical) clustering.

**Relevance to the industry problem.** Recalling the industry problem from Section 1.2, Papers I and II provide demonstrations that the order preserving hierarchies contain flat clusterings that are better than what is achieved by the methods compared against. But it is also interesting to consider how the concept of order preserving hierarchical clustering fits into the use case of identifying equivalent machinery in terms of ultrametrics.

Theorem II.2.11 of Paper II says that the more elements there are between two elements in a partial order, the more different they are under an ultrametric corresponding to an order preserving hierarchical clustering. Now, consider the engineer looking for machinery that is equivalent to that of type  $x$ . Given the ultrametric, she can sort the inventory with respect to ultrametric distance to  $x$ . At the top of the list, she will find the items in the inventory most similar to  $x$ . And, according to Theorem II.2.11, if an element is several levels of containment inside  $x$ , this element will appear as more and more dissimilar to  $x$ . In particular, this effectively eliminates the problem of accidental part and sub-part similarity due to similar metadata.

**On representations of the order relation.** While Paper I considers the order relation to be a strict partial order relation, Paper II not only replaces the strict relation by a (non-strict) partial order, but goes further to introduce the idea of a *relaxed relation*  $\omega : X \times X \rightarrow [0, 1]$  that does not even have to be an order relation. Moving to the relaxed relation has several benefits, both theoretically and practically:

- **Scaling:** The relaxation  $\omega$  leads to a compatibility with calculus that greatly surpasses the binary representation. We can scale the order relation, and the concept allows for application of standard calculus operations. Without this, the linear combination making up the objective function of Paper II would not be possible.
- **Less bug traps:** In the context of greedy search algorithms, bug-traps are topological features of the search space that causes the search algorithm to

use excessive time; such as a part of a labyrinth with only dead ends. In the same way, the method of Paper I is hampered by all the dead ends in the partial lattice constituting the legal hierarchical clusterings (Paper I, Fig.I.2). Passing to a non-strict partial order removes all the dead ends, since the hierarchical clusterings of a partial order constitutes a complete lattice. Still, the nature of order preserving hierarchical clustering will eventually split comparable elements apart, also in the case of a partial order. For example, the objective function of Paper II favours this over keeping comparable elements inside one cluster. This means that also the model of Paper II is well suited for strict partial orders; one must simply take care to discard the clusterings that are not *strictly* order preserving.

The bug-trap effect is even more profoundly removed with the relaxed relation. In Paper II, we allow clusterings to violate order preservation, but we value order preserving solutions more. Such a model allows for algorithms to explore areas in the search space that are off limit to the stricter cases, and may open up for faster approximations. This is a relaxation in the true meaning of the word.

**Embedding semantic information in machine learning.** Both Paper I and II can be seen as methods for combining logically structured information with classical machine learning techniques. In particular, they manage to satisfy both the semantic interpretation of hierarchical clustering and that of order preservation, but in different ways.

The method of Paper I treats the strict partial order as a logical object, and presents an algorithm that preserves the partial order as an invariant while doing agglomerative clustering. The method works, but it is not obvious how to, for example, add more constraints in addition to the partial order.

The method of Paper II is, as such, more general. The work provides an objective function that will produce order preserving hierarchical clusterings under mild assumptions (Paper II, Thm.II.4.3). However, the order relation is given a quantitative representation that allows it to be combined with the similarity in Dasgupta's objective. This allows us to search for a solution that satisfies both the partial order and the similarity, for example in terms of a Pareto optimal solution.

The method can be extended to incorporate any type of semantic information, as long as the goal of incorporating the semantics can be described in terms of objective functions over splits  $S \rightarrow (A, B)$ , and that the objective functions are invariant under positive scaling.

### 1.6.1 Open questions and future work topics

Below are listed the open main problems that are common to the methods set forth in this work.

**Problem complexity.** An open question of both Papers I and II is that of problem complexity. While both theories yield NP-hard optimisation problems

when the order relation is excluded, and therefore are NP-hard in the worst case, it is not obvious what the problem complexity is when both the similarity and the order relation is present. Nor do we know the complexity class when we remove the similarity or dissimilarity and only the order relation is present. This latter case is possibly related to a class of already described antichain partitioning problems, which complexity classes are still undetermined [15].

**Performance.** For both of the presented methods, performance is a challenge.

In Paper I, since the (strict) partial order relation is treated as an unbreakable constraint, this implies a hard limit on the complexity lower bound, even for the approximation algorithm. Although recent developments in approximate ultrametric fitting show promising results [6], and recent results within acyclic partitioning are promising [9], it is not obvious how to combine these methods to achieve a balancing of the goals of these solutions. This is partly because the referenced solutions are algorithmic, and blending algorithms is generally not straight forward.

The method of Paper II also has issues with performance, and currently does not scale to industrial applications. There are two obvious paths to pursue in this case. The first is that of providing faster approximations of DIRECTEDSPARSESTCUT.<sup>4</sup> This is a cut that has received limited attention in the research community, and it is to hope that it is possible to produce a time-efficient approximation for this cut. The other is that of agglomerative approximation algorithms, and an obvious track to investigate is to produce an agglomerative method using the objective function of Paper II to compute the gain of merging two clusters.

**Flat clustering.** The models set forth by this work are in the realm of hierarchical clustering. It is natural to also consider theories and algorithms for flat order preserving clustering for use in data analysis. This may yield a string of papers or advanced master thesis projects, extending existing methods to partially ordered sets. Technically, the main challenge will most likely be in finding ways of quantifying order preservation in the context of the different methods' objectives.

## 1.6.2 Other possible applications

It is possible to imagine several alternative applications of order preserving clustering, and some have already been mentioned in the papers. Below are mentioned two more; one almost trivial, but very descriptive of the nature of the method: that of sorting segments of a book. The second application, for ranking based recommendation systems, serves to show that in one sense, order preserving hierarchical clustering can be considered as a method for sorting of data according to two different sort orders.

---

<sup>4</sup>Or any approximations, ref. Paper II, Section II.8.

## 1. Introduction

---

**Sorting the segments of a book.** I corresponded briefly with Vijaya Kumari Yeruva, a data analyst and fellow PhD student, who needed to cluster segments of a book while keeping the ordering of the segments intact, as indicated in Figure 1.5.

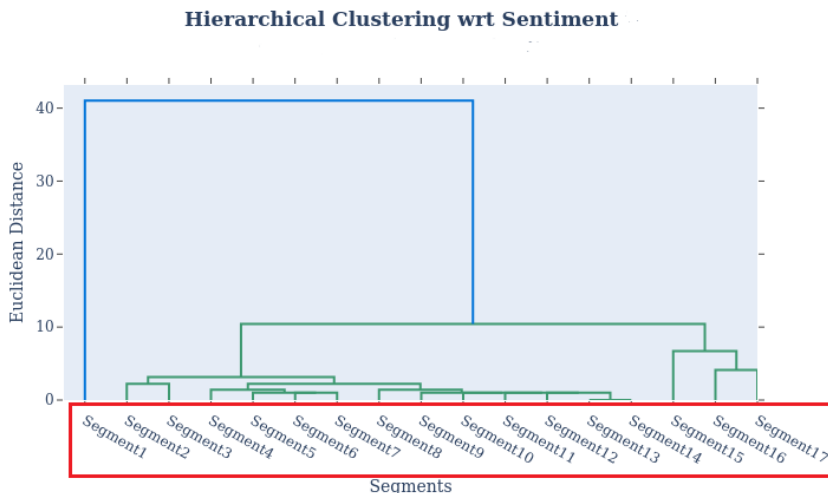


Figure 1.5: An example of order preserving hierarchical clustering of segments of a book. The segments are linearly ordered and cannot be shuffled about. *The figure is downloaded from [stackoverflow<sup>a</sup>](https://stackoverflow.com/questions/70069193/order-preserving-hierarchical-agglomerative-clustering-python) and reprinted here with permission from V. K. Yeruva.*

---

<sup>a</sup><https://stackoverflow.com/questions/70069193/order-preserving-hierarchical-agglomerative-clustering-python>

Yeruva had already attempted to use the `ophac`<sup>5</sup> library for this (`ophac` realises the theory of Paper I), but the result was not very useful. This is as expected, since Paper I treats the order relation as strict, and since the segments of a book constitutes a linear order, the resulting hierarchy will consist of only the singleton- and the trivial clustering. However, by applying the theory from Paper II, you can have both order preservation and non-trivial clustering. Alas, at the time of this writing, Yeruva communicated by e-mail that she had not come to test this out yet.

Now, Yeruva works in the field of sentiment analysis, but we can imagine a simpler application for the sake of example: Given a reasonable similarity measure between segments, an order preserving hierarchical clustering can be used to suggest a division of the book into chapters, sections and subsections.

**Recommender systems.** Consider the task of recommender systems that make use of ranking data. If a user ranks a set of items, this constitutes a

---

<sup>5</sup><https://pypi.org/project/ophac/>

linearly ordered set. And if the user ranks several sets of data, combining all the sets yields a partially ordered set. Assume now that we are given a similarity measure over the items, so that we can say which items are similar. Further, assume that we perform order preserving hierarchical clustering with the ranking information, but *replacing the similarity with its corresponding dissimilarity*. The resulting order preserving tree will induce an ordering of all the ranked items. In particular, due to Lemma II.2.7 of Paper II, the leaf nodes of the hierarchical clustering will be ordered left-to-right so that the right hand elements are 1) the highest ranked items according to the user, and 2) the most dissimilar items, since dissimilar items are placed close together. Hence, from a bird’s-eye view, we can use order preserving hierarchical clustering to offer highly ranked topics of high diversity.

This is not to say that the theories set forth in this work outperforms existing methods, or are even useful, for ranking based recommendation. But it serves to show that the *concept* of order preserving hierarchical clustering may give rise to new methods for deducing recommendations.

## References

- [1] Blyth, T. *Lattices and Ordered Algebraic Structures*. Universitext. Springer London, 2005. URL: <https://www.springer.com/gp/book/9781852339050>.
- [2] Carlsson, G. and Mémoli, F. ‘Characterization, Stability and Convergence of Hierarchical Clustering Methods’. In: *J. Mach. Learn. Res.* vol. 11 (Aug. 2010), pp. 1425–1470. URL: <http://www.jmlr.org/papers/v11/carlsson10a.html>.
- [3] Carlsson, G. and Mémoli, F. ‘Classifying Clustering Schemes’. In: *Foundations of Computational Mathematics* vol. 13, no. 2 (Apr. 2013), pp. 221–252. URL: <https://doi.org/10.1007/s10208-012-9141-9>.
- [4] Chatziafratis, V., Mahdian, M. and Ahmadian, S. ‘Maximizing Agreements for Ranking, Clustering and Hierarchical Clustering via MAX-CUT’. In: *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 1657–1665.
- [5] Chierchia, G. and Perret, B. ‘Ultrametric fitting by gradient descent’. In: *Journal of Statistical Mechanics: Theory and Experiment* vol. 2020, no. 12 (2020).
- [6] Cohen-Addad, V., S., K. C. and Lagarde, G. ‘On Efficient Low Distortion Ultrametric Embedding’. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by III, H. D. and Singh, A. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 2078–2088. URL: <https://proceedings.mlr.press/v119/cohen-addad20a.html>.

## 1. Introduction

---

- [7] Dasgupta, S. ‘A Cost Function for Similarity-Based Hierarchical Clustering’. In: *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*. STOC ’16. Cambridge, MA, USA: Association for Computing Machinery, 2016, pp. 118–127. URL: <https://dl.acm.org/doi/10.1145/2897518.2897527>.
- [8] Fortunato, S. and Hric, D. ‘Community detection in networks: A user guide’. In: *Physics Reports* vol. 659 (2016). Community detection in networks: A user guide, pp. 1–44. URL: <https://www.sciencedirect.com/science/article/pii/S0370157316302964>.
- [9] Herrmann, J. et al. ‘Acyclic Partitioning of Large Directed Acyclic Graphs’. In: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. May 2017, pp. 371–380. URL: <https://hal.inria.fr/hal-01744603>.
- [10] Janowitz, M. F. *Ordinal and Relational Clustering*. WORLD SCIENTIFIC, 2010. eprint: <https://www.worldscientific.com/doi/pdf/10.1142/7449>. URL: <https://www.worldscientific.com/doi/abs/10.1142/7449>.
- [11] Jardine, N. and Sibson, R. *Mathematical Taxonomy*. Wiley series in probability and mathematical statistics. Wiley, 1971.
- [12] Johnson, S. C. ‘Hierarchical clustering schemes’. In: *Psychometrika* vol. 32, no. 3 (1967), pp. 241–254. URL: <https://link.springer.com/article/10.1007/BF02289588>.
- [13] Kaufman, L. and Rousseeuw, P. J. *Finding groups in data : an introduction to cluster analysis*. eng. New York, 1990.
- [14] Leighton, T. and Rao, S. ‘Multicommodity Max-Flow Min-Cut Theorems and Their Use in Designing Approximation Algorithms’. In: *J. ACM* vol. 46, no. 6 (Nov. 1999), pp. 787–832. URL: <https://doi.org/10.1145/331524.331526>.
- [15] Lonc, Z. ‘On complexity of some chain and antichain partition problems’. In: *Graph-Theoretic Concepts in Computer Science*. Ed. by Schmidt, G. and Berghammer, R. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 97–104. URL: [https://doi.org/10.1007/3-540-55121-2\\_9](https://doi.org/10.1007/3-540-55121-2_9).
- [16] Macqueen, J. ‘Some methods for classification and analysis of multivariate observations’. In: *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*. 1967, pp. 281–297. URL: <https://projecteuclid.org/euclid.bsm/1200512992>.
- [17] Moseley, B. and Wang, J. R. ‘Approximation Bounds for Hierarchical Clustering: Average Linkage, Bisecting K-Means, and Local Search’. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 3097–3106.
- [18] Rescher, N. ‘Axioms for the part relation’. In: *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition* vol. 6, no. 1 (1955), pp. 8–11.

- [19] Rosenfield, D. B. ‘Disposal of excess inventory’. In: *Operations research* vol. 37, no. 3 (1989), pp. 404–409.
- [20] Roy, A. and Pokutta, S. ‘Hierarchical Clustering via Spreading Metrics’. In: *Journal of Machine Learning Research* vol. 18, no. 88 (2017), pp. 1–35. URL: <http://jmlr.org/papers/v18/17-081.html>.
- [21] Schröder, B. *Ordered Sets, An Introduction*. Springer Science + Business Media, LLC, 2003. URL: <https://link.springer.com/content/pdf/10.1007/978-3-319-29788-0.pdf>.
- [22] Sørensen, T. A. ‘A Method of Estimating Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content’. In: *Biologiske Skrifter*. Vol. 5. Det Kongelige Danske Videnskabernes Selskab. Kommissionær: Munksgaard, 1948, pp. 1–34.





# Papers



# Order preserving hierarchical agglomerative clustering

Daniel Bakkelund

Published in *Machine Learning*, Dec 2021,  
DOI: 10.1007/s10994-021-06125-0.

## Abstract

Partial orders and directed acyclic graphs are commonly recurring data structures that arise naturally in numerous domains and applications and are used to represent ordered relations between entities in the domains. Examples are task dependencies in a project plan, transaction order in distributed ledgers and execution sequences of tasks in computer programs, just to mention a few. We study the problem of *order preserving hierarchical clustering* of this kind of ordered data. That is, if we have  $a < b$  in the original data and denote their respective clusters by  $[a]$  and  $[b]$ , then we shall have  $[a] < [b]$  in the produced clustering. The clustering is similarity based and uses standard linkage functions, such as single- and complete linkage, and is an extension of classical hierarchical clustering.

To achieve this, we develop a novel theory that extends classical hierarchical clustering to strictly partially ordered sets. We define the output from running classical hierarchical clustering on strictly ordered data to be *partial dendrograms*; sub-trees of classical dendrograms with several connected components. We then construct an embedding of partial dendrograms over a set into the family of ultrametrics over the same set. An optimal hierarchical clustering is defined as the partial dendrogram corresponding to the ultrametric closest to the original dissimilarity measure, measured in the  $p$ -norm. Thus, the method is a combination of classical hierarchical clustering and ultrametric fitting.

A reference implementation is employed for experiments on both synthetic random data and real world data from a database of machine parts. When compared to existing methods, the experiments show that our method excels both in cluster quality and order preservation.

**Keywords:** Hierarchical clustering · Order preserving clustering · Partial dendrogram · Unsupervised classification · Ultrametric fitting · Acyclic partition

### I.1 Introduction

Clustering is one of the oldest and most frequently used techniques for exploratory data analysis and unsupervised classification. The toolbox contains a large variety of methods and algorithms, spanning from the initial, but still popular ideas of  $k$ -means [32] and hierarchical clustering [25], to more recent methods, such as density- and model based clustering [12, 28], and semi-supervised methods [2], plus a large list of variants. All these methods have one thing in common: they try to extract hidden structure from the data, and make it visible to the analyst. But they also share another feature: if the analysed data is already endowed with some form of structure, the structure is lost in the clustering process; the clustering does not try to retain the structure.

In this paper, we show how to extend hierarchical clustering to relational data in a way that preserves the relations. In particular, if the input is a set  $X$  equipped with a strict partial order  $<$ , and if  $a, b \in X$ , we ensure that if  $a < b$  then we will have  $[a] <' [b]$  after clustering, where  $[a]$  and  $[b]$  are the respective clusters of  $a$  and  $b$ , and  $<'$  is a partial order on the clusters naturally induced by  $<$ .

Since directed acyclic graphs (DAGs) correspond to partial orders, our method works equally well for DAGs. If the input is a DAG, then every clustering in the produced hierarchy is a DAG of clusters, and there exists a DAG homomorphism from the original DAG to the cluster DAG.

#### I.1.1 Motivating real-world use case

The motivation for our method comes from an industry database of machine parts that are arranged in part-of relations: parts are registered as sub-parts of other parts. For historical reasons, there have been incidents of copy-paste of machine designs, and the copies have been given entirely new identifiers with no links to the original design. In hindsight, there is a wish to identify these equivalent machine parts, but telling them apart is hard. Also, the metadata that is available has a tendency of displaying high similarity between a part and its sub-parts, leading to “vertical clustering” in the data.

Since the motivation is to identify equivalent machinery with the aim of replacing one piece of machinery with an equivalent part, and since a part and its sub-parts by no means can be interchanged, it is essential to maintain this parent-child relationship. Moreover, since a part and its sub-part are never equivalent, this is a strict order relation. The set of all machine parts thus makes up a strictly partially ordered set. By preserving these relations in the clustering process, we can eliminate the errors due to close resemblance between the part and the sub-part, resulting in improved over all quality of the clustering.

It is possible to imagine other use cases. We choose to mention two; citation network analysis and time series alignment:

Citation networks are partial orders, where the order is defined by the citations. If we perform order preserving clustering in the above sense on citation networks, the clusters will contain related research, and the clusters will be

ordered according to appearance relative other related research. This differs from clustering with regards to time: when clustering with time as a parameter, you have to choose, implicitly or explicitly, a time interval for each cluster. When the citation graph is used for ordering, the clusters will contain research that occurred in parallel, citing similar sources, and being cited by similar sources, regardless to whether they occurred in some particular time interval.

A time series is a totally ordered set of events, so that a family of time series is a partially ordered set. Assume that you want to do time series alignment, matching events from one time series with events from another, but for some reason the time stamps are corrupted and cannot be used for this purpose. Given a measure of (dis-)similarity between events, we can cluster the events to figure out which events are the more similar. Since an optimal order preserving clustering is one that both preserves all event orders and matches the most similar events across the time series, ideally the result is a series of clusters with each cluster containing the events that correspond to each other across the time series.

### I.1.2 Problem overview

Given a set  $X$  together with a notion of (dis-)similarity between the elements of  $X$ , a hierarchical agglomerative clustering can be obtained as follows [21, §3.2]:

1. Start by placing each element of  $X$  in a separate cluster.
2. Pick the two clusters that are most similar according to the (dis-)similarity measure, and combine them into one cluster by taking their union.
3. If all elements of  $X$  are in the same cluster, we are done. Otherwise, go to Step 2 and continue.

The result from this process is a dendrogram; a tree structure showing the sequence of the clustering process (Figure I.1).

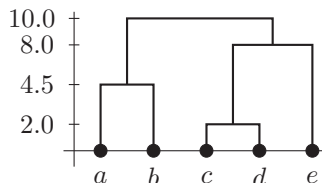


Figure I.1: A dendrogram over the set  $X = \{a, b, c, d, e\}$ . The elements of  $X$  are the leaf nodes of the dendrogram, and, starting at the bottom, the horizontal bars indicate which elements are joined at which step in the process. The numbers on the  $y$ -axis indicate at which dissimilarity level the different clusters were formed.

## I. Order preserving hierarchical agglomerative clustering

Now, given a partially ordered set  $X = \{a, b, c, d\}$  where  $a < b$  and  $c < d$ , we can use arrows to denote the order relation, thinking of  $X$  as a directed acyclic graph with two connected components. If we want to produce a hierarchical clustering of  $X$ , while at the same time maintaining the order relation, our options are depicted in the Hasse digram in Figure I.2.

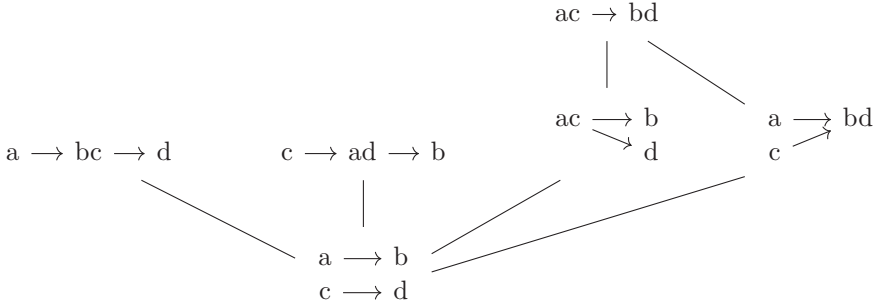


Figure I.2: Possible order preserving hierarchical clusterings over the set  $X = \{a, b, c, d\}$  with  $a < b$  and  $c < d$ . Adjacent elements indicate clusters.

Each path in this diagram, starting at the bottom and advancing upwards, represents a hierarchical clustering. But, since we are required to preserve the strict order relation, we cannot merge any more elements than what we see here. This means that we will never obtain dendrograms like the one in Figure I.1, that joins at the top when all elements are placed in a single cluster. Rather, the output of hierarchical agglomerative clustering would take the form of *partial dendrograms* like those of Figure I.3.

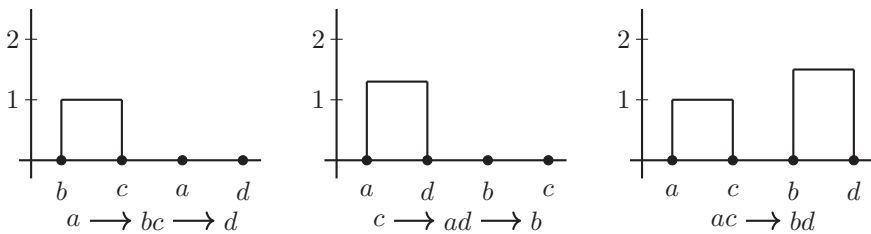


Figure I.3: Partial dendrograms over the set  $X = \{a, b, c, d\}$  with  $a < b$  and  $c < d$ . Each partial dendrogram corresponds to a path in Figure I.2 starting at the bottom and advancing upwards to the ordered set depicted below the dendrogram.

To complicate matters, if both  $(a, d)$  and  $(a, c)$  are pairs of minimal dissimilarity, then they are both candidates for the first merge. From Figure I.2 we can see that  $ad$  and  $ac$  are mutual exclusive merges, and that choosing one

over the other leads to very different solutions. We therefore need a method to decide which candidate merge, or which candidate partial dendrogram, is the better.

### 1.1.3 Outline of our method and contributions

As our first contribution, to solve the problem of picking one candidate merge among a set of tied connections, we present a permutation invariant method for hierarchical agglomerative clustering. The method uses the classical linkage functions of single-, average- and complete linkage, but is optimisation based, as opposed to the algorithmic definition of classical hierarchical clustering. Recalling that every hierarchical clustering corresponds to a unique ultrametric [23], the optimisation criterion is that of minimising the matrix norm of the difference between the original dissimilarity and the ultrametric corresponding to the hierarchical clustering, a method known as *ultrametric fitting* [11].

We have seen that order preserving hierarchical agglomerative clustering on strictly partially ordered sets leads to partial dendrograms. In order to evaluate the ultrametric fitting of a partial dendrogram, our next contribution is an embedding of partial dendrograms over a set into the family of ultrametrics over the same set.

Our main contribution, *order preserving hierarchical agglomerative clustering of strictly partially ordered sets*, is the combination of the two. We define an optimal order preserving hierarchical clustering to be the hierarchical clustering with the partial dendrogram that has the best ultrametric fit relative the original dissimilarity measure.

In want of an efficient algorithm, we present a method of approximation that can be computed in polynomial time. We demonstrate the approximation on synthetic data generated as random directed acyclic graphs and random dissimilarity measures, as well as on data from the parts database motivating this research. We evaluate the quality of the obtained clustering by computing the adjusted Rand index relative a planted partition [19]. We provide a novel method for comparing two induced order relations using a modified adjusted Rand index, which we believe is a first of its kind. We also provide simple method for computing the level of order preservation of a clustering of an ordered set by counting the number of induced loops.

Beyond our main contribution, we believe that the embedding of partial dendrograms into ultrametrics may be of interest to a larger audience. The embedding provides a means for treating partial dendrograms as complete dendrograms, offering access to the entire rack of tools that already exists in this domain. An obvious example candidate is that of hierarchical clustering with must-link and no-link constraints. The no-link constraints will necessarily lead to partial dendrograms that can be easily evaluated in our framework.

### I.1.3.1 Summary of contributions

Our main contribution is the theory for order preserving hierarchical agglomerative clustering for strict posets. Further contributions we wish to highlight are:

- A theory for embedding partial dendrograms over a set into the set of complete dendrograms over the same set.
- An optimisation based, permutation invariant hierarchical clustering methodology for non-ordered sets that is very similar to classical hierarchical clustering.
- A polynomial time approximation scheme for order preserving hierarchical agglomerative clustering
- A novel method for comparison of induced order relations over a set based on the adjusted Rand index.
- A measure of the level of order preservation of a clustering of an ordered set.

### I.1.4 Related work

Hierarchical agglomerative clustering is described in a plethora of books and articles, and we shall not try to give an account of that material. For an introduction to the subject, see [21, §3.2].

#### I.1.4.1 Clustering of ordered data

There are quite a few articles presenting clustering of ordered data, placing themselves in one of two categories.

The first is clustering of sets where the (dis)similarity measure is replaced by information about whether one pair of elements is more similar than another pair of elements, for example based on user preferences. This is sometimes referred to as *comparison based clustering*. See the recent article by Ghoshdastidar, Perrot and Luxburg [14] for an example and references. In this category, we also find the works of Janowitz [22], providing a wholly order theoretic description of hierarchical clustering, including the case where the dissimilarity measure is replaced by a partially ordered set.

The second variant is to partition a family of ordered sets so that similarly ordered sets are associated with each other. Examples include the paper by Kamishima and Fujiki [26], where they develop a variation of  $k$ -means, called  $k$ - $o$ 'means, for clustering preference data, each list of preferences being a totally ordered set. Other examples in this category include clustering of times series, identifying which times series are alike [31].

Our method differs from all of the above in that we cluster elements inside one ordered set through the use of a (dis)similarity measure, while maintaining the original orders of elements.



### 1.1.4.2 Clustering to detect order

Another variant is the detection of order relations in data through clustering: In [7], it is demonstrated how hierarchical agglomerative quasi-clustering can be used to deduce a partial order of “net flow” from an asymmetric network.

In this category, it is also worth mentioning dynamic time warping. This is a method for aligning time series, and can be considered as clustering across two time series that is indeed order preserving. See [31] for further references on this.

### 1.1.4.3 Acyclic graph partitioning problems

The problem of order preserving hierarchical agglomerative clustering can be said to belong to the family of *acyclic graph partitioning problems* [16]. If we consider the strict partial order to be a directed acyclic graph (DAG), the task is to partition the vertices into groups so that the groups together with the arrows still makes up a DAG.

Graph partitioning has received a substantial attention from researchers, especially within computer science, over the last 50 years. Two important fields of application of this theory are VLSI and parallel execution.

In VLSI, short for Very Large Scale Integration, the problem can be formulated as follows: Given a set of micro processors, the wires that connect them, and a set of circuit boards, how do you best place the processors on the circuit boards in order to optimise a given objective function? Typically, a part of the objective function is to minimise the wire length. But other features may also be part of the optimisation, such as the amount or volume of traffic between certain processors etc. [33]

For parallel processing, the input data is a set of tasks to be executed. The tasks are organised as a DAG, where predecessors must be executed before descendants. Given a finite number of processors, the problem is to group the tasks so that they can be run group-wise on a processor, or running groups in parallel on different processors, in order to execute all tasks as quickly as possible. Typically additional information available is memory requirements, expected execution times for the tasks, etc. [5]

It is not difficult to understand why both areas have received attention, being essential in the development of modern computers. The development of theory and methods has been both successful and abundant, and a large array of techniques are available, both academic and commercially.

Although both problems do indeed perform clustering of strict partial orders, their solutions are not directly transferable to exploratory data analysis. Mostly because they have very specific constraints and objectives originating from their respective problem domains.

The method we propose in this paper has as input a strict partial order (equivalently; a DAG) together with an arbitrary dissimilarity measure. We then use the classical linkage functions single-, average-, and complete linkage to suggest clusterings of the vertices from the input dataset, while preserving the original order relation.

Our method therefore places itself firmly in the family of acyclic graph partitioning methodologies, but with different motivation, objective and solution, compared to existing methods.

### I.1.4.4 Hierarchical clustering as an optimisation problem

Several publications aim at solving hierarchical clustering in terms of optimisation. However, due to the procedural nature of classical hierarchical clustering, combined with the linkage functions, pinning down an objective function may be an impossible task. Especially since classical hierarchical clustering is not even well defined for complete linkage in the presence of tied connections. This leads to a general abandonment of linkage functions in optimisation based hierarchical clustering.

Quite commonly, optimisation based hierarchical clustering is done in terms of *ultrametric fitting*. That is, it aims to find an ultrametric that is as close to the original dissimilarity measure as possible, perhaps adding some additional constraints [8, 15]. It is well known that solving single linkage hierarchical clustering is equivalent to finding the so called *maximal sub-dominant ultrametric*. That is; the ultrametric that is pointwise maximal among all ultrametries not exceeding the original dissimilarity [35]. But for the other linkage functions, there is no equivalent result.

Optimisation based hierarchical clustering therefore generally present alternative definitions of hierarchical clustering. Quite often based on objective functions that originate from some particular domain. Exceptions from this are, for example, Ward's method [39], where the topology of the clusters are the focus of the objective, and also the recent addition by Dasgupta [9], where the optimisation aims towards topological properties of the generated dendrogram.

Although our method is, eventually, based on ultrametric fitting, we optimise over a very particular set of dendrograms. Namely the dendrograms that can be generated through classical hierarchical clustering with linkage functions. It is therefore reasonable to claim that our method places itself between classical hierarchical clustering and optimised models.

### I.1.4.5 Clustering with constraints

A significant amount of research has been devoted to the topic of clustering with constraints in the form of pairwise *must-link* or *no-link* constraints, often in addition to other constraints, such as minimal- and maximal distance constraints, and so on. Some work as also been done on hierarchical agglomerative clustering with constraints, starting with the works of Davidson and Ravi [10]. For a thorough treatment of constrained clustering, see [2].

Order preserving clustering (as well as acyclic partitioning) can be seen as a particular version of constrained clustering, where the constraint is a *directed, transitive cannot-link* constraint. A type of constraint that is not found in the constrained clustering literature.

### I.1.4.6 Clustering in information networks

A large amount of research has been conducted on the problem of clustering nodes in networks, and a more recent field of research is that of clustering data organised in *heterogeneous information networks*, or HINs for short [34]. A HIN is an undirected graph where both vertices and edges may have different, or even multiple, types. RDF graphs [29] is but one example of HINs. In a sense, we can say that the availability of multiple types allow HINs to model the real world more closely, but with the penalty of increased complexity. It is fair to consider HIN clustering a generalisation of classical network clustering, where, in the classical setting, all vertices and edges are of one common type.

However, the general case in clustering both classical networks and HINs is that although the network structure serves to influence the clustering, the structure is usually lost in the clustering. The most classical example is where connectedness between vertices contribute to vertex similarity, and then the most connected vertices (clique-like subgraphs) are clustered together. Although this can be seen as a type of relation preserving clustering, in order preserving clustering, the opposite is taking place: the more connected two vertices are, the more reason *not* to place them in the same cluster. Indeed, as we show in Section I.4, for the theory we present in this paper, two elements can only be clustered together if there are *no* paths connecting them.

An example of HIN clustering that *is* structure preserving is [30]. A HIN comes with a schema, or a *schematic graph*, describing which types are related to which other types. For Li et al. [30], the goal is to cluster each set of same-type nodes according to a discovered similarity measure. The result is thus a schematic graph where each node is a clustering of vertices of the same type. This differs from the problem we study in that we do not know which elements are of the same type; to discover this is the goal of the clustering. Hence, the problems are similar but different; we could rephrase our problem as that of *deriving* a directed schematic graph from unlabeled vertices, where each vertex in the schematic graph is a set of equivalent machine parts, and the directed edges are the part-of relations.

### I.1.5 Organisation of the remainder of this paper

Section I.2 provides necessary background material.

In Section I.3, we develop *optimised hierarchical agglomerative clustering* for non-ordered sets; our permutation invariant clustering model that is tailored especially to fit into our framework for agglomerative clustering of ordered sets.

In Section I.4, we tackle the problem of order preservation during clustering: We define what we mean by order preservation, and classify exactly the clusterings that are order preserving. We also provide concise necessary and sufficient conditions for an hierarchical agglomerative clustering algorithm to be order preserving.

Section I.5 defines partial dendrograms and develops the embedding of partial dendrograms over an ordered set into the family of ultrametrics over the same

set.

Our main result, *order preserving hierarchical agglomerative clustering for strict partial orders*, is presented Section I.6.

Section I.7 provides a polynomial time approximation scheme for our method, and Section I.8 demonstrates the efficacy of the approximation on synthetic data.

Section I.9 presents the results from applying our approximation method to a subset of the data in the parts database, comparing with existing methods, and finally, Section I.10 closes the article with some concluding remarks, and a list of future work topics.

## I.2 Background

In this section we recall basic background material. We start by recollecting the required order-theoretical tools together with equivalence relations, before recalling classical hierarchical clustering.

### I.2.1 Relations

**Definition I.2.1.** A **relation**  $R$  on a set  $X$  is a subset  $R \subseteq X \times X$ , and we say that  $x$  and  $y$  are **related** if  $(x, y) \in R$ . The short hand notation  $aRb$  is equivalent to writing  $(a, b) \in R$ .

#### I.2.1.1 Strict and non-strict partial orders

A **strict partial order** on a set  $X$  is a relation  $S$  on  $X$  that is irreflexive and transitive. Recall that, an irreflexive and transitive relation is also anti-symmetric. A **strictly partially ordered set**, or a **strict poset**, is a pair  $(X, S)$ , where  $X$  is a set and  $S$  is a strict partial order on  $X$ . We commonly denote a strict partial order by the symbol  $<$ .

On the other hand a **partial order** on  $X$  is a relation  $P$  on  $X$  that is reflexive, asymmetric and transitive, and the pair  $(X, P)$  is called a **partially ordered set**, or a **poset**. The usual notation for a partial order is  $\leq$ .

We shall just refer to strict and non-strict partial orders as *orders*, unless there is any need for disambiguation: If  $R$  is an order on  $X$ , we say that  $a, b \in X$  are **comparable** if either  $(a, b) \in R$  or  $(b, a) \in R$ . And, if every pair of elements in  $X$  are comparable, we call  $X$  **totally ordered**. A totally ordered subset of an ordered set is called a **chain**, and a subset where no two elements are comparable is called an **antichain**. We denote **non-comparability** by  $a \perp b$ . That is, for any elements  $a, b$  in an antichain, we have  $a \perp b$ .

A **cycle** in a relation  $E$  is a sequence in  $E$  on the form  $(a, b_1), (b_1, b_2), \dots, (b_n, a)$ . The **transitive closure** of  $E$  is the minimal set  $\overline{E}$  for which the following holds: If there is a sequence of pairs  $(a_1, a_2), (a_2, a_3), \dots, (a_{n-1}, a_n)$  in  $E$ , then  $(a_1, a_n) \in \overline{E}$ .

Let  $(X, E)$  be an ordered set. An element  $x_0 \in X$  is a **minimal element** if there is no element  $y \in X - \{x_0\}$  for which  $(y, x_0) \in E$ . Dually,  $y_0$  is a **maximal element** if there is no  $x \in X - \{y_0\}$  for which  $(y_0, x) \in E$ . If  $(X, E)$

has a unique minimal element, then this is called the **bottom element** or the **least element**, and a unique maximal element is called the **top element** or the **greatest element**.

Finally, a map  $f : (X, <_X) \rightarrow (Y, <_Y)$  is **order preserving** if  $a <_X b \Rightarrow f(a) <_Y f(b)$ , and if  $f$  is a set isomorphism (that is, a bijection) for which  $f^{-1}$  is also order preserving, we say that  $f$  is an **order isomorphism**, and that the sets  $(X, <_X)$  and  $(Y, <_Y)$  are **order isomorphic**, writing  $(X, <_X) \approx (Y, <_Y)$ .

### 1.2.1.2 Partitions and equivalence relations

A **partition** of  $X$  is a collection of disjoint subsets of  $X$ , the union of which is  $X$ . The family of all partitions of  $X$ , denoted  $\mathfrak{P}(X)$ , has a natural partial order defined by partition-refinement: If  $\mathcal{A} = \{A_i\}_i$  and  $\mathcal{B} = \{B_j\}_j$  are partitions of  $X$ , we say that  $\mathcal{A}$  is a **refinement** of  $\mathcal{B}$ , writing  $\mathcal{A} \Subset \mathcal{B}$ , if, for every  $A_i \in \mathcal{A}$  there exists a  $B_j \in \mathcal{B}$  such that  $A_i \subseteq B_j$ . The sets of a partition are referred to as **blocks**.

An **equivalence relation** is a relation  $\mathcal{R}$  on  $X$  that is reflexive, symmetric and transitive. Let the family of all equivalence relations over a set  $X$  be denoted by  $\mathfrak{R}(X)$ . If  $\mathcal{R} \in \mathfrak{R}(X)$  and  $(x, y) \in \mathcal{R}$ , we say that  $x$  and  $y$  are **equivalent**, writing  $x \sim y$ . The maximal set of elements equivalent to  $x \in X$  is called the **equivalence class of  $x$** , and is denoted  $[x]$ .  $\mathfrak{R}(X)$  is also partially ordered, but by subset inclusion: that is, for  $\mathcal{R}, \mathcal{S} \in \mathfrak{R}(X)$ , we say that  $\mathcal{R}$  is less than or equal to  $\mathcal{S}$  if and only if  $\mathcal{R} \subseteq \mathcal{S}$ .

The **quotient of  $X$  modulo  $\mathcal{R}$** , denoted  $X/\mathcal{R}$ , is the set of equivalence classes of  $X$  under  $\mathcal{R}$ . Notice that  $[x]$  is an element of  $X/\mathcal{R}$ , but a subset of  $X$ . Since the equivalence classes are subsets of  $X$  that together cover  $X$ ,  $X/\mathcal{R}$  is a partition of  $X$  with equivalence classes being the blocks of the partition. The family of partitions of  $X$  is in a one-to-one correspondence with the equivalence relations of  $X$ , and the correspondence is order preserving; if  $\mathcal{A} = X/\mathcal{A}$  and  $\mathcal{B} = X/\mathcal{B}$ , we have

$$\mathcal{A} \Subset \mathcal{B} \Leftrightarrow \mathcal{A} \subseteq \mathcal{B}.$$

Both  $\mathfrak{P}(X)$  and  $\mathfrak{R}(X)$  have top- and bottom elements: The least element of  $\mathfrak{P}(X)$  is the singleton partition  $S(X)$ , where each element is in a block by itself:  $S(X) = \{\{x\} \mid x \in X\}$ . The singleton partition corresponds to the diagonal equivalence relation, given by  $\Delta(X) = \{(x, x) \mid x \in X\}$ , which is the least element of  $\mathfrak{R}(X)$ . The greatest element of  $\mathfrak{P}(X)$  is the trivial partition  $\{X\}$ , corresponding to the equivalence relation  $X \times X$ , where all element are equivalent. That is

$$S(X) = X/\Delta(X) \quad \text{and} \quad \{X\} = X/(X \times X).$$

If  $\mathcal{A}$  and  $\mathcal{B}$  are partitions of  $X$  with  $\mathcal{A}$  being a refinement of  $\mathcal{B}$ , we say that  $\mathcal{A}$  is **finer** than  $\mathcal{B}$ , and that  $\mathcal{B}$  is **coarser** than  $\mathcal{A}$ . We use the exact same terminology for the corresponding equivalence relations.

For a subset  $A \subseteq X$ , let the notation  $X/A$  denote the partition of  $X$  where all of  $A$  is one equivalence class, and the rest of  $X$  remains as singletons. Formally,

this corresponds to the equivalence relation  $\mathcal{R}_A = \Delta(X) \cup (A \times A)$ . And finally, the **quotient map** corresponding to an equivalence relation  $\mathcal{R} \in \mathfrak{R}(X)$  is the unique map  $q_{\mathcal{R}} : X \rightarrow X/\mathcal{R}$  defined as  $q_{\mathcal{R}}(x) = [x]$ . That is,  $q_{\mathcal{R}}$  sends each element to its equivalence class.

### I.2.2 Classical hierarchical clustering

In this section, we recall classical hierarchical clustering in terms of Jardine and Sibson [23]. Our theory builds directly on the theory for classical hierarchical clustering, so we need to provide a fair bit of detail, especially since there is a general lack of standardised notation for hierarchical clustering theory.

We start by recalling the formal definition of a dendrogram, before recalling dissimilarity measures and ultrametrics. Thereafter, we recall linkage functions, before finally tying all the concepts together to define classical hierarchical agglomerative clustering.

**Definition I.2.2.** A **clustering** of a set  $X$  is a partition of  $X$ , and a **hierarchical clustering** is a chain in  $\mathfrak{P}(X)$  containing both the bottom and top elements. A **cluster** in a clustering is a block in the partition.

Alternatively, a clustering of  $X$  is an equivalence relation  $\mathcal{R} \in \mathfrak{R}(X)$ , and a hierarchical clustering is a chain in  $\mathfrak{R}(X)$  containing both the bottom- and top elements of  $\mathfrak{R}(X)$ . A cluster is, then, an equivalence class in  $X/\mathcal{R}$ . We will refer to clusters as equivalence classes, clusters or blocks depending on the context, all terms being frequently used in clustering literature.

For the remainder of the paper, let  $\mathbb{R}_+$  denote the non-negative reals. We generally assume that  $\mathbb{R}_+$  is equipped with the usual total order  $\leq$ .

**Definition I.2.3.** Given a set  $X$ , let  $\mathfrak{P}(X)$  be partially ordered by partition refinement. A **dendrogram** is an order preserving map  $\theta : \mathbb{R}_+ \rightarrow \mathfrak{P}(X)$  for which the following properties holds:

- D1.  $\forall t \in \mathbb{R}_+ \exists \varepsilon > 0$  s.t.  $\theta(t) = \theta(t + \varepsilon)$ ;
- D2.  $\theta(0) = S(X)$ , the least element of  $\mathfrak{P}(X)$ ;
- D3.  $\exists t_0 > 0$  s.t.  $\theta(t_0) = \{X\}$ , the greatest element of  $\mathfrak{P}(X)$ .

We will use the term dendrogram to denote both the graphical and the functional representation. If  $im(\theta) = \{\mathcal{B}_i\}_{i=0}^n$ , we assume that the enumeration is compatible with the order relation on  $\mathfrak{P}(X)$ ; in other words, that  $\{\mathcal{B}_i\}_{i=0}^n$  is a chain in  $\mathfrak{P}(X)$ . We denote **the family of all dendrograms over  $X$**  by  $\mathcal{D}(X)$ .

A **dissimilarity measure** on a set  $X$  is a function  $d : X \times X \rightarrow \mathbb{R}_+$ , satisfying

- d1.  $\forall x \in X : d(x, x) = 0$ ,
- d2.  $\forall x, y \in X : d(x, y) = d(y, x)$ .

If  $d$  additionally satisfies

$$d3. \forall x, y, z \in X : d(x, z) \leq \max\{d(x, y), d(y, z)\},$$

we call  $d$  an **ultrametric** [35]. The pair  $(X, d)$  is correspondingly called a **dissimilarity space** or an **ultrametric space**. The family of all dissimilarity measures over  $X$  is denoted by  $\mathcal{M}(X)$ , and the family of all ultrametries by  $\mathcal{U}(X)$ .

**Example I.2.4** (Ultrametric). Property  $d3$  is referred to as the **ultrametric inequality**, and is a strengthening of the usual triangle inequality. In an ultrametric space  $(X, \mathbf{u})$ , *every triple of points is arranged in an isosceles triangle*: Let  $a, b, c \in X$ , and let the pair  $a, b$  be of minimal distance such that  $\mathbf{u}(a, b) \leq \min\{\mathbf{u}(a, c), \mathbf{u}(b, c)\}$ . The ultrametric inequality gives us

$$\left. \begin{array}{l} \mathbf{u}(a, c) \leq \max\{\mathbf{u}(a, b), \mathbf{u}(b, c)\} = \mathbf{u}(b, c) \\ \mathbf{u}(b, c) \leq \max\{\mathbf{u}(b, a), \mathbf{u}(a, c)\} = \mathbf{u}(a, c) \end{array} \right\} \Leftrightarrow \mathbf{u}(a, c) = \mathbf{u}(b, c).$$

Ultrametries show up in many different contexts, such as  $p$ -Adic number theory [18], infinite trees [20], numerical taxonomy [37] and also within physics [35], just to cite a few. For hierarchical clustering, ultrametries are relevant because the dendrograms over a set are in a bijective relation to the ultrametries over the same set [6].

We shall also need the following terms, which apply to any dissimilarity space: The **diameter** of  $(X, d)$  is given by the maximal inter-point distance:

$$\text{diam}(X, d) = \max\{d(x, y) \mid x, y \in X\}.$$

And the **separation** of  $(X, d)$  is the minimal inter point distance:

$$\text{sep}(X, d) = \min\{d(x, y) \mid x, y \in X \wedge x \neq y\}.$$

It is a well known fact that there exists an injective map from dendrograms to ultrametries [23]:

$$\Psi_X : \mathcal{D}(X) \longrightarrow \mathcal{U}(X).$$

In [6] the map  $\Psi_X$  is shown to be a bijection. If  $\theta \in \mathcal{D}(X)$ , the map is defined as

$$\Psi_X(\theta)(x, y) = \min\{t \in \mathbb{R}_+ \mid \exists B \in \theta(t) : x, y \in B\}. \quad (\text{I.1})$$

That is, the ultrametric distance is the least real number  $t$  for which  $\theta$  maps to a partition where  $x$  and  $y$  are in the same block. The minimisation is well defined due to Axiom D1. The ultrametric can be read from the diagrammatic representation of the dendrogram as the minimum height you have to ascend to in order to traverse from one element to the other following the paths in the tree.

Before we provide a formal definition of classical hierarchical clustering, we need to recall linkage functions. Our definition follows the lines of Carlsson and Mémoli [6]:

**Definition I.2.5.** Let  $\mathcal{P}(X)$  denote the power set of  $X$ . A **linkage function** on  $X$  is a map

$$\mathcal{L} : \mathcal{P}(X) \times \mathcal{P}(X) \times \mathcal{M}(X) \longrightarrow \mathbb{R}_+,$$

so that for each partition  $Q \in \mathfrak{P}(X)$  and dissimilarity measure  $d \in \mathcal{M}(X)$ , the restriction  $\mathcal{L}|_{Q \times Q \times \{d\}}$  is a dissimilarity measure on  $Q$ .

The classical linkage functions are defined as

$$\begin{aligned} \text{Single linkage} & : \mathcal{SL}(p, q, d) = \min_{x \in p} \min_{y \in q} d(x, y), \\ \text{Complete linkage} & : \mathcal{CL}(p, q, d) = \max_{x \in p} \max_{y \in q} d(x, y), \\ \text{Average linkage} & : \mathcal{AL}(p, q, d) = \frac{\sum_{x \in p} \sum_{y \in q} d(x, y)}{|p| \cdot |q|}. \end{aligned}$$

**Definition I.2.6** (Classical  $\mathcal{HC}$ ). Given a dissimilarity space  $(X, d)$  and a linkage function  $\mathcal{L}$ , if we follow the procedure outlined in Section I.1.2, using  $\mathcal{L}$  as the “notion of dissimilarity”, the result is a chain of partitions  $\{Q_i\}_{i=1}^{|X|-1}$  together with the dissimilarities  $\{\rho_i\}_{i=1}^{|X|-1}$  at which the partitions were formed. The sequence of pairs  $\mathcal{Q} = \{(Q_i, \rho_i)\}_{i=1}^{|X|-1}$  corresponds uniquely to a dendrogram  $\theta_{\mathcal{Q}}$  as follows:

$$\theta_{\mathcal{Q}}(x) = Q_{\max\{i \in \mathbb{N} \mid \rho_i \leq x\}}. \quad (\text{I.2})$$

We define a **classical hierarchical clustering of  $(X, d)$  using  $\mathcal{L}$**  to be a dendrogram

$$\mathcal{HC}^{\mathcal{L}}(X, d) = \theta_{\mathcal{Q}}$$

obtained through this procedure.

*Remark I.2.7.* Notice that (I.2) maps  $\{(Q_i, \rho_i)\}_{i=1}^{|X|-1}$  to a dendrogram if and only if

$$\text{sep}(Q_i, \mathcal{L}) \leq \text{sep}(Q_{i+1}, \mathcal{L}) \quad \text{for } 0 \leq i < |X| - 1. \quad (\text{I.3})$$

Otherwise, the  $\rho_i$  will not make up a monotone sequence, and the resulting function  $\theta_{\mathcal{Q}}$  will not be an order preserving map. Although all of  $\mathcal{SL}$ ,  $\mathcal{AL}$  and  $\mathcal{CL}$  satisfy (I.3), it is fully possible to define linkage functions that do not.

Finally, two distinct pairs of elements  $(p_1, q_1), (p_2, q_2) \in Q \times Q$  for which

$$\mathcal{L}(p_1, q_1, d) = \mathcal{L}(p_2, q_2, d) = \text{sep}(Q, \mathcal{L}),$$

are referred to as **tied**, since they are both eligible candidates for the next merge.

### I.3 Optimised hierarchical clustering

In this section we devise a permutation invariant version of hierarchical clustering based on the classical definition. The key to permutation invariance is in dealing with tied connections. If we consider the procedure for hierarchical clustering outlined in Section I.1.2, we can resolve tied connections by picking a random



minimal dissimilarity pair. The way the procedure is specified, this turns  $\mathcal{HC}^{\mathcal{L}}$  into a *non-deterministic* algorithm; it may produce different dendrograms for the same input in the presence of ties, depending on which tied pair is selected. But more importantly, it is capable of producing *any* dendrogram that can be produced by *any* tie resolution order:

**Definition I.3.1.** Given a dissimilarity space  $(X, d)$  and a linkage function  $\mathcal{L}$ , let  $\mathcal{D}^{\mathcal{L}}(X, d)$  be **the set of all possible outputs from  $\mathcal{HC}^{\mathcal{L}}(X, d)$ .**

A dissimilarity measure  $d$  over a finite set  $X$  can be described as an  $|X| \times |X|$  real matrix  $[d_{i,j}]$ . Hence, given an ultrametric  $\mathbf{u} \in \mathcal{U}(X)$  we can compute the pointwise difference

$$\|\mathbf{u} - d\|_p = \sqrt[p]{\sum_{x,y \in X} |\mathbf{u}(x,y) - d(x,y)|^p}. \quad (\text{I.4})$$

We suggest the following definition, recalling the definition of  $\Psi_X$  (I.1):

**Definition I.3.2.** Given a dissimilarity space  $(X, d)$  and a linkage function  $\mathcal{L}$ , **the optimised hierarchical agglomerative clustering over  $(X, d)$  using  $\mathcal{L}$**  is given by

$$\mathcal{HC}_{opt}^{\mathcal{L}}(X, d) = \arg \min_{\theta \in \mathcal{D}^{\mathcal{L}}(X, d)} \|\Psi_X(\theta) - d\|_p. \quad (\text{I.5})$$

That is; among all dendrograms that can be generated by  $\mathcal{HC}^{\mathcal{L}}(X, d)$ , optimised hierarchical agglomerative clustering picks the dendrogram that is closest to the original dissimilarity measure. In the tradition of ultrametric fitting, this is the right choice of candidate.

As  $\mathcal{D}^{\mathcal{L}}(X, d)$  contains all dendrograms generated over all possible permutations of enumerations of  $X$ , the below theorem follows directly from Definition I.3.2:

**Theorem I.3.3.**  $\mathcal{HC}_{opt}^{\mathcal{L}}$  is permutation invariant. That is, the order of enumeration of the elements of the set  $X$  does not affect the output from  $\mathcal{HC}_{opt}^{\mathcal{L}}(X, d)$ .

And since  $\mathcal{HC}^{S\mathcal{L}}$  is permutation invariant, we have  $|\mathcal{D}^{S\mathcal{L}}(X, d)| = 1$ , yielding

**Theorem I.3.4.**  $\mathcal{HC}_{opt}^{S\mathcal{L}}(X, d) = \mathcal{HC}^{S\mathcal{L}}(X, d)$ .

Since  $\mathcal{HC}^{A\mathcal{L}}$  and  $\mathcal{HC}^{C\mathcal{L}}$  are not permutation invariant, there is no corresponding result in these cases. For complete linkage, however, we have the following theorem. First, notice that due to the definition of complete linkage (Definition I.2.5), if  $\theta$  is a solution to  $\mathcal{HC}_{opt}^{C\mathcal{L}}(X, d)$  and  $\mathbf{u} = \Psi_X(\theta)$  is the corresponding ultrametric, then

$$\mathbf{u}(x, y) \geq d(x, y) \quad \forall x, y \in X.$$

Hence, in the case of complete linkage we can reformulate (I.5) as follows:

$$\mathcal{HC}_{opt}^{C\mathcal{L}}(X, d) = \arg \min_{\theta \in \mathcal{D}^{C\mathcal{L}}(X, d)} \|\Psi_X(\theta)\|_p. \quad (\text{I.6})$$

## I. Order preserving hierarchical agglomerative clustering

---

To see why this is the case, notice that if  $u, u' \in \mathcal{M}(X)$  and both  $d \leq u$  and  $d \leq u'$  pointwise, then we can produce two non-negative functions  $\delta, \delta'$  on  $X \times X$  so that  $u = d + \delta$  and  $u' = d + \delta'$ . In particular, we have  $u - d = \delta$ , from which we deduce

$$\|u - d\|_p \leq \|u' - d\| \Leftrightarrow \|\delta\|_p \leq \|\delta'\|_p \Leftrightarrow \|d + \delta\|_p \leq \|d + \delta'\|_p \Leftrightarrow \|u\|_p \leq \|u'\|_p.$$

**Theorem I.3.5.** *Solving  $\mathcal{HC}_{opt}^{\mathcal{CL}}(X, d)$  is NP-hard.*

*Proof.* Let  $G = (V, E)$  be an undirected graph with vertices  $V$  and edges  $E \subseteq V \times V$ . Recall the *clique* problem: Given a positive integer  $K < |V|$ , is there a clique in  $G$  of size at least  $K$ ? Equivalently: is there a set  $V' \subseteq V$  with  $|V'| \geq K$  for which  $V' \times V' \subseteq E$ ? This is a known NP-hard problem [27].

To reduce *clique* to  $\mathcal{HC}_{opt}^{\mathcal{CL}}$ , define a dissimilarity measure on  $V$  as follows:

$$d(v, v') = \begin{cases} 1 & \text{if } (v, v') \in E, \\ 2 & \text{otherwise.} \end{cases} \quad (\text{I.7})$$

Then  $(V, d)$  is a dissimilarity space. Let  $\theta$  be a solution of  $\mathcal{HC}_{opt}^{\mathcal{CL}}(V, d)$ , and set  $\mathfrak{d} = \Psi_V(\theta)$ .

An intrinsic property of  $\mathcal{CL}$  is that if two blocks  $p, q \in Q_i$  are merged, then

$$\forall v, v' \in p \cup q : d(v, v') \leq \mathcal{CL}(p, q, d).$$

And since we have  $d(v, v') = 1 \Leftrightarrow (v, v') \in E$ , it means that for a subset  $V' \subseteq V$ , we have that

$$\forall v, v' \in V' : \mathfrak{d}(v, v') = 1 \Leftrightarrow V' \text{ is a clique in } G. \quad (\text{I.8})$$

It follows that a largest possible cluster at proximity level 1 is a maximal clique in  $G$ .

We claim that minimising the norm is equivalent to producing a maximal cluster at proximity level 1: Let  $\mathfrak{d}$  be the  $|V| \times |V|$  distance matrix  $[\mathfrak{d}_{i,j}]$ . Due to the definition of  $\mathcal{CL}$ , we have  $\mathfrak{d}(v, v') \in \{0, 1, 2\}$ . If  $\theta(1) = \{V_i\}_{i=1}^s$ , then these are exactly the blocks that are subsets of cliques, so each  $V_i$  contributes with  $|V_i|(|V_i| - 1)$  ones in  $[\mathfrak{d}_{i,j}]$ .

Having more ones reduces the norm of  $\mathfrak{d}$ . Let  $V_j$  be of maximal cardinality in  $\{V_i\}_{i=1}^s$ . Assume first that  $V_j$  has at least two elements more than the next to largest block, and let  $|V_j| = P$ .

Removing one element from  $V_j$  reduces the number of ones in the dissimilarity matrix by  $P(P - 1) - (P - 1)(P - 2) = 2(P - 1)$ . Let the next to largest block have  $Q$  elements. Transferring the element to this block then increases the number of ones by  $(Q + 1)Q - Q(Q - 1) = 2Q$ . Since  $Q < P - 1$ , this means that the total number of ones is reduced by moving an element from the largest block to any of the smaller blocks. Hence, achieving the largest possible number of ones implies maximising the size of the largest block.

If now,  $V_j$  only has one element more than the next to largest block, moving an element as above corresponds to keeping the number of ones. Since each  $V_i$  for  $1 \leq i \leq s$  is a subset of a clique in  $G$ , the maximal number of ones is achieved by producing a block  $V_j$  that contains exactly a maximal clique of  $G$ .

Therefore, if  $\mathcal{I}_{\{1\}}(x)$  is the indicator function for the set  $\{1\}$ , the size of a maximal clique in  $G$  can be computed as

$$\max_{1 \leq i \leq |V|} \left\{ \sum_{j=1}^{|V|} \mathcal{I}_{\{1\}}(\mathfrak{d}_{i,j}) \right\},$$

counting the maximal number of row-wise ones in  $[\mathfrak{d}_{i,j}]$  in  $O(N^2)$  time. We therefore conclude that  $\mathcal{HC}_{opt}^{C\mathcal{L}}$  is NP-hard. ■

*The computational hardness of  $\mathcal{HC}_{opt}^{C\mathcal{L}}$  is directly connected to the presence of tied connections: every encounter of  $n$  tied connections leads to  $n!$  new candidate solutions.*

Since neither  $\mathcal{HC}_{opt}^{A\mathcal{L}}$  is permutation invariant, the authors strongly believe that this is also NP-hard, although that remains to be proven.

We cannot in general expect the mapping  $\theta \mapsto \|\Psi_X(\theta) - d\|_p$  to be injective, meaning that the answer to (I.5) may not be unique. Recall that  $\mathcal{P}(X)$  denotes the power set of  $X$ . We shall consider  $\mathcal{HC}_{opt}^{\mathcal{L}}(X, -)$  to be the function

$$\mathcal{HC}_{opt}^{\mathcal{L}}(X, -) : \mathcal{M}(X) \longrightarrow \mathcal{P}(\mathcal{D}(X)),$$

mapping a dissimilarity measure over  $X$  to a set of dendrograms over  $X$ .

### I.3.1 Other permutation invariant solutions

Carlsson and Mémoli [6] offer an alternative approach to permutation invariant hierarchical agglomerative clustering. In their solution, when they face a set of tied connections, they merge all tied the pairs in one operation, resulting in permutation invariance.

In the case of order preserving clustering, a family of tied connections can contain several mutually exclusive merges due to the order relation. Using the method of Carlsson and Mémoli leads to a problem of figuring which blocks of tied connections to merge together, and in which combinations and order. This leads to a combinatorial explosion of alternatives. The method we have suggested is utterly simple, but it is designed to circumvent this very problem.

## I.4 Order preserving clustering

In this section, we determine what it means for an equivalence relation to be order preserving with regards to a strict partial order, and establish precise conditions that are necessary and sufficient for a hierarchical agglomerative clustering algorithm to be order preserving.

### I.4.1 Order preserving equivalence relations

Recalling the definition of a clustering (Definition I.2.2), let  $(X, <)$  be a strict poset. If  $\mathcal{R}$  is an equivalence relation on  $X$  with quotient map  $q : X \rightarrow X/\mathcal{R}$ , we have already established, in Section I.1.1, that we require

$$\forall x, y \in X : x < y \Rightarrow q(x) <' q(y).$$

*That is, we are looking for a particular class of equivalence relations; namely those for which the quotient map is order preserving.*

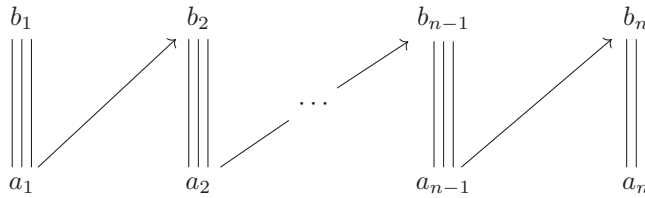
Given a strict poset  $(X, E)$ , there is a particular induced relation on the quotient set  $X/\mathcal{R}$  for any equivalence relation  $\mathcal{R} \in \mathfrak{R}(X)$  [3, §3.1]:

**Definition I.4.1.** Given a strict poset  $(X, E)$  and an equivalence relation  $\mathcal{R} \in \mathfrak{R}(X)$ , first define the relation  $S_0$  on  $X$  by

$$([a], [b]) \in S_0 \Leftrightarrow \exists x, y \in X : a \sim x \wedge b \sim y \wedge (x, y) \in E. \quad (\text{I.9})$$

The transitive closure of  $S_0$  is called **the relation on  $X/\mathcal{R}$  induced by  $E$** . We denote this relation by  $S$ .

**Example I.4.2.** An instructive illustration of what the relation  $S_0$  looks like for a strict poset  $(X, <)$  under the equivalence relation  $\mathcal{R}$  is that of an  $\mathcal{R}$ -fence [3], or just fence, for short:



Triple lines represent equivalences under  $\mathcal{R}$ , and the arrows represent the order on  $(X, <)$ . The fence illustrates visually how one can traverse from  $a_1$  to  $b_n$  along arrows and through equivalence classes in  $X/\mathcal{R}$ , and in that case we say that the fence **links**  $b_1$  to  $a_n$ . **The induced relation  $S$  has the property that  $(a, b) \in S$  if there exists an  $\mathcal{R}$ -fence in  $X$  linking  $a$  to  $b$ .**

Recall that a cycle in a relation  $R$  is a sequence of pairs starting and ending with the same element:  $(a, b_1), (b_1, b_2), \dots, (b_n, a)$ . The below theorem is an adaptation of [3, Thm.3.1] to strict partial orders.

**Theorem I.4.3.** *Let  $(X, E)$  be a strict poset,  $\mathcal{R} \in \mathfrak{R}(X)$ , and let  $S$  be the relation on  $X/\mathcal{R}$  induced by  $E$ . Then the following statements are equivalent:*

1.  $S$  is a strict partial order on  $X/\mathcal{R}$ ;
2. There are no cycles in  $S_0$ ;
3.  $q_{\mathcal{R}} : (X, E) \rightarrow (X/\mathcal{R}, S)$  is order preserving.

*Proof.* From the definition of strict posets, they contain no cycles, so  $1 \Rightarrow 2$ . Since a non-cyclic set is irreflexive, and since  $S$  is transitive by construction,  $2 \Rightarrow 1$ .

Let  $q_{\mathcal{R}}$  be order preserving. Notice that if  $S_0$  is the set defined in (I.9), we have  $S_0 = q_{\mathcal{R}} \times q_{\mathcal{R}}(E)$ . In particular, for all  $x, y \in X$  for which  $(x, y) \in E$ , we have  $([x], [y]) \in S_0$ . Assume that  $S$  is not a strict order. Then there is a cycle in  $S_0$ ; that is there are  $x, y \in X$  for which  $(x, y) \in E$ , but  $([y], [x]) \in S_0$  also. This yields

$$\exists a', b' \in X : a' \sim x \wedge b' \sim y \wedge (b', a') \in E.$$

But, since  $([x], [y]) \in S_0$ , we also have

$$\exists a, b \in X : a \sim x \wedge b \sim y \wedge (a, b) \in E.$$

This yields  $a \sim a'$  and  $b \sim b'$ , so we have

$$(q_{\mathcal{R}}(a), q_{\mathcal{R}}(b)) \in S_0 \wedge q_{\mathcal{R}}(b) = q_{\mathcal{R}}(b') \wedge (q_{\mathcal{R}}(b'), q_{\mathcal{R}}(a')) \in S_0.$$

But, since we have both  $q_{\mathcal{R}}(a) = q_{\mathcal{R}}(a')$  and  $(a, b) \in E$ , this contradicts the fact that  $q_{\mathcal{R}}$  is order preserving, so our assumption that both  $([x], [y])$  and  $([y], [x])$  are elements of  $S_0$  must be wrong. Hence, if  $q_{\mathcal{R}}$  is order preserving, there are no cycles in  $S_0$ , and  $S$  is a strict partial order on  $X/\mathcal{R}$ . This shows that  $3 \Rightarrow 1$ .

Finally, let  $S$  be a strict partial order, and assume that  $q_{\mathcal{R}}$  is not order preserving. Then, there exists  $x, y \in X$  where  $(x, y) \in E$  and for which at least one of  $([x], [y]) \notin S$  or  $([y], [x]) \in S$  holds. Now,  $([x], [y]) \in S$  by Definition I.4.1. Therefore,  $([y], [x]) \in S$  implies that  $S$  has a cycle, contradicting the fact that  $S$  is a strict partial order.  $\blacksquare$

**Definition I.4.4.** Let  $(X, E)$  be a strict poset. An equivalence relation  $\mathcal{R} \in \mathfrak{R}(X)$  is **regular** if there exists an order on  $X/\mathcal{R}$  for which the quotient map is order preserving. We denote **the set of all regular equivalence relations** over an ordered set  $(X, <)$  by  $\mathfrak{R}(X, <)$ . Likewise, the family of all **regular partitions** of  $(X, <)$  is denoted  $\mathfrak{P}(X, <)$ .

In general, we will denote the induced order relation for a strict poset  $(X, <)$  and a regular equivalence relation  $\mathcal{R} \in \mathfrak{R}(X, <)$  by  $<'$ .

### I.4.2 The structure of regular equivalence relations

We now establish a sufficient and necessary condition for an agglomerative clustering algorithm to be order preserving. Recall that, if  $A \subseteq X$ ,  $X/A$  denotes the quotient for which the quotient map  $q_A : X \rightarrow X/A$  sends all of  $A$  to a point, and is the identity otherwise. That is, for every  $x, y \in X$ , we have

$$q_A(x) = q_A(y) \Leftrightarrow x, y \in A.$$

**Theorem I.4.5.** *If  $A \subseteq X$  for a strict poset  $(X, <)$ , the quotient map  $q_A : X \rightarrow X/A$  is order preserving if and only if  $A$  is an antichain in  $(X, <)$ .*

*Proof.* If  $A$  is not an antichain, then  $X/A$  places comparable elements in the same equivalence class, so  $q_A$  is not order preserving.

Assume  $A$  is an antichain. If  $q_A$  is not order preserving, then there is a cycle in  $(X/A, <')$ , and since we have only one non-singleton equivalence class, the cycle must be on the form

$$b \begin{array}{c} \xleftarrow{\quad} \\ \xrightarrow{\quad} \end{array} A \xrightarrow{\quad} c.$$

But this means we have  $a, a' \in A$  for which  $b < a$  and  $a' < c$ , but since  $c < b$ , this implies  $a' < a$ , contradicting the fact that  $A$  is an antichain. ■

Since a composition of order preserving maps is order preserving, this also applies to a composition of quotient maps for a chain of regular equivalence relations  $\mathcal{R}_1 \subseteq \dots \subseteq \mathcal{R}_n$ . Combining this with Theorem I.4.5, we have the following:

*A clustering of a strict poset will be order preserving if it can be produced as a sequence of pairwise merges of non-comparable elements.*

We close the section with an observation about the family of all hierarchical clusterings over a strict poset:

**Theorem I.4.6.** <sup>1</sup> *For a strict poset  $(X, <)$ , the set  $\mathfrak{P}(X, <)$  of regular partitions over  $(X, <)$  has  $S(X)$  as its least element. Unless  $<$  is the empty order, there is no greatest element.*

*Proof.*  $S(X)$  is always a regular partition, so  $S(X) \in \mathfrak{P}(X, <)$ . And since  $S(X)$  is a refinement of every partition of  $X$ ,  $S(X)$  is the least element of  $\mathfrak{P}(X, <)$ .

If the order relation is not empty, then there are at least two elements that are comparable, and, according to Theorem I.4.5, they cannot be in the same equivalence class. Hence, there is no greatest element. ■

---

<sup>1</sup>Thm I.4.6 has been identified to contain an error. As the theorem appears in a published paper, it is stated here in its original form. A corrected version of the theorem is made available in Appendix A.

The situation of Theorem I.4.6 is depicted in Figure I.2, and has already been discussed in Section I.1.2: In the case of tied connections that represent mutually exclusive merges, choosing to merge one connection over the other may lead to very different results. We therefore need a strategy to select one of these solutions over the others. This will be the main focus of Sections I.5 and I.6.

## I.5 Partial dendrograms

In this section we formally define partial dendrograms, and then construct the embedding of partial dendrograms into ultrametrics.

Based on the discussion of partial dendrograms in Section I.1.2 together with the definition of dendrograms (Definition I.2.3), we suggest the following:

**Definition I.5.1.** A **partial dendrogram over  $(X, <)$**  is an order preserving map  $\theta : \mathbb{R}_+ \rightarrow \mathfrak{P}(X, <)$  satisfying properties D1 and D2 of Definition I.2.3.

The only difference between a dendrogram and a partial dendrogram is that for a partial dendrogram we do not require the existence of a greatest element in the image of  $\theta$ . Partial dendrograms are clearly a generalisation of dendrograms. To distinguish between the two, we will occasionally refer to the non-partial dendrograms as **complete dendrograms**. We denote **the family of partial dendrograms over  $(X, <)$**  by  $\mathcal{PD}(X, <)$ .

For a partial dendrogram  $\theta$ , we will write  $\theta(\infty)$  to denote the maximal partition in the image of  $\theta$ . Since  $\mathfrak{P}(X, <)$  is finite, a partial dendrogram  $\theta \in \mathcal{PD}(X, <)$  is *eventually constant*; that is, there exists a positive real number  $t_0$  for which

$$t \geq t_0 \Rightarrow \theta(t) = \theta(\infty).$$

We refer to this number as the **diameter** of  $\theta$ ; formally,

$$\text{diam}(\theta) = \min\{x \in \mathbb{R}_+ \mid \theta(x) = \theta(\infty)\}.$$

We now turn to the task of constructing the embedding. Looking at the partial dendrograms of Figure I.3, each connected component in a partial dendrogram is a complete dendrogram over its leaf nodes. Since complete dendrograms map to ultrametrics, each connected component gives rise to an ultrametric on the subset of  $X$  constituted by the connected component's leaf nodes. That is, if  $\theta(\infty) = \{B_j\}_{j=1}^k$ , and if  $\theta_j$  is the complete dendrogram over  $B_j$  for  $1 \leq j \leq k$ , we can define the ultrametrics  $u_j = \Psi_{B_j}(\theta_j)$  so that  $\{(B_j, u_j)\}_{j=1}^k$  is a disjoint family of ultrametric spaces, which union covers  $X$ .

Now consider the following general result.

**Lemma I.5.2.** *Given a family of bounded, disjoint ultrametric spaces  $\{(X_j, d_j)\}_{j=1}^n$  together with a positive real number  $K \geq \max_j \{\text{diam}(X_j, d_j)\}$ , the map*

$$d_{\cup} : \bigcup X_j \times \bigcup X_j \longrightarrow \mathbb{R}_+$$

## I. Order preserving hierarchical agglomerative clustering

---

given by

$$d_{\cup}(x, y) = \begin{cases} d_j(x, y) & \text{if } \exists j : x, y \in X_j, \\ K & \text{otherwise} \end{cases}$$

is an ultrametric on  $\bigcup_j X_j$ .

*Proof.* To prove that the ultrametric inequality holds, we start by showing that  $d_{\cup_{1,2}}$  is an ultrametric on the restriction to the disjoint union  $X_1 \cup X_2$ : Let  $x, y \in X_1$  and  $z \in X_2$ , and choose a positive  $K \geq \max\{\text{diam}(X_1, d_1), \text{diam}(X_2, d_2)\}$ . We now have

$$d_{\cup_{1,2}}(x, z) = K \quad d_{\cup_{1,2}}(x, y) = d_1(x, y) \quad d_{\cup_{1,2}}(y, z) = K.$$

This means that every triple of points are either already contained in an ultrametric space, or they make up an isosceles triangle. In both cases, the ultrametric inequality holds, according to the observation in Example I.2.4.

By induction, we can now prove that  $((X_1 \cup X_2) \cup X_3, d_{\cup_{1,2,3}})$  is an ultrametric space, and so on, until all the  $(X_j, d_j)$  are included. ■

Hence, for our partial dendrogram  $\theta$  with  $\theta(\infty) = \{B_j\}_{j=1}^k$  and subspace ultrametrics  $\{u_j\}_{j=1}^k$ , pick a  $K \geq \max_j\{\text{diam}(B_j, u_j)\}$ , and define  $u_\theta : X \times X \rightarrow \mathbb{R}_+$  by

$$u_\theta(x, y) = \begin{cases} u_j(x, y) & \text{if } \exists j : x, y \in B_j, \\ K & \text{otherwise.} \end{cases} \quad (\text{I.10})$$

According to Lemma I.5.2, equation (I.10) is an ultrametric on  $X$ .

**Definition I.5.3.** Given an ordered space  $(X, <)$  and a non-negative real number  $\varepsilon$ , the **ultrametric completion on  $\varepsilon$**  is the map  $\mathfrak{U}_\varepsilon : \mathcal{PD}(X, <) \rightarrow \mathcal{U}(X)$  mapping

$$\mathfrak{U}_\varepsilon : \theta \mapsto u_\theta,$$

where  $u_\theta$  is defined as in (I.10), setting  $K = \text{diam}(\theta) + \varepsilon$ .

**Example I.5.4.** To illustrate how the ultrametric completion turns out in the case of the partial dendrograms of Figure I.3, we have the following figure:

The above discussion serves to show that the construction is well defined. Our next goal is two-fold. First, we wish to provide an (explicit) function from partial dendrograms to dendrograms that realises this map. And second, we wish to establish conditions for this function to be an embedding; that is, an injective map. Injectivity is not strictly required for the theory to work, but it increases its discriminative power. An example to the contrary is provided towards the end of the section.

We have the map  $\Psi_X : \mathcal{D}(X) \rightarrow \mathcal{U}(X)$  from (I.1), mapping dendrograms to ultrametrics. We now seek a map  $\kappa_\varepsilon : \mathcal{PD}(X, <) \rightarrow \mathcal{D}(X)$  making the following



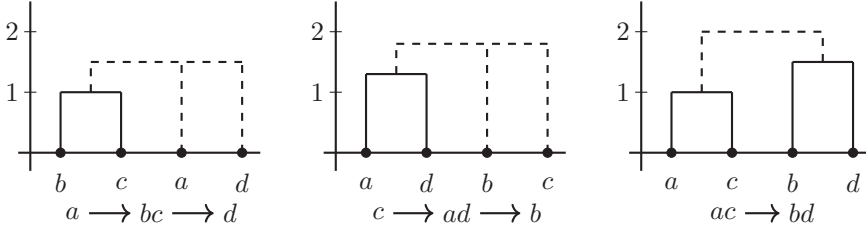


Figure I.4: “Completed” dendrograms corresponding to the partial dendrograms of Figure I.3, using  $\varepsilon = 0.5$ . The completions are marked by the dashed lines.

diagram commute:

$$\begin{array}{ccc}
 \mathcal{D}(X) & \xrightarrow{\Psi_X} & \mathcal{U}(X) \\
 \uparrow \kappa_\varepsilon & \nearrow \mathfrak{U}_\varepsilon & \\
 \mathcal{PD}(X, <) & & 
 \end{array} \quad . \quad (I.11)$$

Seeing that  $\kappa_\varepsilon$  must map partial dendrograms to complete dendrograms, a quick glance at Figure I.4 suggests the following:

$$\kappa_\varepsilon(\theta)(x) = \begin{cases} \theta(x) & \text{for } x < \text{diam}(\theta) + \varepsilon \\ \{X\} & \text{otherwise.} \end{cases}$$

It is straightforward to check that  $\kappa_\varepsilon(\theta)$  is a complete dendrogram.

**Theorem I.5.5.**  $\Psi_X \circ \kappa_\varepsilon = \mathfrak{U}_\varepsilon$ . That is; diagram (I.11) commutes.

*Proof.* Assume first that  $\theta \in \mathcal{PD}(X, <)$  is a proper partial dendrogram, and that  $\text{im}(\theta) = \{\mathcal{B}_i\}_{i=0}^n$ . Let the coarsest partition in the image of  $\theta$  be given by  $\mathcal{B}_n = \{B_j\}_{j=1}^m$ . That is, each block  $B_j$  corresponds to a connected component in the partial dendrogram. Pick a block  $B \in \mathcal{B}_n$  and assume  $x, y \in B$ .

If

$$k = \min\{i \in \mathbb{N} \mid \exists B' \in \mathcal{B}_i : B \subseteq B'\},$$

then  $\mathcal{B}_k$  is the finest partition containing all of  $B$  in one block. Since  $B \subseteq X$ , the partitions

$$\mathcal{B}_i^B = \{B \cap B' \mid B' \in \mathcal{B}_i\} \quad \text{for } 1 \leq i \leq k$$

constitute a chain in  $\mathfrak{P}(B)$  containing both  $S(B)$  and  $\{B\}$ . Hence, we can construct a complete dendrogram over  $B$  by defining

$$\theta_B(x) = \{B \cap B' \mid B' \in \theta(x)\}. \quad (I.12)$$

## I. Order preserving hierarchical agglomerative clustering

---

This is exactly the complete dendrogram corresponding to the connected component of the tree over  $X$  having the elements of  $B$  as leaf nodes. By Definition I.5.3,

$$x, y \in B \Rightarrow \mathfrak{U}_\varepsilon(\theta)(x, y) = \Psi_B(\theta_B)(x, y). \quad (\text{I.13})$$

Due to (I.12), we have

$$\begin{aligned} x, y \in B &\Rightarrow (\exists B \in \theta_B(x) : x, y \in B \Leftrightarrow \exists B' \in \theta(x) : x, y \in B') \\ &\Rightarrow \min\{t \in \mathbb{R}_+ \mid \exists B \in \theta_B(t) : x, y \in B\} \\ &= \min\{t \in \mathbb{R}_+ \mid \exists B' \in \theta(t) : x, y \in B'\}. \end{aligned}$$

Hence, by the definition of  $\Psi_X$  in (I.1) we conclude that

$$x, y \in B \Rightarrow \Psi_B(\theta_B)(x, y) = (\Psi_X \circ \kappa_\varepsilon)(\theta)(x, y).$$

Combining this with (I.13), we get that whenever  $x, y \in B$ , we have  $\Psi_X \circ \kappa_\varepsilon = \mathfrak{U}_\varepsilon$ .

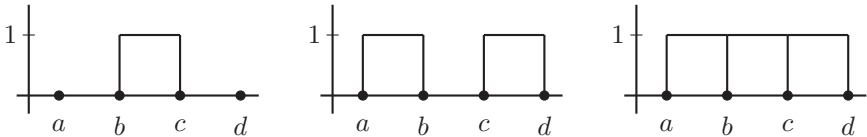
On the other side, let  $x \in B_i$  and  $y \in B_j$  with  $i \neq j$ . By definition, we have  $\mathfrak{U}_\varepsilon(\theta)(x, y) = \text{diam}(\theta) + \varepsilon$ . And, since there is no block in  $\theta(\infty)$  containing both  $x$  and  $y$ , we find that the minimal partition in  $\text{im}(\kappa_\varepsilon(\theta))$  containing  $x$  and  $y$  in one block is  $\{X\}$ . But this means that  $\Psi_X(\kappa_\varepsilon(\theta))(x, y) = \text{diam}(\theta) + \varepsilon$ , so  $\Psi_X \circ \kappa_\varepsilon = \mathfrak{U}_\varepsilon$  holds in this case too.

Finally, if  $\theta$  is a complete dendrogram, we have  $\kappa_\varepsilon(\theta) = \theta$ , so  $\Psi_X \circ \kappa_\varepsilon(\theta) = \Psi_X(\theta)$ . But since  $\theta(\infty) = \{X\}$ , it follows that  $\mathfrak{U}_\varepsilon(\theta)$  maps exactly to the ultrametric over  $X$  defined by  $\Psi_X(\theta)$ . ■

**Theorem I.5.6.** *Let  $(X, <)$  be a strict poset with a non-empty order relation. Then  $\mathfrak{U}_\varepsilon$  is injective if  $\varepsilon > 0$ .*

*Proof.* Injectivity follows if  $\kappa_\varepsilon$  is injective, so assume that  $\kappa_\varepsilon(\theta) = \kappa_\varepsilon(\theta')$ . Then, for every  $x < \text{diam}(\theta) + \varepsilon$ , we have  $\kappa_\varepsilon(\theta)(x) = \kappa_\varepsilon(\theta')(x) \Leftrightarrow \theta(x) = \theta'(x)$ . ■

**Example I.5.7.** If  $\varepsilon$  is not chosen to be strictly positive, the map  $\mathfrak{U}_\varepsilon$  will not necessarily be injective. Consider the below dendrograms.



Both of the partial dendrograms are mapped to the same complete dendrogram (on the right) for  $\varepsilon = 0$ . This illustrates what we mean by *reduced discriminative power* in the case of a non-injective completion. Since the partial dendrograms exhibit distinctively different information, it is desirable that the methodology can distinguish them.

## 1.6 Hierarchical clustering of ordered sets

We are now ready to embark on the specification of order preserving hierarchical clustering of ordered sets. We do this by extending our notion of optimised hierarchical clustering from Section I.3. For the remainder of the paper, let an **ordered dissimilarity space** be denoted by  $(X, <, d)$ .

Consider the following modification of classical hierarchical clustering. The only difference is that for each iteration, we check that there are elements that actually can be merged while preserving the order relation. According to Theorem I.4.5, this means merging a pair of non-comparable elements at each iteration. Recall that  $S(X)$  denotes the singleton partition of  $X$ .

Let  $(X, <, d)$  be given together with a linkage function  $\mathcal{L}$ .

1. Set  $Q_0 = S(X)$ , and endow  $Q_0$  with the induced order relation  $<_0$ .
2. Among the pairs of non-comparable clusters, pick a pair of minimal dissimilarity according to  $\mathcal{L}$ , and combine them into one cluster by taking their union.
3. Endow the new clustering with the induced order relation.
4. If all elements of  $X$  are in the same cluster, or if all clusters are comparable, we are done. Otherwise, go to Step 2 and continue.

The procedure results in a chain of ordered partitions  $\{(Q_i, <_i)\}_{i=0}^m$  together with the dissimilarities  $\{\rho_i\}_{i=0}^m$  at which the partitions were formed. For an ordered set  $(X, <)$ , recall that non-comparability of  $a, b \in X$  is denoted  $a \perp b$ . Let the **non-comparable separation of  $(X, <, d)$** , be given by

$$\text{sep}_\perp(X, <, d) = \min_{x, y \in X} \{d(x, y) \mid x \neq y \wedge x \perp y\}.$$

The reader may wish to compare the following lemma to Remark I.2.7.

**Lemma I.6.1.** *The sequence of pairs  $\{(Q_i, \rho_i)\}_{i=0}^m$  produced by the above procedure maps to a partial dendrogram through application of (I.2) if and only if*

$$\text{sep}_\perp(Q_i, <_i, \mathcal{L}) \leq \text{sep}_\perp(Q_{i+1}, <_{i+1}, \mathcal{L}).$$

Since the singleton partition  $Q_0$  maps to a partial dendrogram, the algorithm will produce a partial dendrogram for any ordered dissimilarity space, and since there can be at most  $|X| - 1$  merges, the procedure always terminates.

As for classical hierarchical clustering, the procedure is non-deterministic in the sense that given a set of tied pairs, we may pick a random pair for the next merge. Hence, the procedure is capable of producing partial dendrograms for all possible tie resolution strategies:

**Definition I.6.2.** Given an ordered dissimilarity space  $(X, <, d)$  and a linkage function  $\mathcal{L}$ , we write  $\mathcal{D}^\mathcal{L}(X, <, d)$  to denote the set of all possible outputs from the above procedure

## I. Order preserving hierarchical agglomerative clustering

---

The set  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  differs from  $\mathcal{D}^{\mathcal{L}}(X, d)$  in two important ways:

- $\mathcal{D}^{\mathcal{L}}(X, <, d)$  contains partial dendrograms, not dendrograms.
- The cardinality of  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  is at least that of  $\mathcal{D}^{\mathcal{L}}(X, d)$ , and often higher, due to mutually exclusive merges and the “dead ends” in  $\mathfrak{P}(X, <)$  (see Figure I.2).

Even for single linkage we have  $|\mathcal{D}^{\mathcal{S}\mathcal{L}}(X, <, d)| > 1$  if there are mutually exclusive tied connections.

In the spirit of optimised hierarchical clustering, we suggest the following definition, employing the ultrametric completion  $\mathfrak{U}_\varepsilon$  from Definition I.5.3:

**Definition I.6.3.** Given an ordered dissimilarity space  $(X, <, d)$  together with a linkage function  $\mathcal{L}$ , let  $\varepsilon > 0$ . An **order preserving hierarchical agglomerative clustering using  $\mathcal{L}$  and  $\varepsilon$**  is given by

$$\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, <, d) = \arg \min_{\theta \in \mathcal{D}^{\mathcal{L}}(X, <, d)} \|\mathfrak{U}_\varepsilon(\theta) - d\|_p. \quad (\text{I.14})$$

The next theorem shows that if we remove the order relation, then optimised clustering and order preserving clustering coincide. Keep in mind that a dissimilarity space is an ordered dissimilarity space with an empty order relation; that is,  $(X, d) = (X, \emptyset, d)$ .

**Theorem I.6.4.** *If the order relation is empty, then order preserving optimised hierarchical clustering and optimised hierarchical clustering coincide:*

$$\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, \emptyset, d) = \mathcal{HC}_{opt}^{\mathcal{L}}(X, d).$$

*Proof.* First, notice that

$$\forall (Q, <_Q) \in \mathfrak{P}(X, \emptyset) : \text{sep}_\perp(Q, <_Q, \mathcal{L}) = \text{sep}(Q, \mathcal{L}),$$

where  $<_Q$  denotes the (trivial) induced order. Hence, we have  $\mathcal{D}^{\mathcal{L}}(X, \emptyset, d) = \mathcal{D}^{\mathcal{L}}(X, d)$ . Since  $\mathfrak{U}_\varepsilon|_{\mathcal{D}(X)} = \Psi_X$ , the result follows.  $\blacksquare$

### I.6.1 On the choice of $\varepsilon$

In  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, <, d)$  we identify the elements from  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  that are closest to the dissimilarity measure  $d$  when measured in the  $p$ -norm. The injectivity of  $\mathfrak{U}_\varepsilon$  induces a relation  $\preceq_{d,\varepsilon}$  on  $\mathcal{PD}(X, <)$  defined by

$$\theta \preceq_{d,\varepsilon} \theta' \Leftrightarrow \|\mathfrak{U}_\varepsilon(\theta) - d\|_p \leq \|\mathfrak{U}_\varepsilon(\theta') - d\|_p,$$

and the optimisation finds the minimal elements under this order.

The choice of  $\varepsilon$  may affect the ordering of dendrograms under  $\preceq_{d,\varepsilon}$ . We show this by providing an alternative formula for  $\|\mathfrak{u} - d\|_p$  that better expresses the effect of the choice of  $\varepsilon$ . Assume  $\theta$  is a partial dendrogram over  $(X, <)$  with  $\theta(\infty) = \{B_i\}_{i=1}^m$ , and let  $\mathfrak{U}_\varepsilon(\theta) = \mathfrak{u}$ . We split the sum for computing  $\|\mathfrak{u} - d\|_p$  in

two: the intra-block differences and the inter-block differences. The **intra-block differences** are independent of  $\varepsilon$ , and are given by

$$\alpha = \sum_{i=1}^m \sum_{x,y \in B_i} |u(x,y) - d(x,y)|^p. \quad (\text{I.15})$$

On the other hand, the **inter-block differences** are dependent on  $\varepsilon$ , and can be computed as

$$\beta_\varepsilon = \sum_{\substack{(x,y) \in B_i \times B_j \\ i \neq j}} |\text{diam}(\theta) + \varepsilon - d(x,y)|^p. \quad (\text{I.16})$$

This yields  $\|u - d\|_p = \sqrt[p]{\alpha + \beta_\varepsilon}$ . If we think of  $u$  as an approximation of  $d$ , and saying that  $|X| = N$ , the mean  $p$ -th error of this approximation can be expressed as a function of  $\varepsilon$ :

$$E_d(\varepsilon|\theta, p) = \frac{1}{N} \|u - d\|_p^p = \frac{\alpha}{N} + \frac{1}{N} \sum_{\substack{(x,y) \in B_i \times B_j \\ i \neq j}} |\text{diam}(\theta) + \varepsilon - d(x,y)|^p.$$

From the formula for  $E_d(\varepsilon|\theta, p)$ , we see that when  $\varepsilon$  becomes large, the inter-block differences dominate the approximation error. Thus, for increasing  $\varepsilon$ , having low error eventually equals having few inter-block pairs. As a result, large  $\varepsilon$  will lead to clusterings where as many elements as possible are placed in one block, since this is the most effective method for reducing the number of inter-block pairs.

On the other hand, a low value of  $\varepsilon$  will move the weight towards optimising the intra-block ultrametric fit and move the bias away from large block sizes.

From the authors' perspective, focusing on block sizes seems to be less in the spirit of ultrametric fitting, compared to optimising the intra-block ultrametric fit. As such, it is the authors' opinion that this points towards selecting a low value for  $\varepsilon$ . In the process of choosing, we have the following result at our aid:

**Theorem I.6.5.** *For any finite ordered dissimilarity space  $(X, <, d)$  and linkage function  $\mathcal{L}$ , there exists an  $\varepsilon_0 > 0$  for which*

$$\varepsilon, \varepsilon' \in (0, \varepsilon_0) \Rightarrow (\mathcal{D}^{\mathcal{L}}(X, <, d), \preceq_{d,\varepsilon}) \approx (\mathcal{D}^{\mathcal{L}}(X, <, d), \preceq_{d,\varepsilon'}).$$

*That is; all  $\varepsilon \in (0, \varepsilon_0)$  induce the same order on the partial dendrograms.*

*Proof.* Since  $X$  is finite,  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  is also finite. And according to  $E_d(\varepsilon|\theta, p)$ , if the cardinality of  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  is  $n$ , there are at most  $pn$  positive values of  $\varepsilon$  that are distinct global minima of partial dendrograms in  $\mathcal{D}^{\mathcal{L}}(X, <, d)$ . But this means there is a finite set of  $\varepsilon$  for which the order on  $(\mathcal{D}^{\mathcal{L}}(X, <, d), \preceq_{\varepsilon,p})$  changes. And since all these values are strictly positive, they have a strictly positive lower bound. ■

Since the value of  $\varepsilon_0$  depends on  $D^{\mathcal{L}}(X, <, d)$ , it is non-trivial to compute. For practical applications, we recommend to choose a very small positive number for  $\varepsilon$ , but not so small that it becomes zero due to floating point rounding when added to the diameter of the partial dendrograms.

### I.6.2 Idempotency of $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}$

A detailed axiomatic analysis along the lines of for example Ackerman and Ben-David [1] is beyond the scope of this paper, and is considered for future work. We still include a proof of idempotency of  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}$ , since this is an essential property of classical hierarchical clustering.

Idempotency of hierarchical clustering necessarily depends on the linkage function. We introduce the following concept, that allows us to prove this property for a range of linkage functions: We say that  $\mathcal{L}$  is a **convex linkage function** if we always have

$$\mathcal{SL}(p, q, d) \leq \mathcal{L}(p, q, d) \leq \mathcal{CL}(p, q, d).$$

Notice that if  $\mathbf{u}$  is an ultrametric on  $X$ , the ultrametric inequality yields

$$\mathbf{u}(a, b) = \text{sep}(X, \mathbf{u}) \Rightarrow \forall c \in X : \mathbf{u}(a, c) = \mathbf{u}(b, c),$$

so if  $\mathcal{L}$  is a convex linkage function and  $\mathbf{u}(a, b) = \text{sep}(X, \mathbf{u})$ , we have

$$\mathcal{L}(\{a, b\}, \{c\}) = \mathcal{L}(\{a\}, \{c\}) = \mathcal{L}(\{b\}, \{c\}) \quad \forall c \neq a, b.$$

This is to say that a convex linkage function preserves the structure of the original ultrametric when minimal dissimilarity elements are merged. As a result, for any  $\mathbf{u} \in \mathcal{U}(X)$ , the set  $\mathcal{D}^{\mathcal{L}}(X, \mathbf{u})$  contains exactly one element, namely the dendrogram corresponding to the ultrametric, which is why classical hierarchical clustering is idempotent.

For ordered spaces, the case is different. It is easy to construct an ordered ultrametric space  $(X, <, \mathbf{u})$  for which  $\mathbf{u}(a, b) = \text{sep}(X, \mathbf{u})$  and  $a < b$ , in which case the ultrametric cannot be reproduced. Hence, all of  $\mathcal{U}(X)$  cannot be fixed points under  $\mathfrak{U}_\varepsilon \circ \mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, <, -)$ , but the mapping is still idempotent:

**Theorem I.6.6** (Idempotency). *For an ordered dissimilarity space  $(X, <, d)$  and a convex linkage function  $\mathcal{L}$ , we have*

$$\theta \in \mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, <, d) \Rightarrow \mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, <, \mathfrak{U}_\varepsilon(\theta)) = \{\theta\}.$$

*Proof.* Let  $\theta(\infty) = \{B_i\}_{i=1}^m$ . Then each  $B_i$  is an antichain in  $(X, <)$ , so we have

$$\forall x, y \in B_i : \text{sep}(B_i, \mathbf{u}|_{B_i}) = \text{sep}_\perp(B_i, \mathbf{u}|_{B_i}) \quad \text{for } 1 \leq i \leq m.$$

Since  $\varepsilon > 0$ , we also have

$$x, y \in B_i \Rightarrow \mathbf{u}(x, y) < \text{diam}(X, \mathbf{u}) \quad \text{for } 1 \leq i \leq m.$$

And, lastly, since every pair of comparable elements are in pairwise different blocks, we have

$$x < y \vee y < x \Rightarrow \mathbf{u}(x, y) = \text{diam}(X, \mathbf{u}).$$

Now, since  $\mathcal{L}$  is convex, based on the discussion preceding the theorem, the intra-block structure of every block will be preserved. And, since every inter-block dissimilarity is accompanied by comparability across blocks, the procedure for generation of  $\mathcal{D}^{\mathcal{L}}(X, <, \mathfrak{U}_\varepsilon(\theta))$  will exactly reproduce the intra block structure of all blocks and then halt. Hence,  $\mathcal{D}^{\mathcal{L}}(X, <, \mathfrak{U}_\varepsilon(\theta)) = \{\theta\}$ . ■

## 1.7 Polynomial time approximation

In the absence of an efficient algorithm for  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}$ , this section provides a polynomial time approximation scheme. The efficacy as approximation is demonstrated in Section I.8, and a demonstration on real world data is given in Section I.9.

Recall the set  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  of partial dendrograms over  $(X, <, d)$  from Definition I.6.2. The algorithm for producing a random element of  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  is described at the beginning of Section I.6; the key is to pick a random pair for merging whenever we encounter a set of tied connections.

The approximation model is deceptively simple; we generate a set of random partial dendrograms, and choose the one with the best ultrametric fit.

**Definition I.7.1.** Let  $(X, <, d)$  be given, and let  $N$  be a positive integer. For any random selection of  $N$  partial dendrograms  $\{\theta_i\}_i$  from  $\mathcal{D}^{\mathcal{L}}(X, <, d)$ , an  **$N$ -fold approximation of  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}(X, <, d)$**  is a partial dendrogram  $\theta \in \{\theta_i\}_i$  minimising  $\|\mathfrak{U}_\varepsilon(\theta) - d\|_p$ . We denote the  $N$ -fold approximation scheme by  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$ .

### 1.7.1 Running time complexity

Assume that  $|X| = n$ . In the worst case, we may have to check  $\binom{n}{2}$  pairs to find one that is not comparable, and the test for  $a \perp b$  has complexity  $O(n^2)$ , leading to a complexity of  $O(n^4)$  of finding a mergeable pair. Since there are up to  $n - 1$  merges, the worst case estimate of the running time complexity for producing one element in  $\mathcal{D}^{\mathcal{L}}(X, <, d)$  is  $O(n^5)$ .

A part of this estimate is the number of comparability tests we have to perform in order to find a mergeable pair. For a sparse order relation, we may have to test significantly less than  $\binom{n}{2}$  pairs before finding a mergeable pair: if  $K$  is the expected number of test we have to do, the expected complexity of finding a mergeable pair becomes  $O(Kn^2)$ . This yields a total expected algorithmic complexity of  $O(Kn^3)$ . If the order relation is empty, we have  $K = 1$ , and the complexity of producing a dendrogram becomes  $O(n^3)$ , which is the running time complexity of classical hierarchical clustering. Hence, if the order relation is sparse, we can generally expect the algorithm to execute significantly faster than the worst case estimate.

When producing an  $N$ -fold approximation, the  $N$  random partial dendrograms can be generated in parallel, reducing the computational time of the approximation. For the required number of dendrograms to obtain a good approximation, please see Section I.8.

### I.8 Demonstration of approximation efficacy on randomly generated data

The purpose of the demonstration is to check to which degree the approximation reproduces the order preserving clusterings of  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}$ . We start by describing the random data model and the quality measures we use in assessing the efficacy of the approximation, before presenting the experimental setup and the results.

#### I.8.1 Random ordered dissimilarity spaces

To test the correctness and convergence ratio of the approximation scheme, we employ randomly generated ordered dissimilarity spaces. The random model consists of two parts: the random partial order and the random dissimilarity measure.

##### I.8.1.1 Random partial order

A partial order is equivalent to a transitively closed directed acyclic graph, so we can use any random model for directed acyclic graphs to generate random partial orders. We choose to use the classical Erdős-Rényi random graph model [4]. Recall that a directed acyclic graph on  $n$  vertices is a binary  $n \times n$  adjacency matrix that is *permutation similar* to a strictly upper triangular matrix; that is, there exists a permutation that, when applied to both the rows and the columns of one matrix, transforms it into the other. Let this family of  $n \times n$  matrices be denoted by  $\mathbb{A}(n)$ . For a number  $p \in [0, 1]$ , the sub-family  $\mathbb{A}(n, p) \subseteq \mathbb{A}(n)$  is defined as follows: for  $A \in \mathbb{A}(n)$ , let  $A'$  be strictly upper triangular and permutation similar to  $A$ . Then each entry above the diagonal of  $A'$  is 1 with probability  $p$ . The sought partial order is the transitive closure of this graph; we denote the corresponding set of transitively closed directed acyclic graphs by  $\overline{\mathbb{A}}(n, p)$ .

##### I.8.1.2 Random dissimilarity measure

If  $|X| = n$ , a dissimilarity measure over  $X$  with no tied connections consists of  $\binom{n}{2}$  distinct values. Hence, any permutation of the sequence  $\{1, \dots, \binom{n}{2}\}$  is a non-tied random dissimilarity measure over  $X$ .

To generate tied connections, let  $t \geq 1$  be the **expected number of ties per level**. That is, for each unique value in the dissimilarity measure, that value is expected to have multiplicity  $t$ . In the case where  $t$  does not divide  $\binom{n}{2}$ , we resolve this by setting the multiplicity of the largest dissimilarity to  $\binom{n}{2} \bmod t$ .



We write  $\mathbb{D}(n, t)$  to denote the family of random dissimilarity measures over sets of  $n$  elements with an expected number of  $t$  ties per level.

**Definition I.8.1.** Given positive integers  $n$  and  $t$  together with  $p \in [0, 1]$ , the family of **random ordered dissimilarity spaces generated by  $(n, p, t)$**  is given by

$$\mathbb{O}(n, p, t) = \overline{\mathbb{A}}(n, p) \times \mathbb{D}(n, t).$$

## I.8.2 Measures of cluster quality

In the demonstration, we start by generating a random ordered dissimilarity space. We then run the optimal clustering method on the space, finding the optimal order preserving hierarchical clustering. Finally, we run the approximation scheme on the space and study to which degree the approximation manages to reproduce the optimal hierarchical clustering. For this, we need a quantitative measure of clustering quality relative a known optimum.

A large body of literature exists on the topic of comparing clusterings (see for instance [38] for a brief review). We have landed on the rather popular *adjusted Rand index* [19] to measure the ability of the approximation in finding a decent partition, comparing against the optimal result.

Less work is done on this type of comparison for partial orders and directed acyclic graphs. We suggest to use a modified version of the adjusted Rand index for this purpose too, based on an adaptation of the Rand index used for network analysis [17]. For an introduction to the Rand index, and also to some of the versions of the adjusted Rand index, see [13, 19, 36].

### I.8.2.1 Adjusted Rand index for partition quality

The Rand index compares two clusterings by computing the percentage of corresponding decisions made in forming the clusterings; that is, counting whether pairs of elements are placed together in both clusterings or apart in both clusterings. An adjusted Rand index reports in the range  $(-\infty, 1]$ , where zero is equivalent to a random draw, and anything above zero is better than chance. We use the adjusted Rand index (ARI) to compute the efficacy of the approximation in finding a partition close to a given planted partition. This corresponds to what Gates and Ahn [13] refers to as a *one sided* Rand index, since one of the partitions are given, whereas the other is drawn from some distribution. In the below demonstration, we assume that the approximating partition is drawn from the set of all partitions over  $X$  under the uniform distribution.

### I.8.2.2 Adjusted Rand index for induced order relations

When comparing induced orders on partitions over a set, unless the partitions coincide, it is not obvious which blocks in one partition correspond to which blocks in the other. To overcome this problem, we base our measurements on the base space projection:

**Definition I.8.2.** For an ordered set  $(X, E)$  and a partition  $Q$  of  $X$  with induced order  $E'$ , the **base space projection of  $(Q, E')$  onto  $X$**  is the order relation  $E_Q$  on  $X$  defined as

$$(x, y) \in E_Q \Leftrightarrow ([x], [y]) \in E'.$$

This allows us to compare the induced orders in terms of different orders on  $X$ . Notice that if the induced order  $E'$  is a [strict] partial order on  $Q$ , then  $E_Q$  is a [strict] partial order on  $X$ .

Hoffman, Steinley and Brusco [17] demonstrate that the adjusted Rand index can be used to detect missing links in networks by computing the similarity of edge sets. The concept relies on the fact that a network link and a link in an equivalence relation are not that different: Both networks and equivalence relations are special classes of relations, and the Rand index simply counts the number of coincidences and mismatches between two relation sets. While Hoffman, Steinley and Brusco [17] uses the ARI to compare elements within a network, we use the same method to compare across networks.

Let  $A$  and  $B$  be the adjacency matrices of two base space projections, and let  $A_i$  denote the  $i$ -th row of  $A$ , and likewise for  $B_i$ . If  $\langle a, b \rangle$  is the inner product of  $a$  and  $b$ , we define

$$\begin{aligned} a_i &= \langle A_i, B_i \rangle & c_i &= \langle A_i, 1 - B_i \rangle \\ b_i &= \langle 1 - A_i, B_i \rangle & d_i &= \langle 1 - A_i, 1 - B_i \rangle. \end{aligned}$$

Here,  $a_i$  is the number of common direct descendants of  $i$  in both relations,  $b_i$  is the number of descendants of  $i$  found in  $A$  but not in  $B$ ,  $c_i$  is the number of descendants of  $i$  in  $B$  but not in  $A$ , while  $d_i$  counts the common non-descendants of  $i$  in the two relations. Using this, we can compute the **element wise adjusted order Rand index**

$$\bar{\text{ARI}}_i = \frac{2(a_i d_i - b_i c_i)}{(a_i + b_i)(b_i + d_i) + (a_i + c_i)(c_i + d_i)} \quad \text{for } 1 \leq i \leq n,$$

measuring the element wise order correlation between the base space projections in the Hubert-Arabie adjusted Rand index [19, 40]<sup>2</sup>. Notice that we compare the  $i$ -th row in  $A$  to the  $i$ -row in  $B$  since these rows correspond to the projections' respective descendant relations for the  $i$ -th element in  $X$ . In [17], the above index is computed for each element pair *within* the network to produce the intra-network similarity coefficient.

Since we are interested in the overall match, we choose to report on the mean value, defining the **adjusted order Rand index for  $A$  and  $B$**  as

$$\bar{\text{ARI}}(A, B) = \frac{1}{n} \sum_{i=1}^n \bar{\text{ARI}}_i.$$

---

<sup>2</sup>This particular formulation of the adjusted Rand index relies on the networks having known and fixed labels, so that we know which vertices map to which vertices [40], which indeed holds for the base space projections of two different induced order relations.

### 1.8.2.3 Normalised ultrametric fit

A natural choice of quality measure is to report the **ultrametric fit**  $\|\mathfrak{U}_\varepsilon(\theta) - d\|_p$  of the obtained partial dendrogram  $\theta$ , especially if we can compare it to the ultrametric fit of the optimal solution. The scale of the ultrametric fit depends heavily on both the size of the space and the order of the norm, so we choose to normalise. Also, we invert the normalised value, so that the optimal fit has a value of 1, and a worst possible fit has value 0. This makes it easy to compare the convergence of the ultrametric fit to the convergence of the ARI and  $\bar{\text{ARI}}$ .

**Definition 1.8.3.** Given a set of partial dendrograms  $\{\theta_i\}$  over  $(X, <, d)$ , let their respective ultrametric fits be given by  $\delta_i = \|\mathfrak{U}_\varepsilon(\theta_i) - d\|_p$ . The **normalised ultrametric fit** are the corresponding values

$$\hat{\delta}_i = 1 - \frac{\delta_i - \min_i\{\delta_i\}}{\max_i\{\delta_i\} - \min_i\{\delta_i\}}.$$

In the presence of a reference solution, we substitute  $\min_i\{\delta_i\}$  with the ultrametric fit of the reference.

### 1.8.2.4 Ultrametric fit relative the optimal ultrametric

The reference partition can be reached through different sequences of merges, and neither  $\mathcal{AL}$  nor  $\mathcal{CL}$  are invariant in this respect. Neither ARI,  $\bar{\text{ARI}}$  nor ultrametric fit captures the match between the optimal hierarchy and the approximated hierarchy. We therefore also include plots of the difference between the optimal ultrametric  $\mathbf{u}_{opt}$  and the approximated ultrametric  $\mathbf{u}_{N,\varepsilon}$ . Since both ultrametries are equivalent to their respective hierarchies, the magnitude  $\|\mathbf{u}_{opt} - \mathbf{u}_{N,\varepsilon}\|_p$  can be interpreted as a measure of difference in hierarchies. In the below plots, this is reported as *opt.fit*. As for the ultrametric fit, we normalise and invert the values for easy comparison.

## 1.8.3 Demonstration on randomly generated data

The experiments in the demonstration split in two. First, we demonstrate the efficacy of the approximation relative a known optimal solution, to see to which degree  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  manages to approximate  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}$ . Second, we study the convergence rate of the ultrametric fit for larger spaces with much larger numbers of tied connections; spaces for which the optimal algorithm does not terminate within any reasonable time.

For each parameter combination in Table I.1, a set of 30 random ordered dissimilarity spaces are generated. For each space, 100 approximations are generated according to the prescribed procedure. We then bootstrap the approximations to generate  $N$ -fold approximations for different  $N$ .

We present the results in terms of convergence plots, showing the efficacy of the approximation as a function of the sample size  $N$ . For the results where a reference solution is available, the plots contain four curves:

## I. Order preserving hierarchical agglomerative clustering

---

	$n$	link probability ( $p$ )	expected ties ( $t$ )	reference
Figure I.5	200	0.01, 0.02, 0.05	5	yes
Figure I.6	200	0.05	3, 7	yes
Figure I.7	500	0.01	10, 50, 100	no
Figure I.8	500	0.05	50, 100	no
Figure I.9	500	0.10	100	no

Table I.1: Parameter settings for the demonstrations. Each of the presented parameter settings are executed for  $\mathcal{L} \in \{\mathcal{SL}, \mathcal{AL}, \mathcal{CL}\}$ . The right-most column indicates whether the reference clustering is available or not. The left-most column refers to the figure wherein the outcome of the corresponding experiment is presented. The parameters have been chosen to illustrate how the algorithm behaviour changes with changing expected number of ties, changing link probability in the random partial order, and choice of linkage function.

- $\mathbb{E}(\text{ARI})$  - The expected adjusted Rand index of the approximated partition.
- $\mathbb{E}(\bar{\text{ARI}})$  - The expected adjusted Rand index of the approximated induced order.
- norm.fit* - The mean of the normalised fit.
- opt.fit* - The mean of the normalised difference between the approximated ultrametric and the optimal ultrametric.

For the results where no reference solution is available, we present the distribution of the normalised fit.

The results are presented in Figures I.5, I.6, I.7 and I.9 on pages 59, 60, 61 and 62, respectively. The parameter settings corresponding to the figures are given in Table I.1 for easy reference, and are also repeated in the figure text.

As we can see from the below results, the approximation generally performs very well. We also see that a large expected number of tied connections requires larger sample size for a good approximation, while a more dense order relation (higher value of  $p$ ) seems to require a smaller sample compared to a more sparse relation. We also see that there is a seemingly strong correlation between the ultrametric fit of the approximation and the similarity between the approximation ultrametric and the optimal ultrametric.

Regarding choice of linkage function, the approximation only requires small samples for both  $\mathcal{SL}$  and  $\mathcal{AL}$ , while  $\mathcal{CL}$  requires larger samples for larger numbers of tied connections.

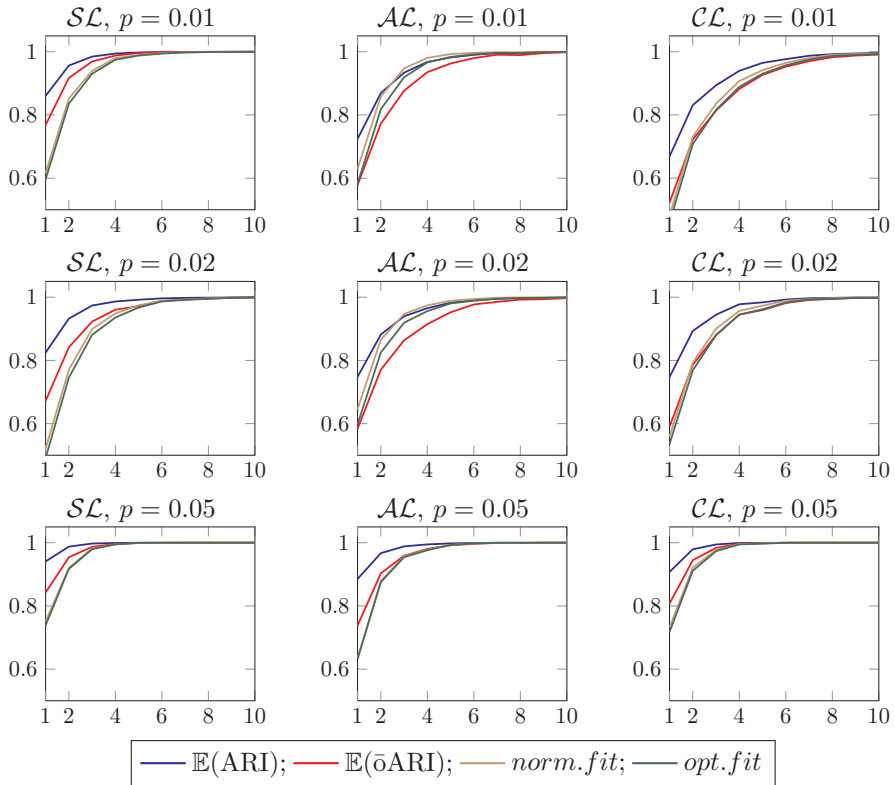


Figure I.5: Efficacy for  $n = 200$  and  $t = 5$  with  $p \in \{0.01, 0.02, 0.05\}$ . The first axis is the size of the drawn sample.

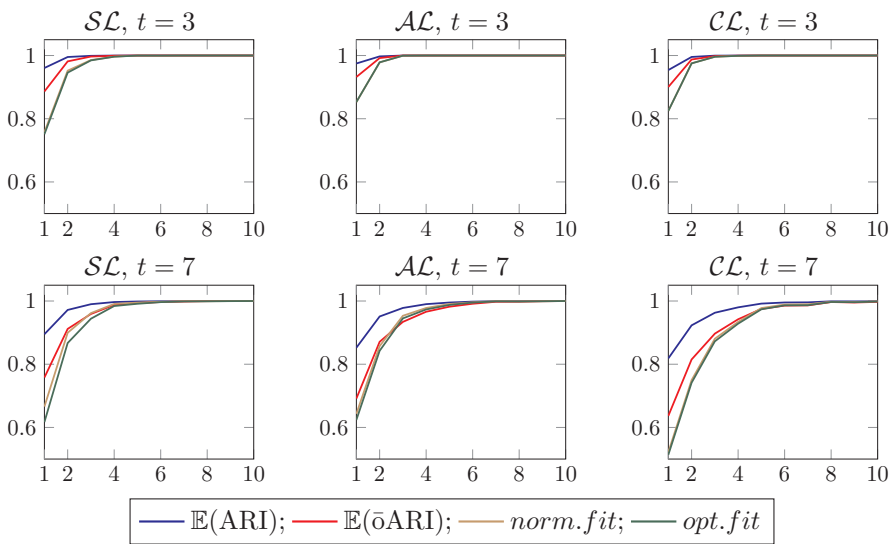


Figure I.6: Efficacy for  $n = 200$  and  $p = 0.05$  with  $t \in \{3, 7\}$ . The first axis is the size of the drawn sample. The plots for  $t = 5$  can be found in the bottom row of Figure I.5.

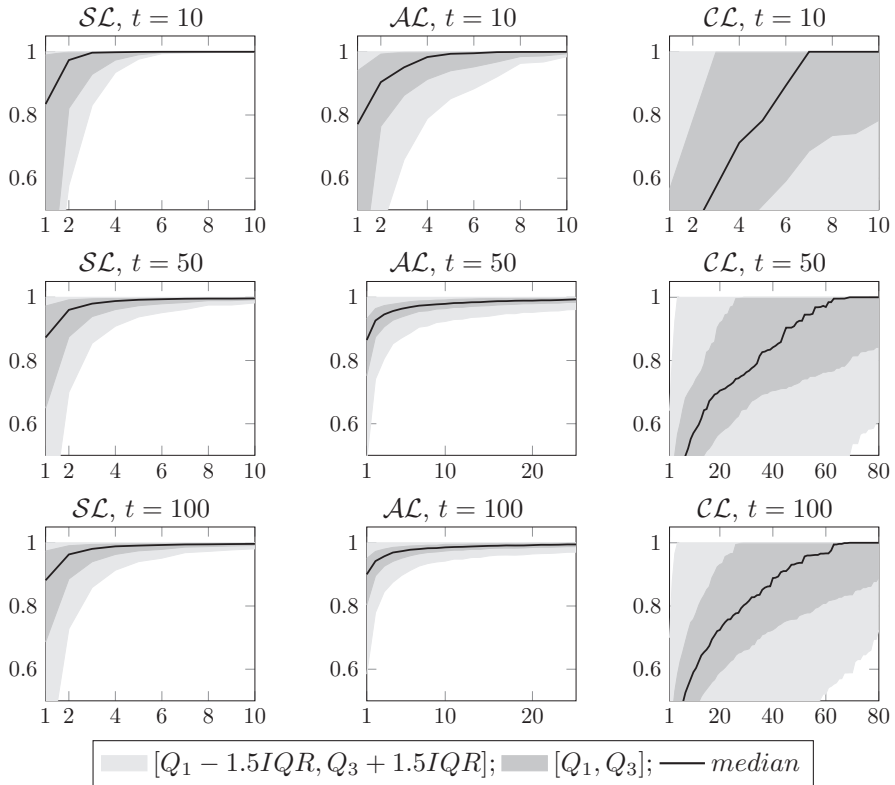


Figure I.7: Polynomial approximation rate for  $n = 500$ ,  $P = 0.01$  and  $t \in \{10, 20, 40\}$ . The first axis is the size of the drawn sample.

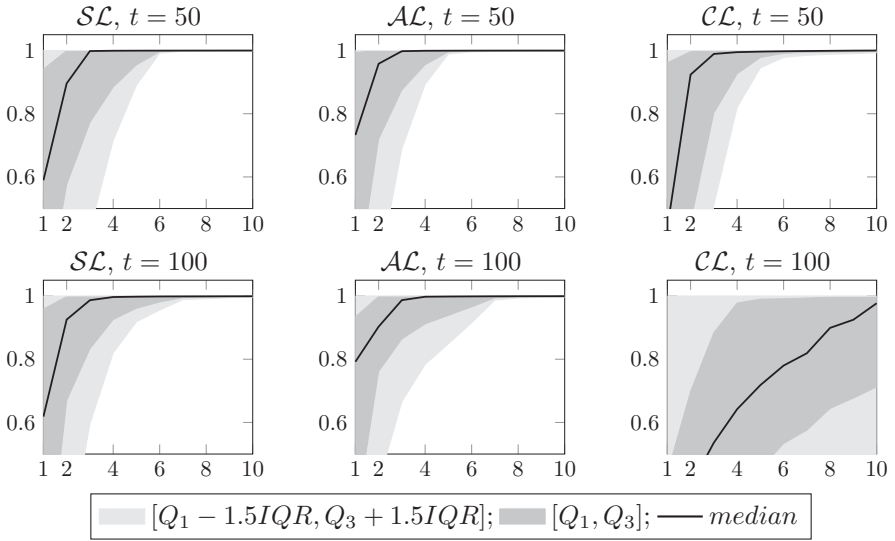


Figure I.8: Polynomial approximation rate for  $n = 500$ ,  $p = 0.05$  and  $t \in \{50, 100\}$ . The first axis is the size of the drawn sample.

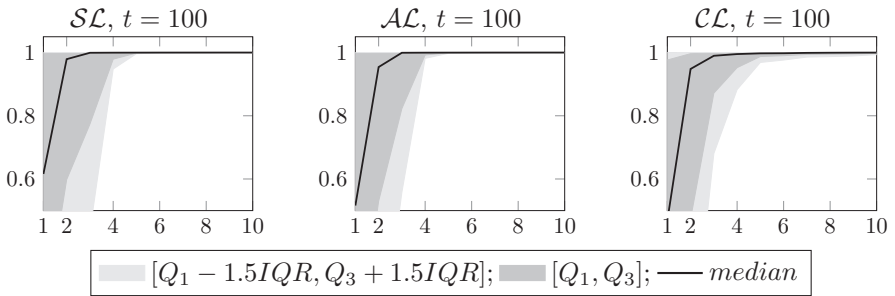


Figure I.9: Polynomial approximation rate for  $n = 500$ ,  $p = 0.10$  and  $t = 100$ . The first axis is the size of the drawn sample.



### I.8.3.1 First conclusions

The first thing that strikes the eye is that the approximations converge very rapidly. Even for moderately sized spaces ( $\sim 500$  elements), it appears to be sufficient with 20 samples for  $\mathcal{SL}$  and  $\mathcal{AL}$ , and for smaller spaces ( $\sim 200$  elements), even fewer samples are required. We also notice that there is a strong correlation between the ARI,  $\bar{\text{ARI}}$  and normalised fit.

For the part of the demonstration where we have no reference clustering, we cannot know for sure whether the best reported fit is also optimal. However, from the convergent behaviour of the data, and the strong correlation between optimality and normalised fit in Figures I.5 and I.6, this points in the direction of convergence to the true optimum.

Only  $\mathcal{CL}$  displays convergence issues, indicating that if one wishes to use  $\mathcal{CL}$  for large spaces or large numbers of tied connections, it may be wise to do so in conjunction with convergence tests.

On the other hand, since  $\mathcal{SL}$  is independent of tie resolution order, every sequence of merges ending in the same maximal partition will produce the same partial dendrogram. This explains why the convergence rate of  $\mathcal{SL}$  is less affected by the expected number of tied connections than, say,  $\mathcal{CL}$ .

The convergence rate is very high in some of the plots of Figures I.8 and I.9. The authors believe this is due the high probability of two random elements being comparable (high  $p$  in  $\bar{\mathcal{O}}(n, p, t)$ ), since a dense relation leads to fewer candidate solutions. This in contrast to the larger set of candidates for a more sparse relation, such as in Figure I.7.

On the other hand, as we can see in Figures I.7 and I.8, keeping  $p$  fixed and increasing the number of tied connections, and thereby the number of possible branch points, causes a slower convergence rate.

To summarise, we see that the approximation is both good and effective for  $\mathcal{SL}$  and  $\mathcal{AL}$ . For  $\mathcal{CL}$ , although the approximation method seems good, the required sample size must be increased in the presence of large amounts of tied connections.

## I.9 Demonstration on data from the parts database

While the above demonstration shows that  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  performs well with respect to approximating  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{L}}$ , another question is how order preserving hierarchical clustering deals with the dust of reality. In this section, we present results from applying the approximation algorithm to subsets of the parts database described briefly in Section I.1.1. As benchmark, we run classical hierarchical clustering on the same problem instances, comparing the performance of the methods using ARI,  $\bar{\text{ARI}}$  and loop frequency (described below). As hierarchical methods for constrained clustering do not offer a no-link constraint, we also propose a simplified approach simulating no-link behaviour for  $\mathcal{AL}$  and  $\mathcal{CL}$  which we call  $\mathcal{HC}^+$ .

### I.9.1 Demonstration dataset

To select data for the demonstration we proceeded as follows: We considered the part-of relations as a directed graph, and extracted all the connected components. As it turned out, there was one gigantic component and a large number of singleton elements, but also a hand-full of connected components of 11 to 40 elements each. We selected these smaller connected components as our demo dataset without any further consideration. Dissimilarities between the elements were obtained from a dissimilarity measure produced by an ongoing project in the company working on the very task of classifying equivalent equipment. Some key characteristics of the data is provided in Table I.2

cc no.	cc size	in/out deg.	$p$	expected ties
0	12	0.92	0.17	2.36
1	14	0.93	0.14	4.79
2	13	0.92	0.15	2.17
3	40	1.27	0.07	8.97
4	20	1.35	0.14	3.96
5	11	1.18	0.24	2.20
6	20	1.10	0.12	4.22
7	20	0.95	0.10	3.96

Table I.2: Some key characteristics of the connected components selected for the demonstration. The in/out deg. column provides the directed average degree when the data is considered as a DAG. The column  $p$  shows the probability for two random elements to be connected in the transitive reduction.

Due to limited labeling of the data, we do not know which elements are copies of other elements, so we have to fake copying to produce planted partitions. For the demonstration, we pick a connected component  $(X^0, E^0)$  where  $X^0 = \{x_1^0, \dots, x_n^0\}$ , and for some positive number  $m$  we make  $m - 1$  copies of  $X^0$  and  $E^0$ , leading to  $m$  partially ordered sets  $\{(X^k, E^k)\}_{k=0}^{m-1}$ . We then form their disjoint union  $(X, E)$  where  $|X| = m|X^0|$ .  $X$  now consists of  $m$  connected components, each a copy of the others. If  $x_i^0 \in X^0$ , then **the set of elements equivalent to  $x_i^0$**  is the set  $\{x_i^k\}_{k=0}^{m-1} \subseteq X$ . Hence, the clusters we seek are the sets on this form.

If we denote the dissimilarity measure that comes with the data by  $d_0$ , we define the extension to all of  $X$  as follows: First, if both elements are in the same component  $X^k$  for  $0 \leq k \leq m$ , then we simply use  $d_0$ . And if they are in different components, indicating that they are in a copy-relationship, we increase their dissimilarity by an offset  $\alpha \geq 0$ . Concretely, the extended dissimilarity  $d^\alpha : X \times X \rightarrow \mathbb{R}_+$  is given by

$$d^\alpha(x_i^r, x_j^s) = \begin{cases} d_0(x_i^0, x_j^0) & \text{if } r = s, \\ \alpha + d_0(x_i^0, x_j^0) & \text{otherwise.} \end{cases}$$

This means that if  $x$  and  $y$  are copies of each other, then  $d^\alpha(x, y) = \alpha$ , and if  $x$  and  $y$  are in the same component and if  $z$  is a copy of  $x$ , then  $d(z, y) = \alpha + d_0(x, y)$ . Furthermore, for each modified distance, we add a small amount of Gaussian noise to  $\alpha$  to induce some variability. As a result, two copies  $x_i^t$  and  $x_i^s$  are offset by approximately  $\alpha$ , and by varying the magnitude of  $\alpha$  we can study how the offset affects the clustering.

### 1.9.2 Simulated constrained clustering

The available methods for hierarchical constrained clustering do not easily incorporate the partial order as a constraint. What we would like to compare against, is hierarchical constrained clustering with do-not-cluster constraints. For  $\mathcal{CL}$  and  $\mathcal{AL}$ , we can obtain this by setting the dissimilarity between comparable elements to a sufficiently large number, causing all comparable elements to be merged towards the end. Indeed, for  $\mathcal{CL}$  it is sufficient to set this dissimilarity to any value exceeding  $\max\{d^\alpha\}$ , and as the below demonstration shows, this value works equally well for  $\mathcal{AL}$ . We denote hierarchical clustering with this kind of modified dissimilarity by  $\mathcal{HC}^{+\mathcal{L}}$ .

Since  $d_0 < 1$  for all pairs of elements, we chose to use 1.0 as our maximum dissimilarity.

### 1.9.3 A measure of order preservation

While the  $\bar{\text{ARI}}$  measures the correlation between the induced order of the planted partition and the induced order of the obtained clustering, the  $\bar{\text{ARI}}$  does not convey information about whether the induced relation is a partial order or not. Since this is a key question for applications where order preservation is of high importance (such as acyclic partitioning of graphs), we suggest the following simple measure.

Let  $(Q, E')$  be a partition of  $(X, E)$ , and let  $E_Q$  be the base space projection of  $(Q, E')$  onto  $X$  (Definition I.8.2). We say that  $(Q, E')$  **induces a loop** if there are elements on the form  $(x, x) \in E_Q$ . The number of loops induced by  $(Q, E')$  is thus the quantity  $|\{(x, y) \in E_Q \mid x = y\}|$ . There is at most one loop per element of  $X$ , and if  $E_Q$  contains a cycle, then every element of the cycle corresponds to a loop. In the name of normalisation, we measure the amount of loops as the fraction of elements in  $X$  that is a part of a cycle:

$$\text{loops}(Q, E') = \frac{|\{(x, y) \in E_Q \mid x = y\}|}{|X|}.$$

### 1.9.4 Picking a clustering in the hierarchy for comparison

Given a problem instance  $(X, <, d)$  and a planted partition  $Q \in \mathfrak{P}(X, <)$ , the planted induced partial order is necessarily the induced relation  $<'$ . But in comparing a hierarchical clustering with a planted partition, we have to make a choice of clustering in the hierarchy. Given a hierarchical clustering, we choose to find the clustering in the hierarchy that has the highest ARI relative the

planted partition. We then report all of ARI,  $\bar{\text{ARI}}$  and loops with regards to this clustering.

### I.9.5 Variance of the difference

In the below plots, we present the mean values of ARI,  $\bar{\text{ARI}}$  and loops together with a visual indication of variability. For each instance of a random ordered dissimilarity space  $(X, <, d)$ , we run all of  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$ ,  $\mathcal{HC}^{\mathcal{L}}$  and  $\mathcal{HC}^{+\mathcal{L}}$ . Thus, we can analyse the performance of the methods by pairwise comparison on a problem instance level. That is, we choose to consider pairwise differences such as

$$\text{ARI}(\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}(X, <, d)) - \text{ARI}(\mathcal{HC}^{+\mathcal{L}}(X, <, d))$$

as one random variable, and likewise for  $\bar{\text{ARI}}$  and loops. The variance of this random variable shows the variance in the difference, and we can use this magnitude to analyse whether the sets of results are statistically distinguishable. For the below plots, we mark a region about each line corresponding to one standard deviation of this random variable. This means that the regions encompassing the lines will not overlap unless the difference between the mean values is less than two standard deviations.

To reduce the number of plots, we choose to plot the results of all three methods together. This is obviously impractical with respect to pairwise comparisons, so we employ the following convention: the indicated variance about the mean of  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  and  $\mathcal{HC}^{+\mathcal{L}}$  is the standard deviation of the differences between these methods. The indicated variance about the mean of  $\mathcal{HC}^{\mathcal{L}}$  represents the standard deviation of the differences between  $\mathcal{HC}^{\mathcal{L}}$  and  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$ .

### I.9.6 Execution and results

The parameters given in Table I.3 define how the ordered dissimilarity spaces are constructed for each of the connected components. For each instance of an ordered dissimilarity space,  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$ ,  $\mathcal{HC}^{\mathcal{L}}$  and  $\mathcal{HC}^{+\mathcal{L}}$  are all run on the same instance with a choice of linkage function  $\mathcal{L} \in \{\mathcal{SL}, \mathcal{AL}, \mathcal{CL}\}$ . This allows us to compare the performance of the methods against each other on a per-instance basis. For each parameter combination in  $\{\alpha\} \times \{\mathcal{SL}, \mathcal{AL}, \mathcal{CL}\}$ , we repeated this process 50 times. The variance of the difference is based on these sets of 50 executions.

We present three families of plots, for ARI,  $\bar{\text{ARI}}$  and loops, respectively. We have picked three connected components for the presentation that we believe represent the span of observations. The full set of plots is provided in the appendix.

First, connected component number 7 (*cc7*) is the sample on which we see the most clear benefit from using  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$ , significantly outperforming both  $\mathcal{HC}^{\mathcal{L}}$  and  $\mathcal{HC}^{+\mathcal{L}}$  on all quality measures. Although *cc7* is not representative for the majority of observations, it is empirical evidence that there exist problem instances for which order preserving clustering cannot be well approximated by hierarchical constrained clustering through do-not-cluster constraints.

parameter	value(s)	explanation
$\alpha$	$\{0.10, 0.15, \dots, 0.50\}$	mean copy dissimilarity
$\sigma$	0.10	variance of $\alpha$
$\mathcal{L}$	$\{\mathcal{SL}, \mathcal{AL}, \mathcal{CL}\}$	linkage models
$m$	$\lceil 200/ X^0  \rceil$	number of copies (see below)
$N$	10	sample size in the $N$ -fold approximation
$\varepsilon$	$10^{-12}$	ultrametric completion level
$p$	1	choice of norm for ultrametric fitting

Table I.3: Parameters for execution of experiments. The number  $m$  of copies is the least number for which the total number of elements,  $m|X^0|$ , is at least 200.

Connected component number 1 (*cc1*) represents the majority of the instances. While  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  still is best in class with respect to all quality measures, we see that for  $\mathcal{AL}$  and  $\mathcal{CL}$  the method  $\mathcal{HC}^{+\mathcal{L}}$  performs equally well with respect to ARI and sometimes also  $\bar{\text{ARI}}$ .

At the other extreme of *cc7* there is connected component number 4 (*cc4*), presented in the bottom row of Figure I.10. For this component, all the clustering models perform equally well in all quality measures, indicating that they produce the exact same clusterings. This can only be explained by the fact that the original dissimilarity measure  $d_0$ , when restricted to this component, both is an ultrametric, and incorporates the order relation (Section I.6.2).

The results are also summarised in Table I.4 after the plots.

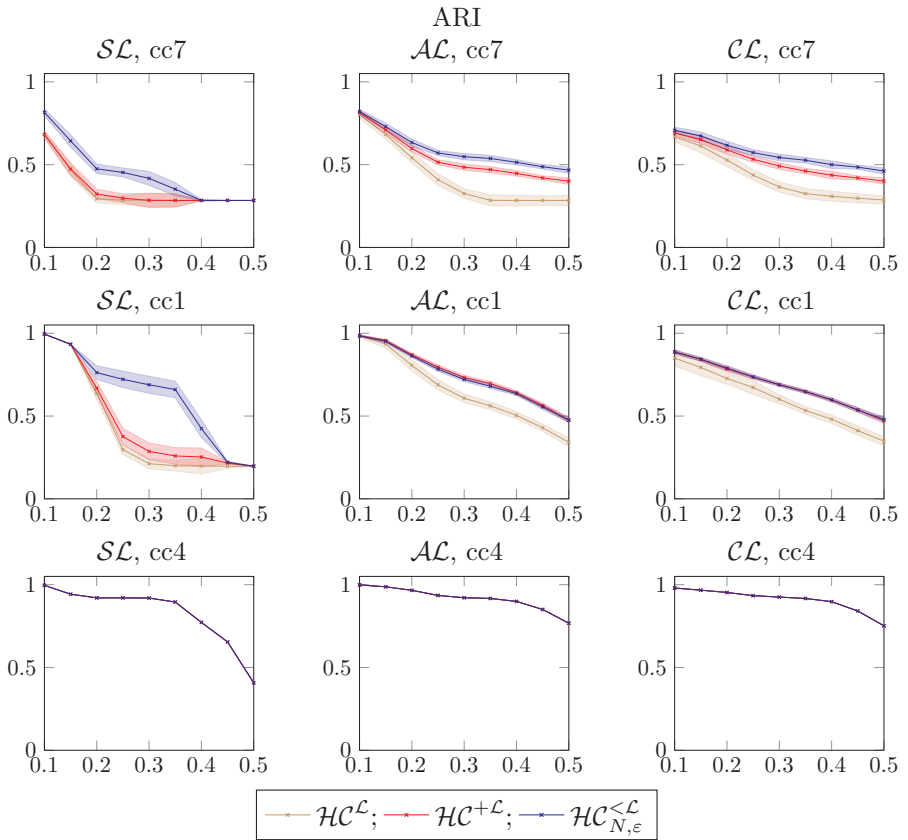


Figure I.10: Performance of the different clustering methods with respect to ARI on connected components 7, 1 and 4. The shaded regions represent one standard deviation of the pairwise differences, as described in Section I.9.5.

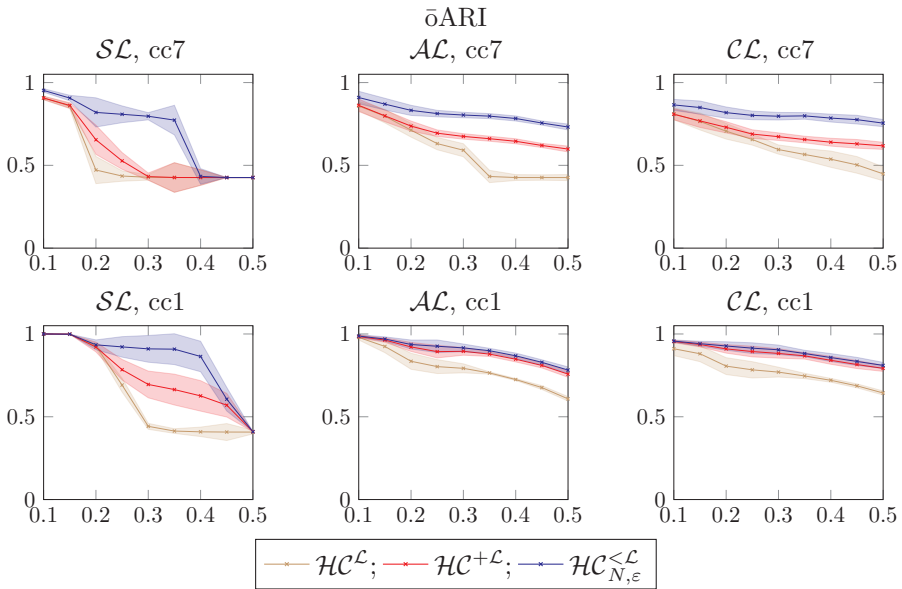


Figure I.11: Performance of the different clustering methods with respect to  $\bar{a}RI$  on connected components 7 and 1. The shaded regions represent one standard deviation of the pairwise differences, as described in Section I.9.5.

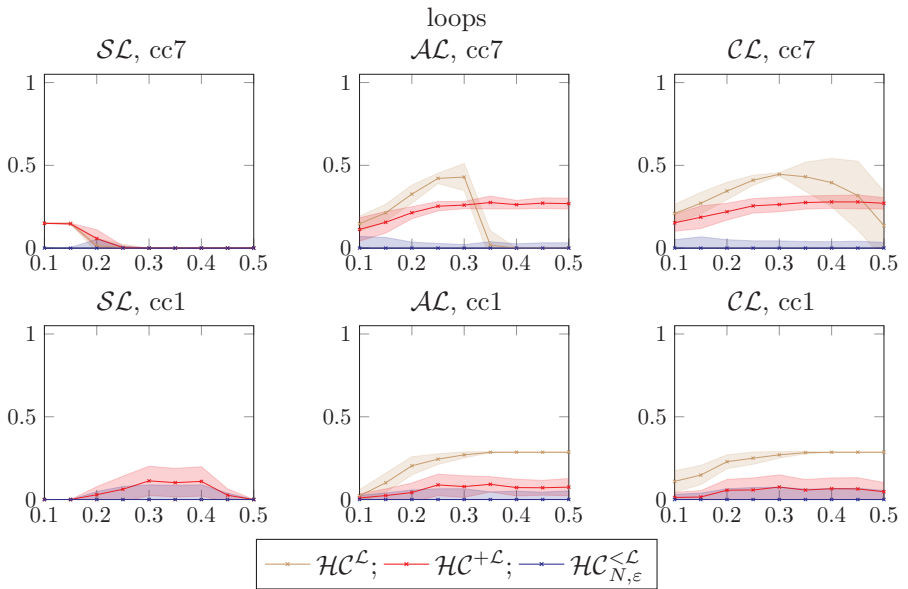


Figure I.12: Performance of the different clustering methods with respect to loops on connected components 7 and 1. The shaded regions represent one standard deviation of the pairwise differences, as described in Section I.9.5.



We summarise the experiment observations in Table I.4. As can be seen from the table,  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  is best in class in every category. However,  $\mathcal{HC}^{+\mathcal{L}}$  is also best in class in 81% of the cases when we restrict our attention to ARI and  $\mathcal{L} \in \{\mathcal{AL}, \mathcal{CL}\}$ .

	$\mathcal{HC}^{<\mathcal{L}}$			$\mathcal{HC}^{\mathcal{L}}$			$\mathcal{HC}^{+\mathcal{L}}$		
	ARI	$\bar{\text{ARI}}$	loops	ARI	$\bar{\text{ARI}}$	loops	ARI	$\bar{\text{ARI}}$	loops
$\mathcal{SL}$	8	8	8	4	4	3	4	4	2
$\mathcal{AL}$	8	8	8	2	2	1	7	6	1
$\mathcal{CL}$	8	8	8	2	3	0	7	6	1
	100%	100%	100%	33%	37%	16%	75%	66%	16%

Table I.4: The table presents for how many of the eight selected samples the different methods are best in class with regards to ARI,  $\bar{\text{ARI}}$  and loops. The scores are based on visual inspection of the plots. For ARI and  $\bar{\text{ARI}}$ , we count a one if there is less than one standard deviation to the best plot in at least half the sampled  $\alpha$  values and zero otherwise. For loops, we count a one if the expected value is zero throughout. The full list of plots can be found in Appendix I.A.

To conclude, we see that if clustering is the sole objective, then  $\mathcal{HC}^{+\mathcal{L}}$  is a good alternative to  $\mathcal{HC}^{<\mathcal{L}}$  whenever  $\mathcal{L} \in \{\mathcal{AL}, \mathcal{CL}\}$ . If order preservation, or acyclic partitioning, is of any importance, then  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  is the only viable method among those we have tested.

Moreover, as demonstrated by the top row of Figure I.10, although  $\mathcal{HC}^{+\mathcal{L}}$  may be a good approximation of  $\mathcal{HC}_{N,\varepsilon}^{<\mathcal{L}}$  when  $\mathcal{L} \in \{\mathcal{AL}, \mathcal{CL}\}$ , there are problem instances on which the latter outperforms the former with significant margin, also for ARI.

## I.10 Summing up

In this paper we have put forth a theory for order preserving hierarchical agglomerative clustering for strictly partially ordered sets. The clustering uses classical linkage functions such as single-, average-, and complete linkage. The clustering is optimisation based, and therefore also permutation invariant.

The output of the clustering process is partial dendrograms; sub-trees of dendrograms with several connected components. We have shown that the family of partial dendrograms over a set embed into the family of dendrograms over the set.

When applying the theory to non-ordered sets, we see that we have a new theory for hierarchical agglomerative clustering that is very close to the classical theory, but that is optimisation based rather than algorithmic. Differently from classical hierarchical clustering, our theory is permutation invariant. We have shown that for single linkage, the theory coincides with classical hierarchical clustering, while for complete linkage, the clustering problem becomes NP-hard.

## I. Order preserving hierarchical agglomerative clustering

---

However, the computational complexity is directly linked to the number of tied connections, and in the absence of tied connections, the theories coincide.

We present a polynomial approximation scheme for the clustering theory, and demonstrate its convergence properties and efficacy on randomly generated data. We also provide a demonstration on real world data comparing against existing methods, showing that our model is best in class in all selected quality measures.

### I.10.1 Future work topics

We suggest the following future work topics:

#### I.10.1.1 Complexity

While NP-hardness of  $\mathcal{HC}_{opt,\varepsilon}^{<\mathcal{CL}}$  follows from Theorem I.3.5, the complexity classes of order preserving hierarchical agglomerative clustering for  $\mathcal{SL}$  and  $\mathcal{AL}$  remain to be established.

#### I.10.1.2 Order versus dissimilarity

Since the order relation is treated as a binary constraint it has a significant effect on the output from the clustering process, and may in some cases lead to undesirable outcomes. For example, if the dissimilarity measure associates “wrong” elements for clustering, the induced order relation may exclude future merges of elements correctly belonging together by erroneously identifying them as comparable. Also, if elements are wrongly identified as comparable to begin with, they can never be merged. Both due to Theorem I.4.5.

Together, these observations indicate that “loosening up” the stringent nature of the order relation may be beneficial in applications where order preservation is not a strict requirement.

#### I.10.1.3 Other models for clustering

While we have chosen to develop our theory based on classical hierarchical clustering, it is likely that the theory we have presented can be extended or adjusted to apply to generalisations of hierarchical clustering too. As an example, we mention overlapping clusters and hierarchies of overlapping clusters [24]. While hierarchies of overlapping clusters lead to DAGs, rather than trees, it still seems likely that a completion along the lines of Section I.5 can be applied to the *partial DAGs* that necessarily arise when this type of hierarchical clustering is applied to strict partial orders in an order preserving fashion.

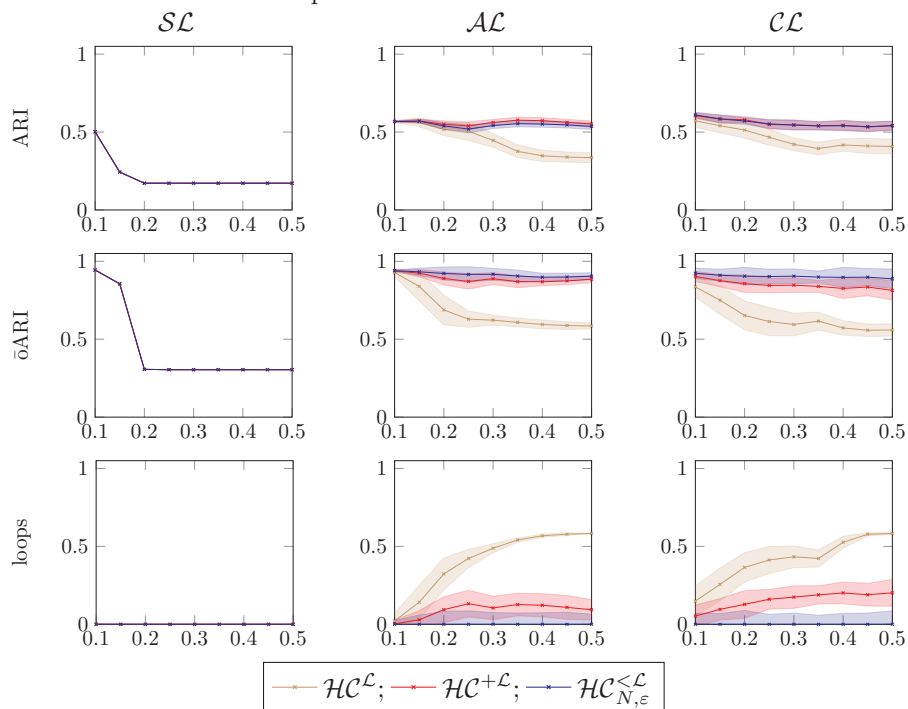
Also, as mentioned in the introduction, the framework we have presented can be applied directly to enable a theory for hierarchical agglomerative clustering in the presence of must-link and no-link constraints: The no-link constraints give rise to partial dendrograms that are easily evaluated via ultrametric completion.

**Acknowledgements.** I wish to thank each of the anonymous reviewers at Machine Learning for constructive feedback and comments greatly improving the exposition. I also owe a great thank you to Henrik Forsell, Department of Informatics (University of Oslo), and Gudmund Hermansen, Department of Mathematics (University of Oslo), for their comments, questions and discussions leading up to this work.

## I.A Plots from the part database demo

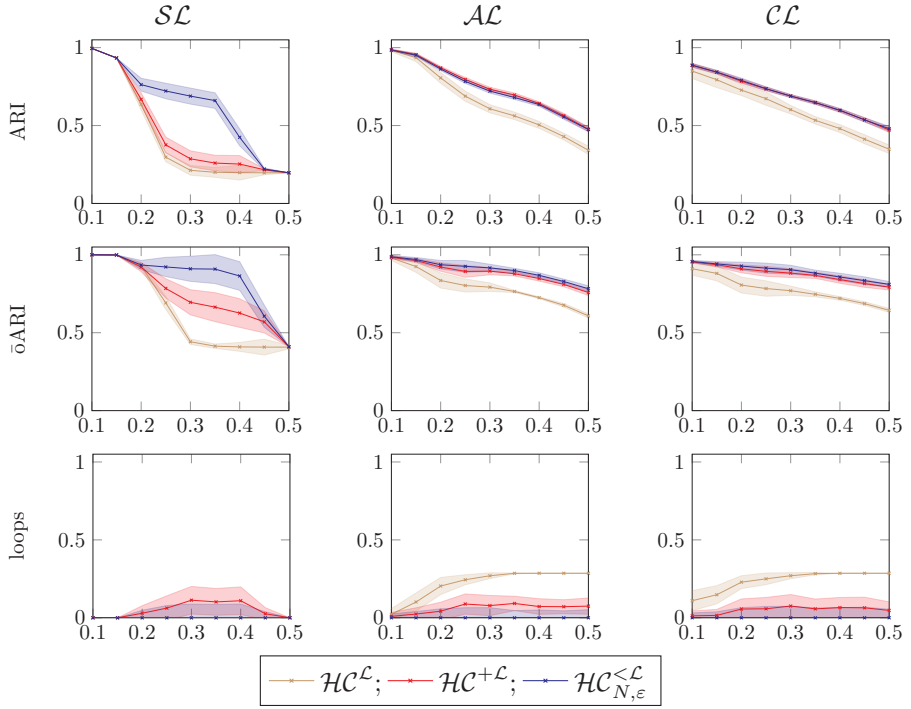
This section lists all the plots from the experiments described in Section I.9. The plots are grouped per connected component, and present results for all clustering methods, quality measures and linkage models. Please see Table I.2 for a list of statistical properties of the different connected components, and Table I.3 for the parameter settings used during the experiments.

Results for connected component no. 0.

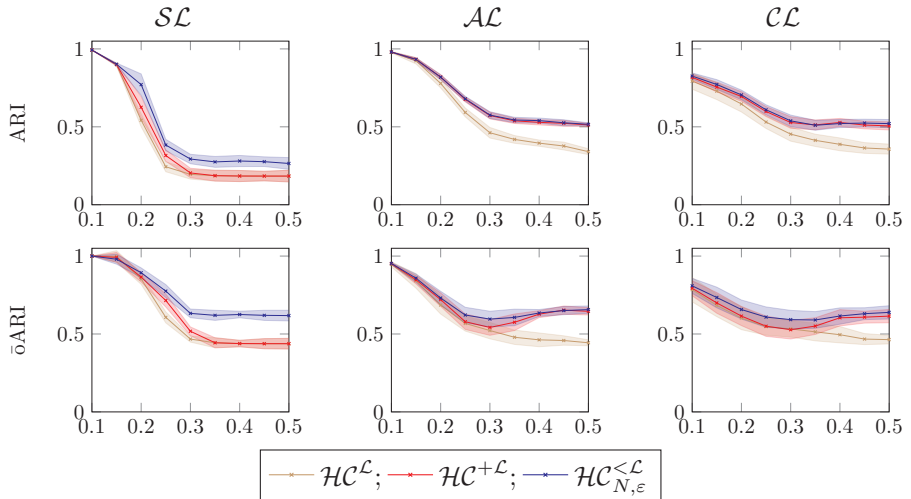


# I. Order preserving hierarchical agglomerative clustering

Results for connected component no. 1.

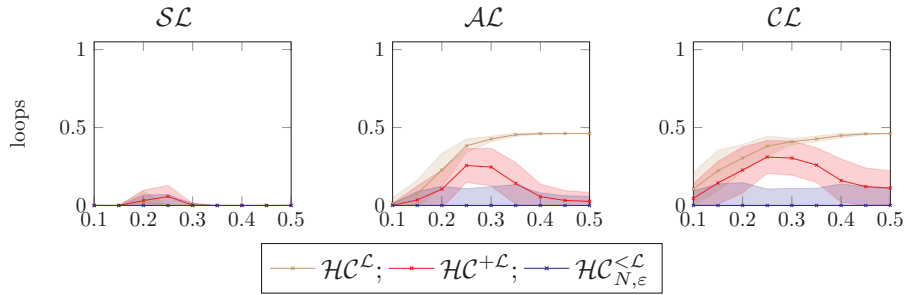


Results for connected component no. 2.

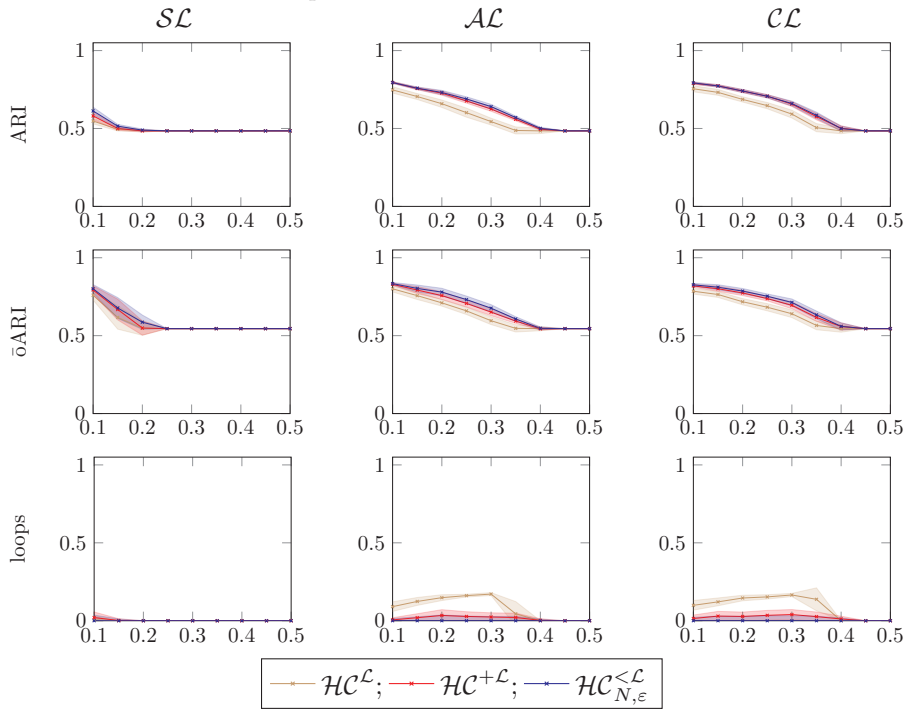


continued on next page...

Continued: results for connected component no. 2.

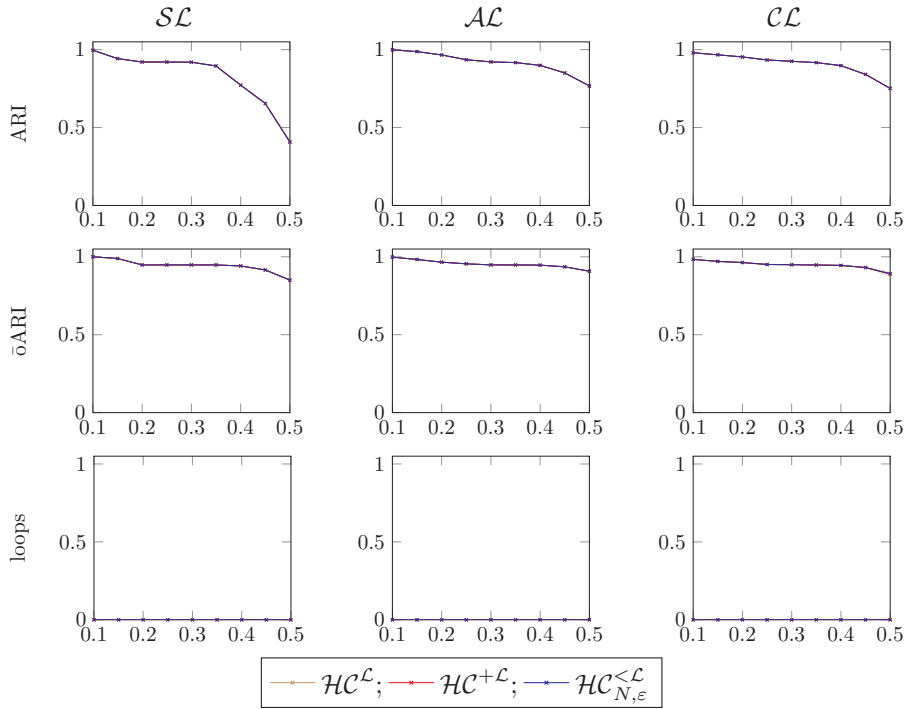


Results for connected component no. 3.

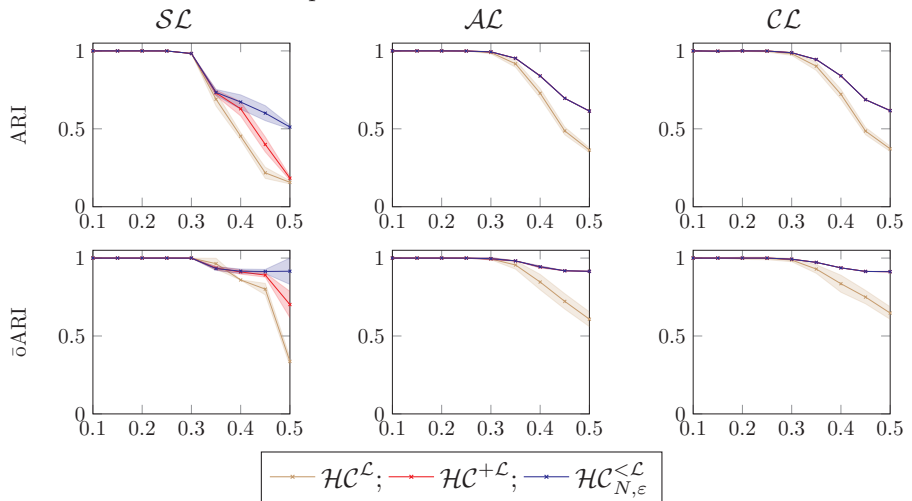


# I. Order preserving hierarchical agglomerative clustering

Results for connected component no. 4.

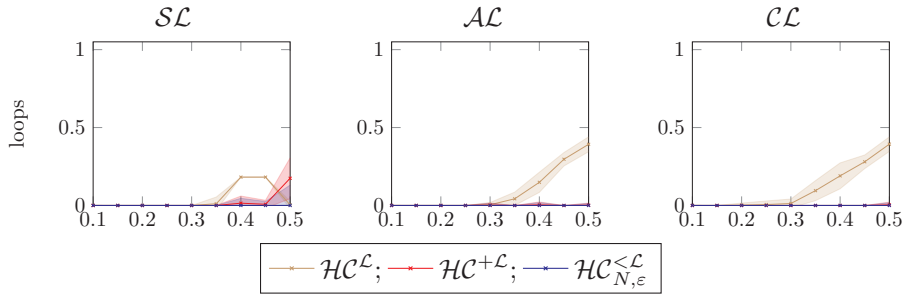


Results for connected component no. 5.

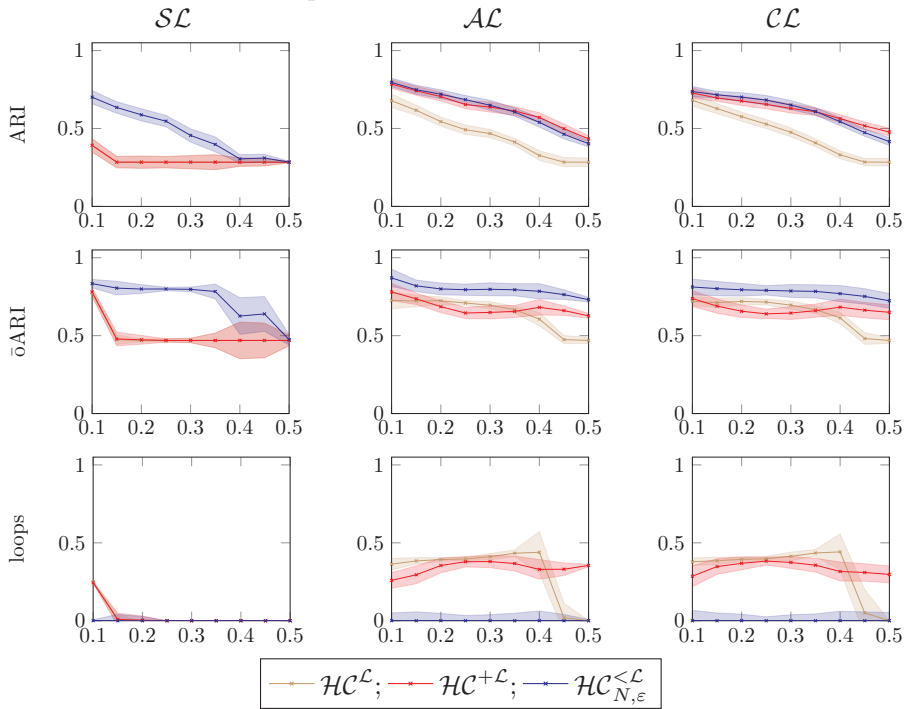


continued on next page...

Continued: results for connected component no. 5.

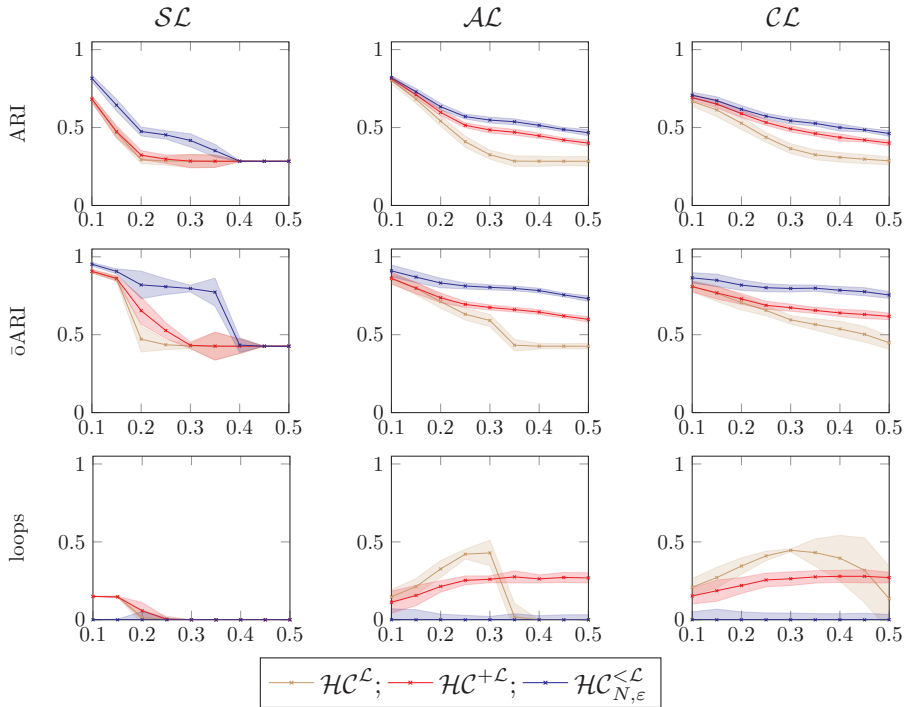


Results for connected component no. 6.



## I. Order preserving hierarchical agglomerative clustering

Results for connected component no. 7.



## I.B Reference implementation

The implementation used for the experiments in Sections I.8 and I.9 is available as open source at <https://bitbucket.org/Bakkelund/ophac>.

## References

- [1] Ackerman, M. and Ben-David, S. ‘A Characterization of Linkage-Based Hierarchical Clustering’. In: *Journal of Machine Learning Research* vol. 17, no. 231 (2016), pp. 1–17. URL: <http://jmlr.org/papers/v17/11-198.html>.
- [2] Basu, S., Davidson, I. and Wagstaff, K. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. 1st ed. Chapman & Hall/CRC, 2008.
- [3] Blyth, T. *Lattices and Ordered Algebraic Structures*. Universitext. Springer London, 2005. URL: <https://www.springer.com/gp/book/9781852339050>.
- [4] Bollobás, B. *Random Graphs*. 2nd ed. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2001.



- 
- [5] Buluç, A. et al. ‘Recent Advances in Graph Partitioning’. In: *Algorithm Engineering: Selected Results and Surveys*. Cham: Springer International Publishing, 2016, pp. 117–158. URL: [https://link.springer.com/chapter/10.1007/978-3-319-49487-6\\_4](https://link.springer.com/chapter/10.1007/978-3-319-49487-6_4).
- [6] Carlsson, G. and Mémoli, F. ‘Characterization, Stability and Convergence of Hierarchical Clustering Methods’. In: *J. Mach. Learn. Res.* vol. 11 (Aug. 2010), pp. 1425–1470. URL: <http://www.jmlr.org/papers/v11/carlsson10a.html>.
- [7] Carlsson, G. et al. ‘Hierarchical Quasi-Clustering Methods for Asymmetric Networks’. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Xing, E. P. and Jebara, T. Vol. 32. Proceedings of Machine Learning Research. Beijing, China: PMLR, 22–24 Jun 2014, pp. 352–360. URL: <http://proceedings.mlr.press/v32/carlsson14.html>.
- [8] Chierchia, G. and Perret, B. ‘Ultrametric Fitting by Gradient Descent’. In: *Advances in Neural Information Processing Systems*. Ed. by Wallach, H. et al. Vol. 32. Curran Associates, Inc., 2019, pp. 3181–3192. URL: <https://proceedings.neurips.cc/paper/2019/file/b865367fc4c0845c0682bd466e6ebf4c-Paper.pdf>.
- [9] Dasgupta, S. ‘A Cost Function for Similarity-Based Hierarchical Clustering’. In: *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*. STOC ’16. Cambridge, MA, USA: Association for Computing Machinery, 2016, pp. 118–127. URL: <https://dl.acm.org/doi/10.1145/2897518.2897527>.
- [10] Davidson, I. and Ravi, S. S. ‘Agglomerative Hierarchical Clustering with Constraints: Theoretical and Empirical Results’. In: *Knowledge Discovery in Databases: PKDD 2005*. Ed. by Jorge, A. M. et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 59–70.
- [11] De Soete, G., Carroll, J. D. and DeSarbo, W. S. ‘Least squares algorithms for constructing constrained ultrametric and additive tree representations of symmetric proximity data’. In: *Journal of Classification* vol. 4, no. 2 (1987), pp. 155–173. URL: <https://link.springer.com/article/10.1007/BF01896984>.
- [12] Fraley, C. and Raftery, A. E. ‘Model-Based Clustering, Discriminant Analysis, and Density Estimation’. In: *Journal of the American Statistical Association* vol. 97, no. 458 (2002), pp. 611–631. eprint: <https://doi.org/10.1198/016214502760047131>. URL: <https://doi.org/10.1198/016214502760047131>.
- [13] Gates, A. J. and Ahn, Y.-Y. ‘The Impact of Random Models on Clustering Similarity’. In: *Journal of Machine Learning Research* vol. 18, no. 87 (2017), pp. 1–28. URL: <http://jmlr.org/papers/v18/17-039.html>.

- [14] Ghoshdastidar, D., Perrot, M. and Luxburg, U. von. ‘Foundations of Comparison-Based Hierarchical Clustering’. In: *Advances in Neural Information Processing Systems 32*. Ed. by Wallach, H. et al. Curran Associates, Inc., 2019, pp. 7456–7466. URL: <http://papers.nips.cc/paper/8964-foundations-of-comparison-based-hierarchical-clustering.pdf>.
- [15] Gilpin, S., Nijssen, S. and Davidson, I. ‘Formalizing Hierarchical Clustering as Integer Linear Programming’. In: *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI’13. Bellevue, Washington: AAAI Press, 2013, pp. 372–378. URL: <https://www.aaai.org/ocs/index.php/AAAI/AAAI13/paper/view/6440>.
- [16] Herrmann, J. et al. ‘Acyclic Partitioning of Large Directed Acyclic Graphs’. In: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. May 2017, pp. 371–380. URL: <https://hal.inria.fr/hal-01744603>.
- [17] Hoffman, M., Steinley, D. and Brusco, M. J. ‘A note on using the adjusted Rand index for link prediction in networks’. In: *Social Networks* vol. 42 (2015), pp. 72–79. URL: <http://www.sciencedirect.com/science/article/pii/S0378873315000210>.
- [18] Holly, J. E. ‘Pictures of Ultrametric Spaces, the p-Adic Numbers, and Valued Fields’. In: *The American Mathematical Monthly* vol. 108, no. 8 (2001), pp. 721–728. URL: <http://www.jstor.org/stable/2695615>.
- [19] Hubert, L. and Arabie, P. ‘Comparing partitions’. In: *Journal of Classification* (1985), pp. 193–218.
- [20] Hughes, B. ‘Trees and ultrametric spaces: a categorical equivalence’. In: *Advances in Mathematics* vol. 189, no. 1 (2004), pp. 148–191. URL: <https://doi.org/10.1016/j.aim.2003.11.008>.
- [21] Jain, A. K. and Dubes, R. C. *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.
- [22] Janowitz, M. F. *Ordinal and Relational Clustering*. WORLD SCIENTIFIC, 2010. eprint: <https://www.worldscientific.com/doi/pdf/10.1142/7449>. URL: <https://www.worldscientific.com/doi/abs/10.1142/7449>.
- [23] Jardine, N. and Sibson, R. *Mathematical Taxonomy*. Wiley series in probability and mathematical statistics. Wiley, 1971.
- [24] Jeantet, I., Miklos, Z. and Gross-Amblard, D. ‘Overlapping Hierarchical Clustering (OHC)’. In: *Intelligent Data Analysis (IDA 2020)*. 2020. URL: <https://hal.inria.fr/hal-02452729/>.
- [25] Johnson, S. C. ‘Hierarchical clustering schemes’. In: *Psychometrika* vol. 32, no. 3 (1967), pp. 241–254. URL: <https://link.springer.com/article/10.1007/BF02289588>.
- [26] Kamishima, T. and Fujiki, J. ‘Clustering Orders’. In: *Discovery Science*. Ed. by Grieser, G., Tanaka, Y. and Yamamoto, A. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 194–207. URL: [https://doi.org/10.1007/978-3-540-39644-4\\_17](https://doi.org/10.1007/978-3-540-39644-4_17).

- [27] Karp, R. M. ‘Reducibility among Combinatorial Problems’. In: *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*. Boston, MA: Springer US, 1972, pp. 85–103. URL: [https://doi.org/10.1007/978-1-4684-2001-2\\_9](https://doi.org/10.1007/978-1-4684-2001-2_9).
- [28] Kriegel, H.-P. et al. ‘Density-based clustering’. In: *WIREs Data Mining and Knowledge Discovery* vol. 1, no. 3 (2011), pp. 231–240. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.30>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.30>.
- [29] Lassila, O. and Swick, R. R. *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation. W3C, Feb. 1999. URL: <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/>.
- [30] Li, X. et al. ‘Semi-Supervised Clustering in Attributed Heterogeneous Information Networks’. In: *Proceedings of the 26th International Conference on World Wide Web*. WWW ’17. Perth, Australia: International World Wide Web Conferences Steering Committee, 2017, pp. 1621–1629. URL: <https://doi.org/10.1145/3038912.3052576>.
- [31] Łuczak, M. ‘Hierarchical clustering of time series data with parametric derivative dynamic time warping’. In: *Expert Systems with Applications* vol. 62 (2016), pp. 116–130. URL: <http://www.sciencedirect.com/science/article/pii/S0957417416302937>.
- [32] Macqueen, J. ‘Some methods for classification and analysis of multivariate observations’. In: *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*. 1967, pp. 281–297. URL: <https://projecteuclid.org/euclid.bsmmsp/1200512992>.
- [33] Markov, I. L., Hu, J. and Kim, M. ‘Progress and Challenges in VLSI Placement Research’. In: *Proceedings of the IEEE* vol. 103, no. 11 (Nov. 2015), pp. 1985–2003. URL: <https://ieeexplore.ieee.org/document/7295553>.
- [34] Pio, G. et al. ‘Multi-type clustering and classification from heterogeneous networks’. In: *Information Sciences* vol. 425 (2018), pp. 107–126. URL: <https://www.sciencedirect.com/science/article/pii/S0020025516321570>.
- [35] Rammal, R., Toulouse, G. and Virasoro, M. A. ‘Ultrametricity for physicists’. In: *Rev. Mod. Phys.* vol. 58 (3 July 1986), pp. 765–788. URL: <https://link.aps.org/doi/10.1103/RevModPhys.58.765>.
- [36] Rand, W. M. ‘Objective Criteria for the Evaluation of Clustering Methods’. In: *Journal of the American Statistical Association* vol. 66, no. 336 (1971), pp. 846–850. URL: <http://www.jstor.org/stable/2284239>.
- [37] Sneath, P. and Sokal, R. *Numerical Taxonomy: The Principles and Practice of Numerical Classification*. A Series of books in biology. W. H. Freeman, 1973.

- [38] Vinh, N. X., Epps, J. and Bailey, J. ‘Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance’. In: *Journal of Machine Learning Research* vol. 11, no. 95 (2010), pp. 2837–2854. URL: <http://jmlr.org/papers/v11/vinh10a.html>.
- [39] Ward, J. H. ‘Hierarchical Grouping to Optimize an Objective Function’. In: *Journal of the American Statistical Association* vol. 58, no. 301 (1963), pp. 236–244. URL: <http://www.jstor.org/stable/2282967>.
- [40] Warrens, M. J. ‘On the Equivalence of Cohen’s Kappa and the Hubert-Arabie Adjusted Rand Index’. In: *Journal of Classification* (2008), pp. 177–183.



## Paper III

# Machine part data with part-of relations and part dissimilarities for planted partition generation

**Daniel Bakkelund**

Published in *Data in Brief*, Mar 2022,  
DOI: 10.1016/j.dib.2022.108065.

### Abstract

Identifying relationships between entities in data is a central topic across various industries and businesses, from social networks to supply chain and heavy manufacturing industries. In this paper we present data from a database of machinery represented in terms of machine parts. The machine parts are originally organised in tree structures where the vertices are machine part types, and the edges are “part-of” relations. Hence, each tree represents a type of machinery broken down into its machine part constituent types. The data we present is the union over these trees, making up a directed acyclic graph describing the type hierarchy of the machine parts.

The motivation for publishing the dataset is the following real-world industry problem: Each tree represents a mechanical design, and over time some designs have been copy-pasted with minor modifications. The new instances have been given new identifiers with no reference to where from they were copied. In hindsight, it is desirable to recover the copy-paste links to for interchange between essentially identical designs. However, telling which parts are copies of which other parts has turned out to be difficult. In particular, the metadata has a tendency of displaying higher similarities within a composite part than between a part and its copy. Due to non-disclosure, we cannot provide the metadata, but we provide element wise dissimilarities that are generated based on the metadata using classical methods such as Jaccard similarity on description texts, material types etc. The dissimilarities are obtained from a data science project in the company owning the data, trying to tackle the very problem of recovering the copy-paste links.

Availability of labeled data on this data set is limited, so based on our in-depth knowledge of the problem domain, we present a data synthetisation method that can generate arbitrarily large problem instances of the copy-paste problem based on the sample data, that provides a realistic



### III. Machine part data with part-of relations for planted partition generation

---

representation of the real world problem. The problems are presented as planted partitions of vertices of directed acyclic graphs with vertex dissimilarities, and thus constitutes a typical classification problem along the lines of graph- or network clustering.

The type of industry data we present is usually company confidential, bound by intellectual property rights, and generally not available to scientists. We therefore publish this anonymised dataset to offer real world sample data and generated problem instances for researchers that are interested in this type of classification problems, and on which theories and algorithms can be tested.

The data and the problem generation methodology are backed by a Python implementation, providing both data access and an API for parameterised problem generation. The data is also available as raw files.

**Keywords:** Machine parts, part-of relations, dissimilarity,planted partition, clustering, link recovery

#### III.1 Data specification

<b>Subject</b>	Data Science
<b>Specific subject area</b>	Applied Machine Learning
<b>Type of data</b>	Table of machine parts types and part-of relations Table of dissimilarities between machine parts types Python code
<b>How the data were acquired</b>	Extraction from company database.
<b>Data format</b>	Analysed, Raw
<b>Description of data collection</b>	<b>Machine part types:</b> The machine part data is extracted from a relational database. The initial raw data is organised as trees of machine parts, where each node has a unique identifier and a type-id. In the data collection process, the trees have been replaced by a graph of types as follows. The vertices of the graph corresponds to the set of types of the tree vertices. Then, edges are added to the graph if there is an edge in a tree between vertices of corresponding types. The resulting graph thus represents the <i>type part-of structure</i> defined by the trees. Since one type of machinery may be a part of different types of high level machinery, in the way a type of tire may be a part of many types of cars, the type hierarchy becomes a graph, rather than a tree. And moreover, since a part cannot another part of the same type [7], the type hierarchy is a directed acyclic graph. <sup>1</sup>

---

<sup>1</sup>If a part contains another part of the same type as a sub-part, this leads to a cycle of containment. Since this leads to an infinitely deep structure, it is not achievable for practical manufacturing.

	<p>Our particular subset of machine part types was chosen as follows. When we generated the above graph for all machine parts, we found that the graph had one very large connected component and a large set of disconnected vertices, but also eight connected component in the range of 11 to 40 vertices. Since each of these connected component closely correspond to single designs, we chose these eight connected components as our sample dataset.</p> <p><b>Dissimilarities:</b> The dissimilarity data is obtained from an internal project in the company owning the data trying to tackle the very problem of recovering the mentioned copy-paste links. The dissimilarities are generated based on metadata about the machine part types, such as description texts, material types, weights etc.</p>
<b>Data source location</b>	Proprietary database owned by TechnipFMC <sup>2</sup> , a privately held company in the Oil & Gas sector.
<b>Data accessibility</b>	Repository name: Mendeley Data Data identification number: 10.17632/dhhxzdzm3v.1 Direct URL to data: <a href="https://data.mendeley.com/v1/datasets/dhhxzdzm3v/">https://data.mendeley.com/v1/datasets/dhhxzdzm3v/</a>
<b>Related research article</b>	<i>Daniel Bakkelund, Order Preserving Agglomerative Hierarchical Clustering, Mach Learn. (2021).</i> <a href="https://doi.org/10.1007/s10994-021-06125-0">https://doi.org/10.1007/s10994-021-06125-0</a> .

### III.2 Value of the Data

- The type of data we present is usually company confidential, and therefore very difficult to come by for researchers. By publishing a small subset of the data together with code that can proliferate the data based on our understanding of the problem domain, we hope to allow other researchers to test their hypotheses and methods on close to real world data.

In this respect, we particularly mention the problem of *order preserving clustering* [1]. This is a field in development where there are currently no public datasets available for benchmarking and/or testing of methods and hypotheses. We therefore wish to publish this dataset, and the model for generating classification problems from this dataset, to support further development of this new branch of classification research.

- Since we present dissimilarity data with additional relations, the main audience is likely to be researchers and practitioners within classification and clustering that work with data that has additional structure. As a non-exhaustive list of examples we mention graph- and network clustering [5], order preserving clustering [3], acyclic partitioning [4] and clustering with constraints [2].

<sup>2</sup><https://www.technipfmc.com/>



- One of the contributions of this paper is a model for generation of planted partitions simulating the copy-paste problem. The model is based on our in-depth knowledge about the problem domain, and we believe that this model, together with the published data, provides realistic representations of the previously described copy-paste problem. Hence, models and algorithms that perform well on these planted partitions can be expected to perform well also on the real dataset.
- As for the mentioned industry problem, this is an *excess inventory problem* in that the machine part manufacturer has an increasing amount of machinery in stock. A traditional approach to excess inventory is that of *excess inventory disposal* [8] to free up capacity. However, this is easily sub-optimal for expensive machinery, both with respect to economy, and also with respect to the environment, as manufacturing of complex steel based machinery has a large CO<sub>2</sub> footprint. Rather, TechnipFMC states that if they can match similar machinery in the described fashion, then this will lead to increased sales from inventory, rather than producing new machinery. Thus, yielding a double up-side compared to decimating the machinery in stock.

### III.3 Data Description

This section describes the format of the flat files containing the machine part data and the dissimilarity data. Note, however, that the data is also available through the provided python API.

The data is available as two CSV<sup>3</sup> files, one file for the machine part structures, and one for the machine part dissimilarities.

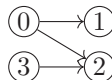
#### III.3.1 The machine part file

The machine part file is named `parts.csv`. Each line in the machine part file is formatted as

$$\langle id \rangle (, \langle child id \rangle)^*$$

That is, each line is a comma separated list of integers. The first integer is the part type identifier (*id*), and the remaining integers (if any) are the part types that occur as “part-of” type *id*.

For example, if the data was constituted by the graph



where the arrows indicate that 1 and 2 are both *part-of* 0 and that 2 is *part-of* 3, then the corresponding file would look like

---

<sup>3</sup>[https://en.wikipedia.org/wiki/Comma-separated\\_values](https://en.wikipedia.org/wiki/Comma-separated_values)

```
0,1,2
1
2
3,2
```

### III.3.2 The dissimilarities file

The dissimilarities are also organised in a CSV file, `dissimilarities.csv`, with each line on the format

$$\langle \text{integer}:a \rangle, \langle \text{integer}:b \rangle, \langle \text{decimal number}:d \rangle$$

Here, the integers are part type ids, as given in `parts.csv`, and the dissimilarity  $d$  is the dissimilarity between the specified types. The indices are always ordered so that  $a < b$ , and for the dissimilarities, we always have  $0 \leq d \leq 1$ .

Given the above example part type structure, a corresponding dissimilarity file would be on the form

```
0,1,0.2021
0,2,0.3141
0,3,0.2718
1,2,0.1414
1,3,0.7071
2,3,0.2600
```

That is, the dissimilarity between 0 and 1 is 0.2021, meaning 0 and 1 are more dissimilar than, say, 1 and 2 that has a dissimilarity of 0.1414.

### III.3.3 Some statistics on the connected components

The part data constitutes eight connected components, where each connected component is a small DAG. Table III.2 lists some typical graph statistics for the connected components.

### III.3.4 Parent-child dissimilarities

A feature of the dissimilarity data, is that there is a high probability for the dissimilarity of a part and a sub-part to be low. This is due to a significant overlap in metadata, stemming from the fact that a part and its sub-parts are often closely related in several ways. For example, machinery wrought out of steel will often have parts with similar material- and mechanical properties. For machinery that will be used under harsh environmental conditions, the environmental characteristics of the parts must necessarily be very similar. Description texts describing a sub-part will often contain references to the containing part, and so on. Deducing the dissimilarities based on this metadata

### III. Machine part data with part-of relations for planted partition generation

cc no.	cc size	in/out deg.	$p$
0	12	0.92	0.17
1	14	0.93	0.14
2	13	0.92	0.15
3	40	1.27	0.07
4	20	1.35	0.14
5	11	1.18	0.24
6	20	1.10	0.12
7	20	0.95	0.10

Table III.2: Some key characteristics of the connected components of the machine parts dataset:

- cc no. – the index of the connected component
- cc size – the number of vertices in the connected component
- in/out deg – the directed average degree of the connected component
- $p$  – the probability that for a pair of random vertices  $a$  and  $b$ , the edge  $(a, b)$  exists in the transitive reduction

*Note: The table is an adaptation of [1, Table 2].*

therefore sometimes lead to low dissimilarity between parent and child. For the copy paste problem, this is a complicating factor, since a part and a sub-part can never be copy-paste related. The dissimilarity distributions between parts and sub-parts are displayed in Figure III.1.

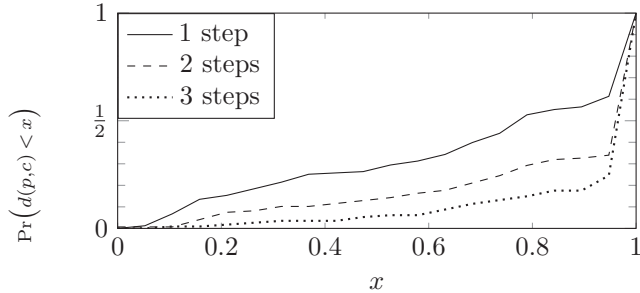


Figure III.1: Probability for a part and a sub-part to have dissimilarity no higher than  $x$ . The first axis value is the dissimilarity in the range  $[0, 1]$ , and the second axis is the probability of a parent part  $p$  and child part  $c$  to have a dissimilarity  $d(p, c)$  no higher than  $x$ ; that is,  $\Pr(d(p, c) < x)$ . The curves represent parent-child pairs that are separated by 1, 2 or 3 levels. We see that there is a higher probability for low dissimilarity between a part and a contained part if the containment is direct (1 step) compared to a nested containment ( $\geq 2$  steps).

### III.4 Experimental design, materials and methods

In this section, we describe the model for generation of planted partitions based on the published dataset. The model is a simplified representation of the copy-paste mechanism in the problem domain. An important goal of the model has been to keep it simple, while at the same time not underplaying the complexity of the copy-paste process and the following changes to the data.

It should be noted that our notion of a planted partition is not the same as the probabilistic concept of planted partitions sometimes encountered in clustering literature [6]. Rather, in the model we present, the generation of the planted partitions is based on our understanding of the copy-paste problem, and our wish to simulate this. We still choose to refer to this as *planted partitions*, since they are, in name, exactly that.

On a high level, the model works as follows. Given a connected component  $C$  from `parts.csv`, the dissimilarities from `dissimilarities.csv`, a positive integer  $n$ , a location parameter  $\mu$  and a scale parameter  $\sigma^2$ , we generate a planted partition with  $n + 1$  parallel instances through the following steps:

1. Make  $n$  copies of  $C$ , providing us with the connected components  $\{C_i\}_{i=0}^n$  where  $C = C_0$ . Denote the vertices of  $C$  by  $\{v_1^0, \dots, v_m^0\}$ , and similarly denote the vertices of  $C_i$  by  $\{v_1^i, \dots, v_m^i\}$  so that  $v_k^i$  is the copy in  $C_i$  of  $v_k^0$ .
2. For every connected component  $C_i$ , define the *intra component dissimilarities* as follows:

$$d(v_r^i, v_s^i) = d_0(v_r^0, v_s^0),$$

where  $d_0$  is the dissimilarity found in `dissimilarities.csv`. That is, the intra component dissimilarities in the copies are the same as in the original connected component.

3. Let  $Y \sim \mathcal{N}(\mu, \sigma^2)$  be a random variable where  $\mathcal{N}(\mu, \sigma^2)$  is the Gaussian distribution located at  $\mu$  with variance  $\sigma^2$ . We define the stochastic function  $\alpha : [0, 1] \rightarrow [0, 1]$  by  $\alpha(x) = x + Y$  through rejection sampling, naively continuing to draw from  $Y$  until  $x + Y \in [0, 1]$ . The *inter component dissimilarities* may now be defined as

$$d(v_r^i, v_s^j) = \alpha\left(d_0(v_r^0, v_s^0)\right).$$

That is, we distort the dissimilarity between the copy-paste instances by adding Gaussian noise.

The result is a set of machine parts with part-of relations that is the union of all the copies  $C_i$  equipped with dissimilarities. The corresponding planted partitions are the sets  $P_k = \{x_k^i\}_{i=0}^n$  for  $1 \leq k \leq m$ , defining the  $m$  sets of copy-paste elements.

An example is depicted in Figure III.2.

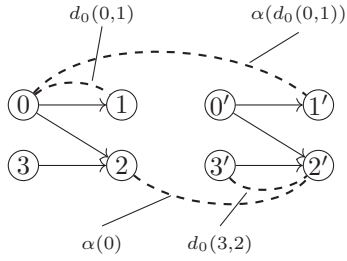


Figure III.2: The dashed lines indicate dissimilarity links. We can see that the dissimilarity between 2 and the copy 2' is  $\alpha(0)$ , the dissimilarity between the element 0 and the copy of the child 1' is  $\alpha(d_0(0,1))$ , which is the perturbed dissimilarity of  $d_0(0,1)$ . And finally, that the intra-component dissimilarity between 3' and 2' is identical to that in the original connected component, namely  $d_0(3,2)$ .

The model is subject to at least two simplifications that deviate from the real world case. In both cases, we chose to do this to keep the model simple. However, we also believe that this does not compromise the problem generation in terms of benchmarking relative the real world problem:

- *The topology of the original and the copy is identical.*  
We do not add or remove vertices or relations when we copy. In the real application, this happens to some extent.
- *The intra component dissimilarities are unchanged when copied.*  
Since metadata is changed after copying, the intra component dissimilarities will also change in the real application.

We summarise the input and output of the planted partition generation in Tables III.3 and III.4.

parameter	explanation
cc-ids	The ids of the connected components that shall be duplicated
n	The number of copies to make
$\mu$	The mean translation of the dissimilarities under $\alpha$
$\sigma^2$	The variance in the noise applied by $\alpha$

Table III.3: Table of inputs to the planted partition generation process.

Now, given a generated problem instance  $(X, E, d, \mathcal{P})$  and a classification procedure  $\mathfrak{C}$ , to which degree can  $\mathfrak{C}$  recover  $\mathcal{P}$  if given only  $X$ ,  $E$  and  $d$ ?

data	explanation
$X$	A set of vertices $X$ making up the union of the original connected components as well as all the copies
$E$	A set of edges $(a, b) \in X \times X$ denoting all the part-of relations of both the original connected components as well as the copies
$d$	A dissimilarity measure defined on all of $X$ generated according to the above procedure
$\mathcal{P} = \{P_i\}_{i=1}^{ X }$	The planted partitions; that is, the sets consisting of machine parts that are copies of each other.

Table III.4: Table of outputs from the planted partition generation process.

### III.4.1 Python implementation

An open source `python` implementation of the above model is made available. The library is most easily installed via `PyPi` by

```
python3 -m pip install machine-parts-pp [--user]
```

Notice that the library requires `python` version 3.0 or higher. For further documentation of the provided functionality, please visit <https://pypi.org/project/machine-parts-pp/>.

**Acknowledgments.** This work has been funded by the Department of Informatics (The Faculty of Mathematics and Natural Sciences, University of Oslo), the SIRIUS Centre for Scalable Data Access (Research Council of Norway, project no.: 237898) and TechnipFMC. We also wish to express our gratitude to TechnipFMC for sharing data with the scientific community, and to Derek Smith and Marcel Castro at TechnipFMC for providing invaluable support in the process of publishing the data.

### 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.
- The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

The author (Daniel Bakkelund) holds a position as data scientist at TechnipFMC.

### References

- [1] Bakkelund, D. ‘Order preserving hierarchical agglomerative clustering’. In: *Mach Learn* (2021). URL: <https://doi.org/10.1007/s10994-021-06125-0>.

### III. Machine part data with part-of relations for planted partition generation

---

- [2] Basu, S., Davidson, I. and Wagstaff, K. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. 1st ed. Chapman & Hall/CRC, 2008.
- [3] Ghoshdastidar, D., Perrot, M. and Luxburg, U. von. ‘Foundations of Comparison-Based Hierarchical Clustering’. In: *Advances in Neural Information Processing Systems 32*. Ed. by Wallach, H. et al. Curran Associates, Inc., 2019, pp. 7456–7466. URL: <http://papers.nips.cc/paper/8964-foundations-of-comparison-based-hierarchical-clustering.pdf>.
- [4] Herrmann, J. et al. ‘Acyclic Partitioning of Large Directed Acyclic Graphs’. In: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. May 2017, pp. 371–380. URL: <https://hal.inria.fr/hal-01744603>.
- [5] Malliaros, F. D. and Vazirgiannis, M. ‘Clustering and community detection in directed networks: A survey’. In: *Physics Reports* vol. 533, no. 4 (2013). Clustering and Community Detection in Directed Networks: A Survey, pp. 95–142. URL: <https://www.sciencedirect.com/science/article/pii/S0370157313002822>.
- [6] Mossel, E., Neeman, J. and Sly, A. ‘Reconstruction and estimation in the planted partition model’. In: *Probability Theory and Related Fields* vol. 162, no. 3 (2015), pp. 431–461.
- [7] Rescher, N. ‘Axioms for the part relation’. In: *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition* vol. 6, no. 1 (1955), pp. 8–11.
- [8] Rosenfield, D. B. ‘Disposal of excess inventory’. In: *Operations research* vol. 37, no. 3 (1989), pp. 404–409.

# Appendix A

## Correction of Theorem I.4.6

Theorem I.4.6 of Paper I has been identified to contain an error, but as the theorem appears in a published paper, the thesis text is not changed; rather, we supply a corrected version of the theorem here. Note that the faulty part of the theorem is never used, and has, thus, caused no harm.

The theorem is concerned with the classification of the strictly partially ordered sets  $(X, <)$  for which the set  $\mathfrak{P}(X, <)$  of regular partitions ordered by partition refinement has a greatest element. Theorem I.4.6 wrongly suggests that whenever the order relation is non-trivial, there is no greatest element of  $\mathfrak{P}(X, <)$ . In the below correction, we first present a class of strictly partially ordered sets that we recognise as architectures of feed-forward neural networks. We then proceed to prove that this is exactly the class of strictly partially ordered sets where  $\mathfrak{P}(X, <)$  has a greatest element.

**Definition A.0.1.** An  $n$ -layered feed-forward network, ( $n$ -ffn) is a directed acyclic graph  $G = (V, E)$  for which there exists a partition  $\{V_i\}_{i=1}^n$  of the vertices so that

$$E = \bigcup_{i=1}^{n-1} V_i \times V_{i+1}.$$

**Example A.0.2.** An example of  $n$ -layered feed-forward networks is exactly the class of feed-forward neural network architectures where the neurons are arranged in  $n$  layers, and every neuron in one layer is connected to every neuron in the next layer, as displayed in Figure A.1.

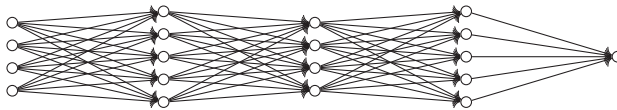


Figure A.1: A 5-layered feed-forward network.

**Definition A.0.3.** A strictly partially ordered set  $(X, <)$  is said to be of **ffn-type** if the transitive reduction of  $(X, <)$  is an  $n$ -ffn for some natural number  $n$ .

**Theorem A.0.4.** Let  $(X, <)$  be a strictly partially ordered set. Then the family of regular partitions  $\mathfrak{P}(X, <)$  over  $(X, <)$  ordered under partition refinement has a greatest element if and only if  $(X, <)$  is of ffn-type.

*Proof.* Assume first that  $(X, <)$  is of ffn-type, and let  $\{X_i\}_{i=1}^n$  be the corresponding partition given by Definition A.0.1. Clearly,  $\{X_i\}_{i=1}^n$  is a regular



partition with respect to  $<$ . Hence, since  $(\{X_i\}_{i=1}^n, <')$  is a linearly ordered set, and since, by construction, the  $X_i$  are the only maximal antichains of  $(X, <)$ , it follows that every other regular partition is a refinement of  $(\{X_i\}_{i=1}^n, <')$ . Thus,  $(\{X_i\}_{i=1}^n, <')$  is the greatest element.

Now assume that  $(\{X_i\}_{i=1}^n, <')$  is the greatest element of  $\mathfrak{P}(X, <)$ . Then each  $X_i$  is an antichain, and  $(\{X_i\}_{i=1}^n, <')$  is a linearly ordered set. We can, thus, without loss of generality, assume that the enumeration of the  $X_i$  are compatible with the induced relation, so that  $i < j \Rightarrow X_i <' X_j$ . Furthermore, for  $1 \leq i < j \leq n$ , we must have

$$(x, y) \in X_i \times X_j \Rightarrow x < y, \tag{A.1}$$

for otherwise  $\{x, y\}$  would be an antichain not contained in any of the  $X_i$ , contradicting that  $(\{X_i\}_{i=1}^n, <')$  is the greatest element of  $\mathfrak{P}(X, <)$ . In particular, (A.1) must hold for  $j = i + 1$ , which implies that the transitive reduction of  $(X, <)$  is an  $n$ -ffn  $(X, E)$  with  $E = \cup_{1 \leq i \leq n-1} X_i \times X_{i+1}$ . ■