Universidade Nova de Lisboa
OMNIS CIVITAS CONTRA SE DIVISA NON STABIT
Faculdade de Ciências e Tecnologia

PhD Report: 2010/2011-1

# Segmenting personal memories

by

Nuno Miguel Soares Datia

Supervisor: João Moura-Pires

February 2011

# Contents

# List of Figures

# 1 Introduction

The primary function of photography is to capture our family life, and to bind us to our own pasts and to the pasts of our social groups [10]. Despite the purpose of a personal collection of photos, it's undeniable that technological evolution enabled a change of habits. Some years ago it was necessary to have a camera with us to take a photo. In those times, most of the photos depict some important events in family life. Nowadays, almost everyone carries digital devices with video and photo capabilities. For example, cell phones have been transformed beyond their primarily function, into small media devices, with increasing photo and video quality. According to a Kodak study [18], the worldwide penetration of such devices is nowadays larger than 90%. Those devices democratise the production of media, but, more important, they transform the action of taking a picture into a habit. Today, personal photo collections include much more photos related to the day life. We are in the pervasive imaging era.

However, photos are difficult to interpret outside the context where they were taken. In fact, the meaning of each photo is closely related with the purpose of the shot. Kindberg et al. [14], defines a taxonomy of reasons for image capture. There are two main categories, *Affective* and *Functional*, each one subdivided into reasons of *social* or *individual* consideration. Thus, it is necessary to complement the photo with information about the context, that, to some extend, only the photographer knows. Emotions and non-visual contextual information can be very important to complete the context. The insertion of metadata in photos can span several moments:

- The moment the photographer take a shot
- In the catalogue process
- During image retrieval

.

When we talk about the context of a photo, we generally mean four important axis: "who", "where", "when" and "what". They describe something ("what") that happens at a given time ("when") and place ("where"), generally with people involved ("who"). However, the "who" is related with the entity depicted, that can be a person or, for instance, a monument.

Some of the information needed to complete the context is impossible, or at least very hard, to automate. First, there are some concepts that have few, or none, visual clues; second, there is no direct connection between the high-level concepts needed for the context and the low-level features present in the photos [13, 22]. Thus, the user should contribute with hand made annotations to support a proper context recall when it is necessary (e.g. during retrieval of relevant images). The first time users have the opportunity to insert new metadata is during the catalogue process. However, annotating each photo is not an option, due its time-consuming nature. This pose an interesting challenge to researchers on how to motivate users to do it [11, 15, 25]. Knowing beforehand that some photos share a similar context, they can be group, enabling the insertion of metadata in batches. This reduces the manual labour [6, 26].

## 1.1 Events and rolls

Some of the photos we take are just fragments of trivial moments of everyday life, whose importance, as a document of our memories, is dubious. Those shots are impulses to capture the moment because we have, for instance, a cell phone in our back pocket. However, taking a photo is still, most of the time, an action with a purpose, and a shot generally takes part of a set that is used to document an event. Those are the ones we want to share among friends and family.
Despite the context of a photo could be described by the four axis mentioned earlier, we will focus our attention to the time axis, owing to its importance. An event occur in a specific time period, with fuzzy limits, since it can span several minutes, hours, or even days. In fact, events em-

body the seasonality that is present in our life. They are controlled by the natural cycles of days and years, but also by the rhythm of the weekly cycle, with more social meaning than the formers [29]. Notice that time is one of the most important dimensions of the social world [28].

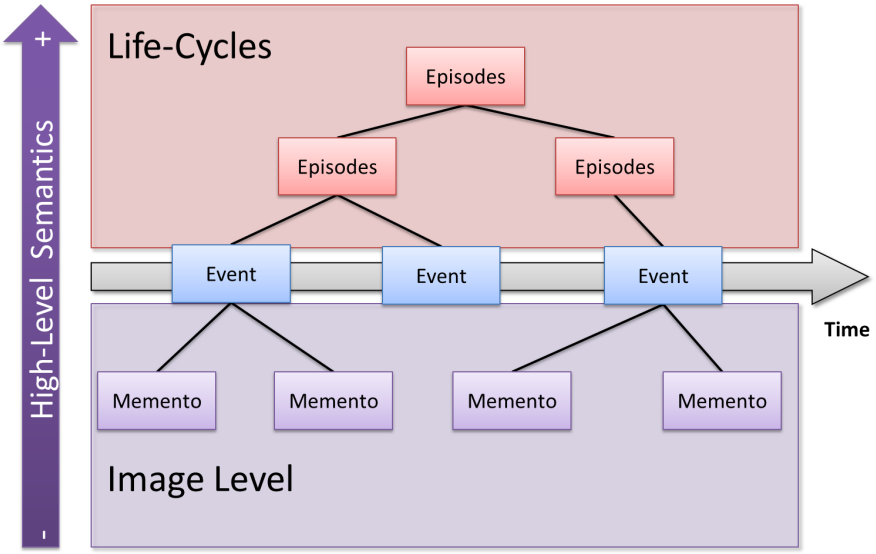


**Figure 1:** *Overview of a possible event structure.*

Despite the events follow a timeline, their representation do not need to be flat. In fact, they can be organised hierarchically, as illustrated by figure 1. This representation does not try to promote an alternative way to visualise the events - it just exposes some of the underlying cycles that govern our lives. We call **base events** to those represented in the timeline, which are derived from the photos taken during a day cycle. Usually, one day accommodate several events. Those events are sometimes part of a more broad activity that follows another cycle, generally the week. One example of such an activity is the summer vacations. The higher levels of the hierarchy represent those activities, and are named **episodes**. Since the number of cycles are limited to day, week, month and year, the number of upper levels is small. The semantics associated with such episodes is more difficult to infer from the context that is extracted from the photos, includ-

ing the image itself. On the bottom of the hierarchical representation, we have a division of the events, into fine grain groups that share similar image content, that we will refer to as **mementos**. The refinement uses the available features present in the metadata or stemmed from the image content itself, but keeps time locality. The arrangement into fine grain groups resemble the notion of photo stacks, available is some professional photo management software [1, 3]. Again, the deepness of this hierarchy is small.

When users take photos, they leave in the support medium a set that spans different periods in time and depicts several social and personal moments. Sometimes this set of assorted photos $\langle p_1, ..., p_n \rangle$ is referred to as a **roll**. Despite their variety, during the catalogue, rolls can be divided in several subsets, where each represents a different event. The division is made, either manually by the user, or automatically based on the metadata of each photo. In fact, we can look to the photos tuples $\langle t, g, i, ... \rangle$, where each element has a specific type of information inserted by the camera in the moment of the shot. As an example, we have the creation time $t$ of each photo, the GPS coordinates $g$ of the place where it was taken, the ISO level $i$ used during the shot, just to name a few.
Despite the variety of metadata available, the segmentation of a collection of photos is, at its core, a time segmentation problem. The minimum requirement to divide a roll is that each photo is timestamp with its creation time. Later, other types of metadata, namely, GPS positioning, can be used to fine-tune the boundaries of each event.

Knowing that is important to have automatic or semi-automatic tools that aid users in the organization of their digital photo collection, this document describes an algorithm that segment a collection of photos using their creation time. It is supported by two base assumptions. The first admits a bursty nature of the shots by the photographer [8]. This means that after a period of massive shooting, comes a period with no photos. Those empty time-slots acts as separators of events. The second assumption is that modern civilisation temporal regularities influence the creation

of photos. The capture of these regularities allows an approximation to the real events. Among them, the daily cycle is subject to a special concern, with an approach never used in this context, as far as we know.

## 1.2 Organization of the document

The document is organised as follows. Section 2 describes the segmentation problem and presents a formal definition of the concepts involved. It also indicates how the segmentations can be summarized, using numeric indicators and visual information. The algorithm of segmentation is then specified in section 3. Section 4 describes the testing of the algorithm. It focused on the collections used, the methods applied and the results. Section 5 present some of the work done in the field, viewed through the formalism developed in section 2. Finally, section 6 draws some conclusion and points out future work.

# 2 The segmentation problem

In this section, we formalise the notions of segments, segmentations, and a set of relations between them. Those formal definitions will support the comparison of different segmentations for the same set of photos.

Given a set of photos, let us denote by $T$ the ordered set of their creation time, represented as $\langle t_1, t_2, ..., t_n \rangle$, such

$$\forall i, j \in \mathbb{N}, \forall t_i, t_j \in T, i, j < |T| \; : \; i < j \Leftrightarrow t_i < t_j \tag{2.1}$$

where $\mathbb{N}$ is used as a index set. In this section, we will use the the subscript letters $i$ and $j$ to denote the indexes of $T$.

**Definition 2.1 (successor in T).** *An element $t_j \in T$ is the successor of $t_i \in T$, denoted by $(t_i)^{\succ} = t_j$, when there is no element in $T$ between $t_i$ and $t_j$, such*

$$\forall i, j \in \mathbb{N}, \forall t_i, t_j \in T \; : \; t_{i+1} = t_j \Leftrightarrow i + 1 = j \tag{2.2}$$

The elements of $T$ can be arranged in the form of non-empty, contiguous sub-sequences of $T$, leading to the notion of segment.

**Definition 2.2 (a segment).** *A segment, denoted by $s = [t^-, t^+]$, is a non-empty, contiguous, sub-sequence of $T$. $t^-$ is the lower limit of the segment and $t^+$ is the upper limit, where $t^- \leq t^+$, holding*

$$\forall t \in T \; : \; t^- \leq t_i \leq t^+ \Leftrightarrow t_i \in s \tag{2.3}$$

**Definition 2.3 (element of a segment).** *An element $t \in T$, is an element of a segment $s = [t^-, t^+]$, denoted $t \in s$, if and only if $t^- \leq t \leq t^+$*

Notice that a segment can be singular, when $t^- = t^+$, and can be equal to $T$, when $t^- = t_1$ and $t^+ = t_{|T|}$. From the definition of a segment, it is easy to see that two segments $s_1$ and $s_2$ are equal when both of their limits are equal, $t_1^- = t_2^-$ and $t_1^+ = t_2^+$. We will denote this relation by the symbol $=$. For simplicity, to represent $\neg(s_1 = s_2)$, we will use the symbol $\neq$.
To ease the understanding of the upcoming definitions, we will use three segments of $T$, defined as $s_1 = [t_1^-, t_1^+]$, $s_2 = [t_2^-, t_2^+]$ and $s_3 = [t_3^-, t_3^+]$.

**2.1 Proposition (properties for the equal relation):**
*The equal relation is reflexive, symmetric, antisymmetric and transitive.*

Since the demonstration of those properties are trivial, we will leave it to the reader.

**Definition 2.4 (precedence relation).** *Let $\prec$ be a order relation defined over $T$ and two segments $s_1 \prec s_2$, where any element of $s_1$ is lower than $t_2^-$. We say that $s_1$* ***precede*** *$s_2$ or $s_2$ succeeds $s_1$ when*

$$\forall s_1, s_2 \,:\, s_1 \prec s_2 \Leftrightarrow t_1^+ < t_2^- \tag{2.4}$$

**2.2 Proposition (properties for the precedence relation):**
*The precedence relation is irreflexive, asymmetric and transitive.*

PROOF The proof of irreflexive property is trivial, so it is left to the reader. Let $s_1 \prec s_2 \Rightarrow \neg(s_2 \prec s_1)$. Now assume that $s_1 \prec s_2 \wedge s_2 \prec s_1$ is true. From (2.4) we know that $t_1^+ < t_2^- \wedge t_2^+ < t_1^-$. This means there are elements between $t_2^+$ and $t_1^+$ that are shared between both segments, which contradicts definition 2.4, proving the asymmetric nature of the relation.
Finally, if $s_1 \prec s_2 \wedge s_2 \prec s_3$ then, from (2.4), we get that $t_1^+ < t_2^- \wedge t_2^+ < t_3^-$. From (2.3), if $t_2^- < t_2^+$ thus $t_1^+ < t_3^-$, which is the same as $s_1 \prec s_3$. This proves the transitivity for the relation. ∎

**Definition 2.5 (contained segments).** *A segment $s_2$ is **contained** in $s_1$, denoted by $\sqsubset$, when all elements of $s_2$ are also elements of $s_1$,*

$$\forall s_1, s_2 \,:\, s_2 \sqsubset s_1 \Leftrightarrow s_1 \neq s_2 \wedge t_1^- \leq t_2^- \wedge t_1^+ \geq t_2^+ \tag{2.5}$$

**2.3 Proposition (properties for the contained relation):**
*The contained relation is irreflexive, asymmetric and transitive.*

PROOF The irreflexive property is a direct consequence of (2.5), since the contained relation does not contemplate the equal case. Therefore, $\neg\ (s_1 \sqsubset s_1)$ is true, for any segment $s_1$.
We will prove the asymmetry, $s_2 \sqsubset s_1 \Rightarrow \neg(s_1 \sqsubset s_2)$, by contradiction. Let us assume that $s_2 \sqsubset s_1 \wedge s_1 \sqsubset s_2$. From (2.5) we can rewrite $(t_1^- \leq t_2^- \wedge t_1^+ \geq t_2^+) \wedge (t_2^- \leq t_1^- \wedge t_2^+ \geq t_1^+) \wedge (s_1 \neq s_2)$. $t_1^- \leq t_2^-$ and $t_2^- \leq t_1^-$ are true, if and only if $t_1^- = t_2^-$. Also, $t_1^+ \geq t_2^+$ and $t_2^+ \geq t_1^+$ are true, if and only if $t_1^+ = t_2^+$. In this situation, $s_1 = s_2$. This result in $s_1 = s_2 \wedge s_1 \neq s_2$ which is a contradiction.
Let us demonstrate the transitive property. If $s_2 \sqsubset s_1 \wedge s_1 \sqsubset s_3$, by (2.5) we can rewrite as $(t_1^- \leq t_2^- \wedge t_1^+ \geq t_2^+) \wedge (t_3^- \leq t_1^- \wedge t_3^+ \geq t_1^+) \wedge (s_1 \neq s_2) \wedge (s_1 \neq s_3)$. Since the relation $\leq$ is transitive, $t_1^- \leq t_2^- \wedge t_3^- \leq t_1^-$, means $t_3^- \leq t_2^-$. Using the same principle, $t_1^+ \geq t_2^+ \wedge t_3^+ \geq t_1^+$ led to $t_3^+ \geq t_2^+$. So, if $t_3^- \leq t_2^- \wedge t_3^+ \geq t_2^+$ holds, we say that $s_2 \sqsubset s_3$, proving the transitivity. ∎

**Definition 2.6 (a segmentation).** *Given $T$, a segmentation $S$ is a non-empty ordered set of segments from $T$, denoted by $S = \{s_a = [t_a^-, t_a^+] : t_a^-, t_a^+ \in T\}$, where $a$ is the index of a segment in the segmentation. For every segmentation, the following properties hold:*

*(i) The first element (lower limit) of the first segment is always the first element of $T$, $t_1^- = t_1$*

*(ii) The last element (upper limit) of the last segment is the last element of $T$, $t_{|S|}^+ = t_{|T|}$*

*(iii) The last element of a segment and the first element of the next segment are successor elements in $T$, $(t_a^+)^\succ = t_{a+1}^-$, where $a \in [1 \ldots |S| - 1]$*

From this point forward, the subscript letters $a$, $b$ and $c$ will denote the index of segments in segmentations. The cardinality of a segmentation $S$ ranges from 1 to $|T|$. In the first case, $t^- = t_1$ and $t^+ = t_{|T|}$. In the later, $S$ contains only singular segments.

**Lemma 2.1 (precedence relation).** $\forall s_a, s_b \in S_1 \; : \; s_a \prec s_b \vee s_b \prec s_a$

PROOF Lets choose a segment $s_a$ from $S$. Now, every other segment we may choose, lets say $s_b$, contains elements that are greater or lower than the ones from $s_a$, because segments from a given segmentation do not share elements. From the definition 2.2, we know that if they are greater, then $t_a^+ < t_b^-$. From from (2.4) this is the same as $s_a \prec s_b$. Otherwise, $t_b^+ < t_a^-$ or $s_b \prec s_a$. ∎

## 2.1 Relations between segmentations

Once the notion of segmentation is properly defined, we will present a series of binary relations that support the comparison between two segmentations. The only requisite to this comparison is that they are defined over the same $T$. We start by showing the possible scenarios that arise from a comparison, present in figure 2 to figure 6. Each one is a representation of a binary relation, where the rectangles represents the timestamps range of a segment.

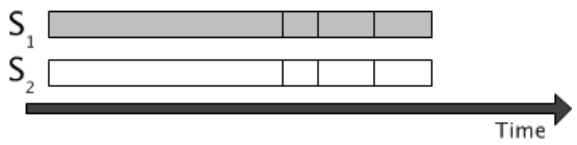


**Figure 2:** *Equal segmentations*

Figure 2 shows two **equal** segmentations $S_1$ and $S_2$. They segment $T$ in the same segments, so their cardinality is equal.

**Definition 2.7 (equal segmentations).** *Two segmentations $S_1$ and $S_2$ are* `equal`*, denoted by $S_1 \equiv S_2$, when all the segments at the same index, are equal. Thus*

$$\forall a, b \in \mathbb{N}, \forall s_a \in S_1, \forall s_b \in S_2, a \leq |S_1|, b \leq |S_2| : a = b \Rightarrow s_a = s_b \tag{2.6}$$

**Lemma 2.2.** *If $S_1 \equiv S_2$ then $|S_1| = |S_2|$*

PROOF Lets assume that $S_1 \equiv S_2 \wedge |S_1| \neq |S_2|$. If $|S_1| < |S_2|$, then the elements of $T$ are distributed through less segments. This makes impossible to have all segments equal at the same index. The same happens in the opposite case. Thus, when $S_1 \equiv S_2$ the distribution of the elements of $T$ in segments must be the same, making $|S_1| = |S_2|$. ∎

**2.4 Proposition (properties for the equal relation):**
*The* `equal` *relation, defined between segmentations, is reflexive, symmetric and transitive.*

Since the demonstration of those properties are trivial, we will leave it to the reader. For simplicity and clarity, we will denote the $\neg(S_1 \equiv S_2)$ as $S_1 \not\equiv S_2$.


**Figure 3:** *Refined segmentations*

Figure 3 illustrates the situation where one segmentation, $S_2$, is a **refinement** of another, $S_1$.

**Definition 2.8 (refinement relation).** *Let $S_1$ and $S_2$ be two segmentations of $T$. $S_2$ is said to be a `refinement` of $S_1$, denoted by $S_2 \lhd S_1$, when each segment of $S_2$ is equal or contained in one segment of $S_1$, thus*

$$\forall s_b \in S_2, \exists^1 s_a \in S_1, S_1 \not\equiv S_2 \; : \; s_a = s_b \vee s_b \sqsubset s_a \tag{2.7}$$

For simplicity and clarity, we will denote the $\neg(S_2 \lhd S_1)$ as $S_2 \not\lhd S_1$.

**Lemma 2.3.** *Since the refinement relation means a fine grain division of $T$, if $S_2 \lhd S_1$, then $|S_2| > |S_1|$*

PROOF If $S_2 \lhd S_1$ then from (2.7), $S_2 \not\equiv S_1$, then at least one segment of $S2$ is contained in one segment of $S_1$. This means that one or more segments of $S_2$ have at least one less element than a segment of $S_1$. Otherwise (2.7) will not hold. Thus, this element must be in another segment of $S_2$. This means the elements from $s_a$ are split in at least two elements of $S_2$, making $|S_2| \geq 1 + |S_1|$, proving the lemma 2.3. ∎

**2.5 Proposition (properties for the refinement relation):**
*The `refinement` relation is irreflexive, asymmetric and transitive.*

Let demonstrate those properties.

PROOF The demonstration of the irreflexive property is straightforward. Lets assume that $S_1 \lhd S_1$ is true. From lemma 2.3 we get $|S_1| > |S_1|$ which is a contradiction, since the cardinality of a segmentation cannot be greater than itself.
We will demonstrate the asymmetric property by contradiction. Let $S_2 \lhd S_1 \wedge S_1 \lhd S_2$ be true. Using lemma 2.3, we can rewrite $|S_2| > |S_1| \wedge |S_1| > |S_2|$, which is a contradiction, because a value cannot be simultaneous greater and lesser than some value. So it proves the asymmetry of the relation, $S_2 \lhd S_1 \Rightarrow \neg(S_1 \lhd S_2)$.

Finally, the demonstration of the transitive property. Let $S_2 \lhd S_1 \wedge S_1 \lhd S_3 \wedge S_2 \not\lhd S_3$ be true. Let $s_a$ be a segment from $S_1$, $s_b$ be a segment of $S_2$, and $s_c$ a segment from $S_3$. From (2.7), we know each segment of $S_2$ is equal or have its limits within the limits of one segment of $S_1$. The same happens between $S_1$ and $S_3$. Thus, we can rewrite $(t_a^- \leq t_b^- \wedge t_b^+ \leq t_a^+) \wedge (t_c^- \leq t_a^- \wedge t_a^+ \leq t_c^+) \wedge (t_c^- \geq t_b^- \vee t_b^+ \geq t_c^+)$. Since the relations $\leq$ and $\geq$ are transitive, $t_c^- \leq t_b^- \wedge t_b^+ \leq t_c^+ \wedge t_c^- \geq t_b^- \vee t_b^+ \geq t_c^+$. The expression is true if and only if $t_c^- = t_b^- \wedge t_c^+ = t_b^+$. But this is in contradiction with (2.7), because $S_2$ and $S_3$ cannot be equal. So, it is proved that the refinement relation is transitive. ∎

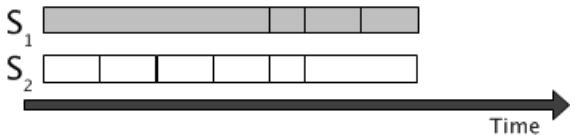


**Figure 4:** *Compatible segmentations*

Figure 4 represent the case of two **compatible** segmentations.

**Definition 2.9 (compatible segmentations).** *Let $S_1$ and $S_2$ be two segmentations of $T$. $S_1$ and $S_2$ are **compatible**, denoted by $\lessgtr$, when they are not equal, nor one is a refinement of the other, but is possible to divide them in such a way that at least one subset of each is a refinement of the other, holding*

$$\forall s_a \in S_1, \forall s_b \in S_2, t \in T \ :$$

$$t \in s_a \wedge t \in s_b \Rightarrow s_a = s_b \vee s_a \sqsubset s_b \vee s_b \sqsubset s_a \quad (2.8)$$

**2.6 Proposition (properties for the compatible relation):**
*The compatible relation is irreflexive and symmetric.*

PROOF Le us assume that $S_1 \leqslant S_1$ is true. From definition 2.9 we know that, at least, there is one segment from the first segmentation that is contained in another of the second segmentation. However, from (2.5) we know that two equal segments are not contained in each other. Then, $S_1 \leqslant S_1$ is false, since we are comparing the same segmentation. That is, the relation is irreflexive.
From definition 2.9 we know also that each segmentation share a common restriction: that some segments are a refinement of the other. This duality and the fact that all other segments are equal, make the compatible relation symmetric. ■

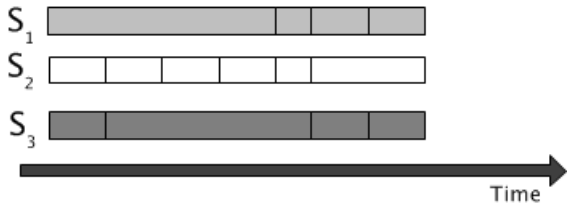


**Figure 5:** *Illustration of the non-transitive nature of the compatible relation*

Given the former definition, the compatible relation is not transitive. Figure 5 depicted an example used to demonstrate the non-transitive nature of the relation. Notice that $S_1 \leqslant S_2$ and $S_2 \leqslant S_3$, but when we compare $S_1$ and $S_3$, it is clear that $S_1 \leqslant S_3$ is not true. For instance, the first segment of $S_1$ do not hold (2.8).

The last relation between segmentations, showed in figure 6, is the **incompatible** relation, denoted by $\|$.

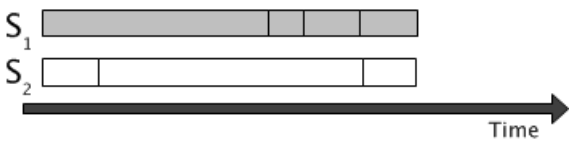


**Figure 6:** *Incompatible segmentations*

**Definition 2.10 (incompatible segmentations).** *Let $S_1$ and $S_2$ be two segmentations from the same $T$. They are incompatible, when they are not equal, $S_1$ is not a refinement of $S_2$ nor $S_2$ is a refinement of $S_2$, and they are not compatible.*

**Lemma 2.4.** *If two segmentations, $S_1$ and $S_2$, are incompatible, then they hold*

$$\exists s_a \in S_1, \exists s_b \in S_2 \; : \; s_a \neq s_b \wedge \\ \left[ (t_a^- < t_b^- \wedge t_a^+ < t_b^+) \vee (t_b^- < t_a^- \wedge t_b^+ < t_a^+) \right] \quad (2.9)$$

For the example showed in figure 6, lemma 2.4 is verified by the first segment of $S_1$ and the second segment of $S_2$.

PROOF We know that if $S_1$ and $S_2$ are not equal, at least one segment of each segmentation have their lower or upper limit different. Since they are not compatible nor their relation is a refinement, for some segments, the limits do not coincide and are not comprised within the limits of a segment of the other segmentation. The only case where this is possible, is when the lower limit or upper limit of one segment is between the limits of a segment of the other segmentation. ∎

**2.7 Proposition (properties for the incompatible relation):**
*The incompatible relation is irreflexive and symmetric.*

PROOF The irreflexive property is a consequence of the lemma 2.4, since no segmentation is incompatible with itself. Thus, $S_1 \| S_1$ is false because at least one segment must begin in the middle of the other, so they share elements. But this is in contradiction with lemma 2.1, which tells that segments precede each other and with definition 2.6, where no two segments share an element.
We will prove the symmetric property by contradiction. Lets assume that $S_1$ is incompatible with $S_2$ but the symmetric is not, that is $S_1 \| S_2 \wedge \neg(S_2 \| S_1)$ is true. We can expand this statement in respect with (2.9), as $[s_a \neq s_b \wedge exp_1] \wedge$

$[s_a = s_b \vee exp_2]$, where $exp_1$ and $exp_2$ are the second part of (2.9). We can see now if $S_1 \| S_2$ then $s_a = s_b$ must be false, forcing $exp_2$ to be true. The expansion of $exp_2$ is $(t_a^- \geq t_b^- \vee t_a^+ \geq t_b^+) \wedge (t_b^- \geq t_a^- \vee t_b^+ \geq t_a^+)$. As we know that $s_a$ and $s_b$ are not equal, the conditions where both limits are equal can be removed. Thus, we have two possibilities to make the statement true. When $t_a^+ > t_b^+ \wedge t_b^- > t_a^-$, that is the same as $s_b \sqsubset s_a$, and $t_a^- > t_b^- \wedge t_b^+ > t_a^+$, meaning that $s_a \sqsubset s_b$. However, this makes $exp_2$ in contraction with $exp_1$, since from (2.9) we know that at least one segment from both segmentations verify $(t_a^- \geq t_b^- \vee t_a^+ \geq t_b^+) \wedge (t_b^- \geq t_a^- \vee t_b^+ \geq t_a^+)$, that is, $s_a$ are not contain and does not contain $s_b$. This contradiction proves that $S_2 \| S_1 \wedge S_1 \| S_2$ is true. ∎

The incompatible relation is not transitive. At first, this may seem awkward. We will use an example to illustrate why, depicted in figure 7. $S_1$ is incompatible with $S_2$ due the first segment of $S1$ and the fourth of $S_2$. $S_2$ is incompatible with $S_3$ because of the second segment of $S_3$ and the forth segment of $S_2$. However, when we compare $S_1$ and $S_3$, instead of incompatible, we notice that $S_1 \lhd S_3$. This simple example shed any doubt about the non-transitive nature of the relation.

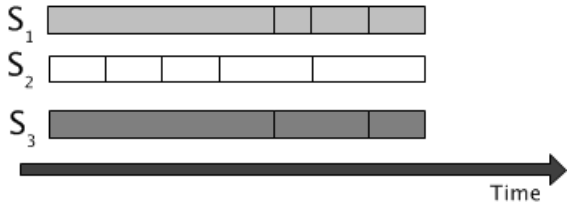


**Figure 7:** *Illustration of the non-transitive nature of the incompatible relation*

## 2.2 Distance function between segmentations

As stated before, the comparison between segmentations can only take place if they result from the same set of timestamps $T$. Sometimes, besides the relations between them, it is convenient to quantify how distant

two segmentations are. For example, when comparing segmentations produced by different algorithms.

To calculate more easily the similarity between a pair of segmentations, we need a different way of representing the segmentations. Using $T$ as the reference, we can represent each segmentation as a binary vector, with an equal size of $T$, defined in a normed vector space. Each position of the vector marks the absence or existence of a segment start, using 0 and 1 respectively. For instance, given a $T$ with 5 timestamps, $\vec{v}_1 = (1, 0, 1, 0, 1)$ and $\vec{v}_2 = (1, 1, 1, 1, 1)$ represent two segmentations, where the second is a `refinement` of the first. The nature of the segmentation and the selected representation, led to the following characteristics:

(i) It is impossible to have a vector filled with zeros, since a segmentation is a non-empty set of segments

(ii) The first position of the vectors represents the early segment of both segmentations, and thus, are always equal to one. This also means they are aligned with $T$

(iii) The number of ones ranges from 1 to $|T|$. The minimum happens when a segmentation has a single segment and the maximum is achieved when the segmentation has only singular segments

(iv) The number of zeros are comprised between 0 and $|T| - 1$, achieved when the segmentation has only singular segments and a single segment, respectively

With this representation, it is possible to use several distance functions defined in the literature, namely the Hamming distance [19], the one that will be used.

Although the computation of the distance is between two segmentations, in the notation we will use the vector representation instead. We will de-

| Relation | min | max |
|---|---|---|
| `Refinement` | 1 | $\|T\|-1$ |
| `Compatible` | 2 | $\|T\|-2$ |
| `Incompatible` | 2 | $\|T\|-2$ |

**Table 1:** *Distance ranges for segmentations with different relations*

note the function by

$$\Delta : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{N}_0, \text{where} \Delta(\vec{v}_1, \vec{v}_2) \stackrel{\text{def}}{=} \text{hamming\_distance}(\vec{v}_1, \vec{v}_2)$$

For a given finite $n$, $\Delta$ is a metric in the vector space of size $n$. It returns the number of mismatch symbols between two vectors. Besides, it has an important interpretation in the segmentation' domain. Since ones represent the beginning of segments, the hamming distance indicates the number of mismatches between segment' starts, despite the relation between segmentations. Nevertheless, for different relations, the distance ranges varies, as showed in table 1.

To complement the description of the subject, it is worth to mentioned the simplest $v$ for each possible relation. For the `Refinement` relation,the simplest case is given by

$$\vec{v}_1 = (1,0) \quad \text{and} \quad \vec{v}_2 = (1,1)$$

where $v_2$ refine $v_1$. For the `Compatible` relation, one case is given by

$$\vec{v}_1 = (1,0,1,1) \quad \text{and} \quad \vec{v}_2 = (1,1,1,0)$$

Finally, the simplest case for the `Incomparable` relation is

$$\vec{v}_1 = (1,0,1) \quad \text{and} \quad \vec{v}_2 = (1,1,0)$$

## 2.3 Descriptors for a segmentation

The function $\Delta$ is important to show how distant two segmentations are, despite the relation between them. However, it does not provide sufficient information about the segmentation itself, for instance, the number of segments or its relation with $T$, to name a few. To overcame this, we define a set of measures, that describe the segmentation.
The first measure is the **number of the segments** of a segmentation, denoted by $(.)^{\#}$. As its name indicates, the measure gives the number of segments in a segmentation, which can be helpful when we are comparing different segmentations from the same set $T$. The measure ranges from 1 to the size of $T$.
To widen the scope of usage, we define a relative version of the measure, called **relative number of segments**, denoted by $(.)^{\#}_r$. This measure is a normalization of the previous one. It is calculated as

$$(.)^{\#}_r = \frac{(.)^{\#}}{|T|}$$

Notice that in this case, it is possible to compare segmentations from different $T$, because the value is in the interval $]0, 1]$.

Next, we define the **minimum size of a segment** and the **maximum size of a segment**, denoted by $min_{\#}(.)$ and $max_{\#}(.)$, respectively. This measures shows the minimum (and maximum) number of elements in a segment for a given segmentation. From them is possible to calculate the range value for the cardinality of the segments.

We define other measure, the **average size of a segment**, denoted as $avg_{\#}(.)$. This measure gives the average cardinality of the segments in a segmentation. The value ranges from 1 to $|T|$.
There is also a normalized version of the average, called **relative average size**, denoted as $avg_r(.)$, calculated as

$$avg_r(.) = \frac{avg_{\#}(.)}{|T|}$$

This measure indicates the percentage of elements of $T$ per segment. Once more, it can be used to compared segmentations from different $T$.

The next measure is the **standard deviation of a segmentation**, denoted by $std_{\#}(.)$. This measure is similar to the standard deviation used in statistics, as it represents the standard deviation of the segments from the average size, for a given segmentation. It is calculated as

$$std_{\#}(S) = \sqrt{\frac{\sum (s_a - avg_{\#}(S))^2}{|S|}}$$

where $S$ is a given segmentation and $s_a$ is a segment from $S$, with $a \in [1..|S|]$.

## 2.4 Visualisation of segmentations

The use of these measures can be supplemented with a visual representation of the segmentations. We will use a histogram where the X-axis represents the number of photos and in the Y-axis the number of segments. Figure 8 shows a example of such a histogram. Segmentation $S$ has 18 segments, where two have a single element, five have two elements, ten have five elements and two have six elements.
Despite the simplicity of the visual representation, we need a way to settle the number of bins to use in the histogram. Description statistics have some methods to settle the width $h$ of the bins[1] for presenting frequency distributions. The method used is based on the Freedman-Diaconis rule [7], here rounded to the nearest integer,

$$h = \left\lfloor \left(2 \times IQR \times n^{-\frac{1}{3}}\right) + 0.5 \right\rfloor \tag{2.10}$$

where $IQR$ is the interquartile range [24] and $n$ is equal to $(S)^{\#}$. For the example depicted in figure 8, we have $IQR = 3$ which leads to $h = 2$. The number of bins $b$ is given by

[1]We assume the same width for all bins is enough for the purpose.

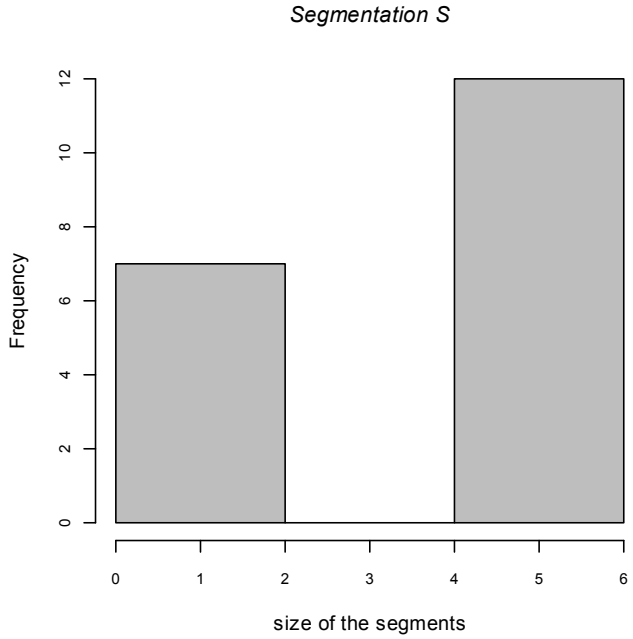


**Figure 8:** *Illustration of the visual representation of the segmentation*

$$b = \begin{cases} 1, & \text{if } max_{\#}(S) = min_{\#}(S) \\ \left\lceil \frac{max_{\#}(S) - min_{\#}(S)}{h} \right\rceil, & \text{otherwise} \end{cases}$$

For the given example $b = 3$.

This visualisation of a segmentation provides additional information about the distribution of the segments in the segmentation. In particular, it let us observe the dispersion of the segments sizes.

## 2.5 Joint visualization of the collection and segments

Following the same line of though, it would be important to have a visual representation of the collection and, at the same time, be able to see

the position of the segments on top of it. However, each collection may extend for a large period, with many segments to display. An attentive to show all the information would produce a cluttered image with few useful information. Thus, the visualization of the collection will be done by each logical day, on top of which we represent the segments. An histogram will represent the collection, with the width of the bins determined by (2.10). The X-axis denote the period depicted. Figure 9 shows an example of such representation. The collections of photos is represented by solid bars, while the segments are represented by the dashed line.

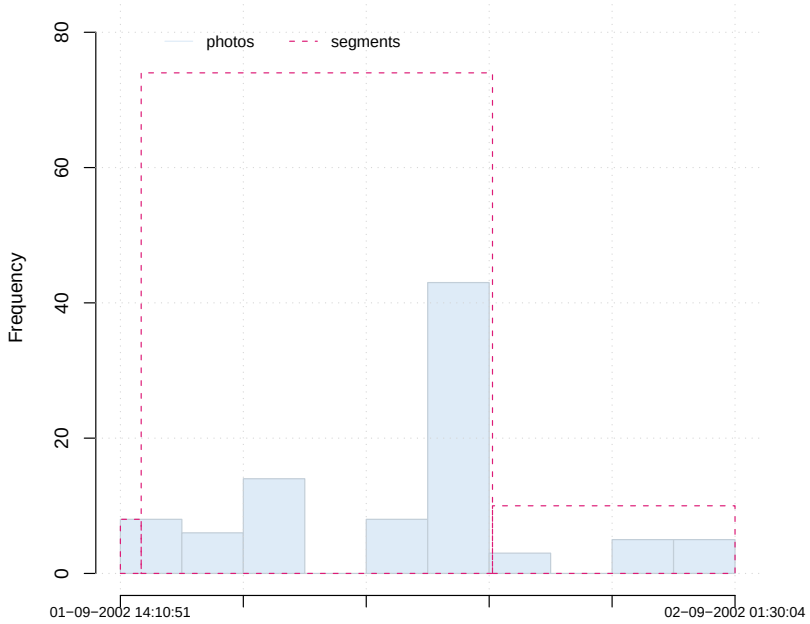


**Figure 9:** *Illustration of the visual representation of one logical day of the collection and the segments*

Even tough we are using different methods to represent both informations, the time boundaries are the same. Thus, the overlap of the two is consis-

tent, providing a interesting way to see the adjustment of the segments to the collection.

# 3 Segmentation algorithm

The algorithm described in this section relies on time to fulfil the segmentation of a collection of photos. Thus, is a requirement that each photo of the collection has the time of creation. However, it deals with time not as a sequence of timestamps but as a sequence that embodies regularities that have social meaning. This is important, since we are trying to find events and those are carried, generally, in some predefined zones of the day, week and year. As a starting point, it uses the day cycle as the main source of information for a first segmentation. However, the notion of day is extended beyond its classical boundaries (0 am to 11.59 pm). Figure 10 show the block diagram of the algorithm, representing its three steps, with its inputs and outputs. Before the segmentation procedure takes place, the collection is ordered using the photo's timestamps, guaranteeing the conditions stated in section 2.

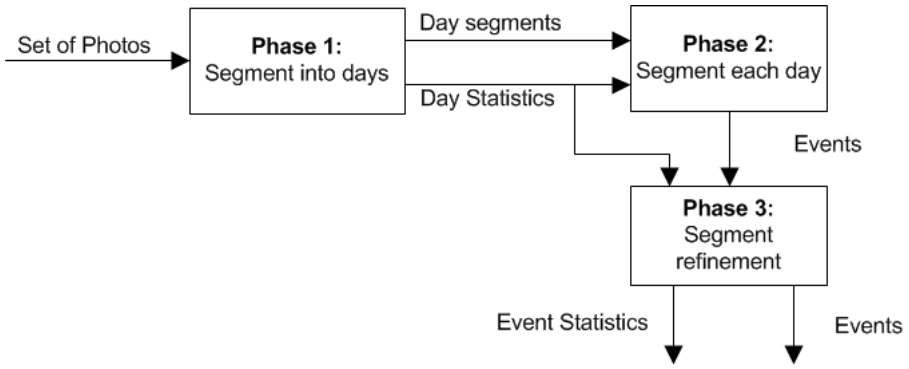


**Figure 10:** *Overview of the segmentation algorithm*

## 3.1 Phase 1 - segment roll into days

The first phase of the algorithm has two goals:

(i) produce a segmentation $S$ where each segment contains the timestamps for a single logical day;

(ii) produce a set of statistics for each segment, that will drive the later steps.

**Code 1** Segment into logical days

**Input:** An ordered set of timestamps $T$ and a threshold value $w$
**Output:** A segmentation $S$ and a set of statistics $STAT$ for each segment

```
PHASE_1(T, S, w)
 1  S = ∅
 2  STAT = ∅
 3  // Obtain the first timestamp
 4  lastTS = T(0)
 5  for i = 0 to T.length
 6      if DATE(lastTS) ≠ DATE(T(i)) ∨ T(i) − lastTS < w
 7          // Add the index that marks the end of a segment
 8          S.append(i)
 9          STAT.append(currStat)
10      else
11          currStat = UPDATESTATISTICS(currStat, T(i))
12      lastTs = T(i)
13  S.append(T.length)
14  STAT.append(currStat)
15  return STAT
```

The observation of the user's habits demonstrate that events follow some temporal regularity and have some degree of predictability [28]. The notion of a logical day follows closely the daily cycle, but is beyond the normal boundaries. The hours where people do their daily activities delimit a logical day. For example, if someone gets out of bed at 10 pm and goes to bed at 2 am of the next day, the logical day is confined to that period. This is true, even if there are no photos that document each moment of the day[2].

[2]Although in this case it is almost impossible to settle correctly the limits of the logical day.

However, if there are, they represent moments that occur in the period where people have their events, their activities, and so on. Nevertheless, the boundaries are not fixed - they can change in a daily basis, depending on the photos we have, making the notion of a logical day fuzzy. To settle them, more precisely, the upper bound, it is necessary to have a sequence of timestamps that spans from the last hours of one "normal" day into the early hours of the next one. If the timestamps fall within a predefined time window $w$, a parameter of the algorithm, they are considered to belong to the same logical day. If not, a logical and "normal" day is considered to be the same. The rational behind the value of this window, is that most people sleep at night, at least a few hours. Thus, after an event where they took pictures, came up a gap that is enough to delimit the logical day.

Besides the segmentation of the timestamps, this phase of the algorithm also produces the statistics for each segment. They are important for the next steps of the algorithm, namely, to detect events inside the day, and include:

(i) the number of photos in the segment;

(ii) the maximum;

(iii) the minimum and;

(iv) the average time difference between two consecutive timestamps.

With such information, it is possible to adjust the algorithm to the shot behaviour the photograph had each day. Code 1 shows the pseudocode of the algorithm for the first phase of the segmentation process.

## 3.2 Phase 2 - segment each day

The second phase of the algorithm takes the segmentation $S$ and the statistics $STATS$ to produce a segmentation $S_d$, where $S_d \lhd S$. This refinement divides each segment of $S$, a set of timestamps for one logical day, into

**Code 2** Segmentation of the collection logical days

**Input:** An ordered set of timestamps $T$, the day segments $S$, their statistics $STATS$ and a real value $f$

**Output:** A refinement $S_d$ of the segmentation $S$

```
PHASE_2(T, S, STATS, f)
 1  // The new segments
 2  S_d = ∅
 3  // The index of the current element of T
 4  idx = 0
 5  lastTS = T(0)
 6  for i = 0 to S.length
 7       currStats = STATS(i)
 8       for j = idx to idx + S(i).length
 9            if lasTS − T(j) > computeThreshold(currStats, f)
10                 S_d.append(j)
11            lastTS = T(j)
12       idx = idx + S(i).length
13       // Maintain the segments from S
14       push(S_d, idx)
15  return S_d

COMPUTETHRESHOLD(stats, f)
1  return stats.average × f + (stats.max − stats.min) × f
```

fine grained segments that are close to base events (for example, visiting a museum). The algorithm uses the statistics calculated in the previous phase. Since these can change each day, the algorithm adapts the division points to each daily set of timestamps. The decision to create a new segment is based in a reference value, that consist in the sum of two parts. One part is a fraction of the average time difference between consecutive photos. Another, is a fraction of the maximum and minimum time differences. A value $f$, whose range is $0.1 \leq f \leq 0.9$, controls the fraction that will be used. Code 2 shows the pseudocode for the second second phase

of the segmentation. The division point is decided in line 9.

## 3.3 Phase 3 - fine tuning

**Code 3** Fine tuning the segments

**Input:** An ordered set of timestamps $T$, the list of the day segments indexes $S$ and the threshold value $w$
**Output:** A fine tuned segmentation $S$

```
PHASE_3(T, S, w)
1  for curIdx = 0 to S.length
2      ADJUSTDAYTURNSEGMENTS(curIdx, S, T, w)
3  return

ADJUSTDAYTURNSEGMENTS(curIdx, S, T, w)
1  if curIdx ≠ 0
2      TSprevSegment = T(S(curIdx − 1))
3      TSsegment = T(S(curIdx))
4      if DAY(TSprevSegment) + 1 == DAY(TSSegment)
           and TSSegment − TSprevSegment > w
5          S.removeAt(curIdx)
```

The previous phases of the segmentation process already group the timestamps, the segments, which are close to the base events. However, there are still refinements to do, namely, guarantee the notion of a logical day, as the window $w$ is generally greater than the reference value for each day. During the second phase of the segmentation, some of the photos that belong to the same logical day can be put into different different segments, breaking their logical relation. This means that photos that are taken pass 0 am and have a sufficient [3] time gap in front, can be incorporated in the

[3]The notion of "sufficient" is related with our habits, for instance, our necessity to sleep some hours.

previous segment, since the activity is broadly the same. The procedure `AdjustDayTurnSegments` in code 3 do this adjustment.

# 4 Evaluation

In this section we will test the segmentation algorithm against some personal collection of photos. The goals of the test are:

(i) Find out if there is an appropriate default value for the parameters, that suits the general needs for different collections;

(ii) Study the relation between the segmentations when the parameters changed.

Before we describe the tests and the method used, it is important to say that it is the author belief that there is no algorithm capable of segment all the collections as the user does. There are so much context and semantics associated with a photo collection that some aspects will always be left off for the user to take care. Nevertheless, the help of such "imperfect" algorithm is very important, as it lifted some of the manual labour.

## 4.1 The collections

In this evaluation, we use five different types of collections:

(i) A set $C_1$ with 952 photos, that document a seven day vacation trip;

(ii) A set $C_2$ with 730 photos, that document an eight day trip;

(iii) A set $C_3$ of 113 photos that occur over five years, with sparse shots and long empty periods;

(iv) A set $C_4$ of 25 photos that occur over two days and includes several bursts of photos and an outlier;

(v) A set $C_5$ with 36 photos with 30 minutes of duration.

Collection $C_1$ documents a typical vacation trip. It contains several episodes of the vacation, including the departure and arrival. Among them, there are many sequences of photos with few minutes apart, some capturing alternative shots of the same scene. It includes at least one case where the

notion of a logical day applies. There is only one camera, but it was used for more than one photographer.

The collection $C_2$ also documents a vacation trip. Most of the images depict local scenes, monuments and landscapes. The photos were taken from just one camera and the owner is different from the above collection.

The third collection, $C_3$, gather several photos taken along five years. It is a typical role of occasional shots made, for example, from a cellphone. There are many single photos and the events contains only a dozen photos at most. The owner of the collection is different from the previous two.

Collection $C_4$ contains photos from a weekend travel. Most of the photos are bursts, trying to capture the motion that exists in the depicted activity. The number of events in each day are few, just two or three.

Finally, collection $C_5$ document a short visit to a museum. It consists in a single episode with multiple events, representing the passage between rooms.

## 4.2 Default values

Default values play an important role in the life of an algorithm. It promotes its usage, even if the user is not an expert of the domain, or if it simply does not understand the scope of the parameters. Those values make the algorithm useful off-the-shelf. Thus, for an automatic segmentation algorithm, there is a strong need for those values, in order the segmentations produced fulfil the user's needs over different collections.

The algorithm was run 5 times for each collection. In each run, the sensitive parameter $f$ was changed with the values $0.1, 0.25, 0.5, 0.75, 0.9$. Each run was labelled from $a$ to $e$ respectively. Do notice this parameter govern the level of segmentation, i.e., when is small it forces the algorithm to

segment more; when is large it produces larger and fewer segments. The descriptors for each run can be found in the appendix-A (page 49), depicted on figure 11 to figure 15.

Let start with figure 11. In each graphic, we have the run identification in the X-axis and the value for the descriptor in the Y-axis. From left to right top to down, the figure has: the number of photos in the collection, the number of segments, the relative number of segments, the average cardinality of the segments, the relative average, the standard deviation from the average size of the segments, minimum size of a segment and the maximum size of a segment. When we observe the values for each descriptor, we can see the number of segments decreasing when the sensitivity grows. However, this decrease slows after the third run. In fact, if we see the relative number of segments, it almost freeze from the third run upwards. Giving these indicators, the value of 0.5 is a good candidate suitable to segment this collection. Moving to collection $C_2$, we see something similar when we watch figure 12. The numbers of segments reach the stability after the second run. The same happens with the collection $C_3$ (figure 13). However, the number of segments does not vary as the previous ones, which are no surprise, since the collection has episodes from five years, each one with few photos. Nevertheless, it seems that after the second run, the number of segments steadies. Moving to the collection $C_4$, we found a similar scenario when observe figure 14. The number of segments does not vary much and from the third run forward the value stabilise. Almost the same happens with the collection $C_5$. The number of segments also stabilises from the third run forward. These observations suggest that the default value should be settle between 0.25 and 0.5, with some advantage for the latter.

## 4.3 Relations between segmentations

In the previous test, we have changed the sensitivity parameter of the algorithm. This means the number of segments tend to decrease when this value increase towards 1. What are the effects in the segmentations? Are they compatible with each other or not? To answer these questions, we compare each of the segmentations produced in each run, for a given collection. The results of this comparison can be found in the appendix B (page 57), depicted in figures 16, 18, 20, 22 and 24. To simplify the reading, the results are present in a contingency-like fashion. The X-axis and Y-axis represents the segmentations produced for each run, according to the labelling already established (from $a$ to $e$). Observing those figures, is clear the relations bounce between equal and refinement for all the cases. This means that changing the sensitivity parameter produce more segments, but preserve at the same time some key points that support the refinement at the worst case[4]. Since the collections were chosen to be as representative and diverse as possible, it seems this behaviour is independent of the collection itself.

Figures 17, 19, 21, 23 and 25 depict the distance between the segmentations produce by each run. As we can see, the distance varies more when the number of photos in the collection is greater (collection $C_1$ and $C_2$). The impact of the sensitive parameter is important as it can produce near more 10% of mismatches for the same collection (for collection $C_1$). Nevertheless, this difference is only relevant when the parameter is at his extremes. For the values suggested earlier, the differences are small even for the large collections.

[4]We consider there is an order of compatibility where *Equal* is more compatible, followed by the *Refinement* and then by the *Compatible* relation.

# 5 Related work

Several studies (for example [8]) suggest that users take photos in bursts, which can be used to separate different groups of photos. Also, psychological studies (for example, [12]) shows the importance of time in our process of recalling past events. So, it is not odd to see the information used to perform the event detection in the work presented here has time as the common denominator. In the work under analysis, time is always used to perform the first segmentation of photos, where each segment are close as possible to an event. Besides time, the content of photos [16, 21, 23] and the GPS information [4, 20] are the most used features to fine tune the segments. Before we are going in details, it worth clarifying the meaning of two terms that will be used frequently in this section: segmentation and clustering. A segmentation is a division of a totally ordered set into different parts, the segments. A clustering procedure also divides a set, known as clusters, but it does not need to be ordered.

One of the most referenced work in the field was developed by Plat, et al [21]. It uses the time as the main source of information to segment a digital photo collection, using the time difference (gap) between two consecutive photos and comparing it to the neighbourhood average differences. The size of the neighbourhood depends on a parameter $D$. Since the goal is to help the visualisation of the photo collection, the segments should have a maximum size, which has a predefined value. Whenever the segmentation produces segments whose cardinality exceeds that maximum, they are refined using a content-based clustering technique.

Loui, et al. [16], presents an algorithm that cluster photos in two levels: events and subevents. The main purpose is to automatically generate albums from a collection of photos. The time-based clustering takes the assumption that time differences between photos in the same event are smaller than time differences between photos from different events. The time clustering uses a time difference histogram, scaled along the time dif-

ference axis, to accommodate collection that spans several years. Then, the histogram is divided in two using the K-Means [17] algorithm. The cluster with higher values contains the time differences that should be used to separate events. In segmentation terms, those are the boundaries of the segments. Next, each segment is split into fine grain segments, representing subsevents, using a content-based clustering technique. The subevent segmentation is a refinement of the event segmentation.

Naaman, et al. [20], describe a system that uses time and location to automatically organise a personal photo collection, into groups of events or geographical locations. At first, they use the time gaps and geographical distance between pairs of consecutive photos, to settle event boundaries. A gap of $h$ hours between consecutive photos often indicates a new event, as within one event the photos are taken at a steady rate. Photos that differ with more than 12 hours are considered to belong to different events. Finally, the physical distance between locations of two consecutive photos can be also used to mark the generation of a new event.

In their work, Suh et al. [23], use event clustering for semi-automatic photo annotation. They present a framework that helps users in the annotation process, providing groups of photos that can be annotated in bulks, with two main targets in mind: events and people. These two targets are treated independently, so we will focus on the time algorithm only. The grouping method that uses time is based on the timestamps and produce a segmentation of the photo's collection, based on Platt's algorithm. However, the original algorithm was adapted to incorporate the notion of hierarchy. It identifies different event levels and let the user choose the correct level.

Cao et al. [4], use hierarchical event and scene models to annotated photo collections. Their idea is to explore the correlation that exists between photos in the collection, especially their location. To settle the events, they use a clustering method that group the collection into event-based subcollections. This procedure uses time and GPS information. First, a segmen-

tation is made using only on the time information, using the mean-shift algorithm [5]. Then, for the photos that have simultaneously time and GPS location, the procedure is rerun, leading to another division. This two pass algorithm is necessary since not all photos have GPS information, and thus, the time segments serve as a solid grouping feature.

# 6 Conclusion

This document describes an algorithm that aims to produce segmentations of personal photo collections, considering some relevant cycles of the day life. The segmentation is just a starting point that presents the photos to the users in groups that probably overlap the real events. In this work, there are two main contributions, whose conclusions we briefly present.

The first, is the creation of a formal framework that, among others, define the notion of segment and segmentation, their relations and properties. This formal notation is, as far as we know, a novelty, and enable a more robust and reliable comparison of segmentations, either produced by automatic algorithms or done manually by the users. In the literature, the evaluation tests do not resort to any formality, are hard to reproduce, and the properties of the segmentations were left behind, in favour of subjective feedback from the users.
The second contribution is the introduction of the notion of logical day, that breaks the collection in moments different than the classical 0 am. Some of the most used desktop software for managing collection of photos [2, 9, 27], uses the day as a fixed segment point. However, the behaviour of people is not tight to such rigid schedules. Namely, during the most documented episodes (e.g. a vacation), the occurrence of the events fall off the regular day cycle. This important, yet, neglected feature was never incorporated in the segmentation's algorithms proposed by the scientific community. The usage of such a notion extends the day towards the early hours in the next day. The current implementations is able to detect it, if there is a shot pattern prior the 0 am.

The evaluation of the algorithm shows that it has some desirable properties when the most important parameter, the sensitivity $f$, is changed. The segmentations produced with different values keep the compatibility between them, sharing a refinement relation, in the worst case. Thus, the algorithm converges toward the same events. It was also demonstrated

that after some value, the parameter's effect on the segmentations is small, making possible to define a default value of 0.5 for $f$. Although this value is not consensual, it was found to be a good starting point.

## 6.1 Future work

The segmentation algorithm uses, at this point, only temporal information to produce a segmentation of a collection of photos. However, time is just a small part of the metadata available in each digital photo. There are other types of information that can improve the detection and representation of the events. Namely, the GPS data can be use to improve the phase 3 of the algorithm. Besides, it can unveil higher level cycles, that can be used to improve the representation of the events, as suggested by figure 1.

Another topic that has room for improvement is the definition of the value to assign to the parameter $w$. This value is used by the algorithm to settle the boundaries of the logical day, and it is fixed for all the photo collection. It would be interesting to have a value of $w$ for each day, depending of the observation of the shot pattern. This will improve the detection of the logical days, giving the opportunity to adequate the value to different behaviours.

There is still some work to be done to evaluate the algorithm. All the procedures discussed in section 4, test the algorithm for its properties and characteristics, using the formalism introduced in section 2. However, it is important to test the user's reaction to the segmentations found by the algorithm. Among others, it is relevant to test the "a priori" and "a posteriori" reactions. The first is when the collection is presented already segmented and the user only signals which segments are wrong. The second is to make the user segment the collection manually first, and then compare it to the segmentations found by the algorithm. It is our conviction the results will differ in both cases.

# Nomenclature

| | |
|---|---|
| $T$ | an ordered set of timestamps |
| $\|.\|$ | the size (cardinality) of the a set |
| $s = [t^-, t^+]$ | represents a segment, which is is a contiguous, non-empty, monotone, sub-sequence of $T$ elements |
| $(.)^\succ$ | successor element in $T$ |
| $\prec$ | precedence relation, defined for segments |
| $=$ | equal relation, defined for segments |
| $\neq$ | indicate that two segments are not equal |
| $\sqsubset$ | contained relation, defined for segments |
| $S$ | a segmentation, which is a non-empty ordered set of segments from $T$ |
| $t_i^-$ | the first element (lower limit) of a segment, at index $i$ in a given segmentation |
| $t_i^+$ | the last element (upper limit) of a segment, at index $i$ in a given segmentation |
| $\equiv$ | equal relation, defined for segmentations |
| $\not\equiv$ | indicate that two segmentations are not equal |
| $\lhd$ | refinement relation, defined for segmentations |
| $\not\lhd$ | indicate that one segmentation is not a refinement of the other |
| $\lesseqgtr$ | compatible relation, defined for segmentations |
| $\parallel$ | incompatible relation, defined for segmentations |

| | |
|---|---|
| $\vec{v}$ | binary vector that represents a segmentation of $T$ |
| $\Delta$ | distance between two segmentations |
| $b$ | number of bins for a given histogram |
| $h$ | the width of the bins used in frequency distributions |
| $(.)^{\#}$ | number of the segments of a segmentation |
| $(.)_r^{\#}$ | relative number of the segments of a segmentation |
| $min_{\#}(.)$ | minimum size of a segment in a segmentation |
| $max_{\#}(.)$ | maximum size of a segment in a segmentation |
| $avg_{\#}(.)$ | average cardinality of the segments in a segmentation |
| $avg_r(.)$ | relative average size of the segments in a segmentation. It gives the percentage, in average, of each segment in the whole $T$ |
| $std_{\#}(.)$ | the standard deviation for the segments of a segmentation from the average size |
| $IQR$ | the interquartile range |

# References


[1] ADOBE. Photoshop cs5, 2011. [Online; accessed 08-February-2011], available at http://www.adobe.com/products/photoshop/compare/.

[2] APPLE. Iphoto'11, 2011. [Online; accessed 08-February-2011], available at http://www.apple.com/ilife/iphoto/.

[3] APPLE. Mac aperture 3, 2011. [Online; accessed 08-February-2011], available at http://www.apple.com/aperture/.

[4] CAO, L., LUO, J., KAUTZ, H. S., AND HUANG, T. S. Annotating collections of photos using hierarchical event and scene models. In *CVPR* (2008), IEEE Computer Society.

[5] COMANICIU, D., AND MEER, P. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 24*, 5 (may 2002), 603 –619.

[6] CUI, J., WEN, F., XIAO, R., TIAN, Y., AND TANG, X. Easyalbum: an interactive photo annotation system based on face clustering and re-ranking. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2007), ACM, pp. 367–376.

[7] FREEDMAN, D., AND DIACONIS, P. On the histogram as a density estimator: L 2 theory. *Probability Theory and Related Fields 57*, 4 (1981), 453–476.

[8] GARGI, U. Consumer media capture: Time-based analysis and event clustering. Tech. rep., Technical Report HPL-2003-165, HP Laboratories, 2003.

[9] GOOGLE. Picasa 3.8, 2011. [Online; accessed 08-February-2011], available at http://picasa.google.com/.

[10] GYE, L. Picture this: the impact of mobile camera phones on personal photographic practices. *Continuum 21*, 2 (2007), 279–288.

[11] HANBURY, A. A survey of methods for image annotation. *Journal of Visual Languages & Computing 19*, 5 (2008), 617 – 627.

[12] JANSSEN, S., CHESSA, A., AND MURRE, J. Memory for time: How people date events. *Memory and Cognition 34*, 1 (2006), 138.

[13] KANKANHALLI, M., AND RUI, Y. Application Potential of Multimedia Information Retrieval. *Proceedings of the IEEE 96*, 4 (2008), 712–720.

[14] KINDBERG, T., SPASOJEVIC, M., FLECK, R., AND SELLEN, A. The ubiquitous camera: An in-depth study of camera phone use. *IEEE Pervasive Computing 4*, 2 (2005), 42–50.

[15] KUSTANOWITZ, J., AND SHNEIDERMAN, B. Motivating annotation for personal digital photo libraries: Lowering barriers while raising incentives. *Univ. of Maryland Technical Report HCIL-2004 18* (2005).

[16] LOUI, A., AND SAVAKIS, A. Automated event clustering and quality screening of consumer pictures for digital albuming. *IEEE Transactions on Multimedia 5* (2003), 390–402.

[17] MACQUEEN, J. B. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Symp. Math. Statistics and Probability* (1967), vol. 1, pp. 281–297.

[18] MOSLEH, F. Cameras in handsets evolving from novelty to dsc performance. Tech. rep., Kodak, 2008. [Online; accessed 25-Maior-2010].

[19] MYATT, G. *Making sense of data: a practical guide to exploratory data analysis and data mining*. Wiley-Blackwell, 2007.

[20] NAAMAN, M., SONG, Y. J., PAEPCKE, A., AND GARCIA-MOLINA, H. Automatic organization for digital photographs with geographic coordinates. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries* (New York, NY, USA, 2004), ACM Press, pp. 53–62.

[21] PLATT, J., CZERWINSKI, M., AND FIELD, B. Phototoc: automatic clustering for browsing personal photographs. *Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference 1* (Dec. 2003), 6–10 Vol.1.

[22] SETHI, I., COMAN, I., AND STAN, D. Mining association rules between low-level image features and high-level concepts. *Proceedings of the SPIE Data Mining and Knowledge Discovery 3* (2001), 279–290.

[23] SUH, B., AND BEDERSON, B. B. Semi-automatic photo annotation strategies using event based clustering and clothing based person recognition. *Interacting with Computers 19*, 4 (2007), 524 – 544.

[24] UPTON, G., AND COOK, I. *Understanding statistics*. Oxford University Press, 1996.

[25] VON AHN, L. Games with a purpose. *Computer 39*, 6 (2006), 92–94.

[26] YAN, R., NATSEV, A., AND CAMPBELL, M. Hybrid Tagging and Browsing Approaches for Efficient Manual Image Annotation. *IEEE MultiMedia* (2009), 26–41.

[27] YORBA. Shotwell, 2011. [Online; accessed 08-February-2011], available at `http://yorba.org/shotwell/`.

[28] ZERUBAVEL, E. *Hidden Rhythms: Schedules and Calendars in Social Life*. University of California Press, 1985.

[29] ZUZANEK, J., AND SMALE, J. Life-cycle variations in across-the-week allocation of time to selected daily activities. *SOCIETY AND LEISURE-MONTREAL- 15* (1993), 559–559.

# Appendices

# A Evaluation descriptors

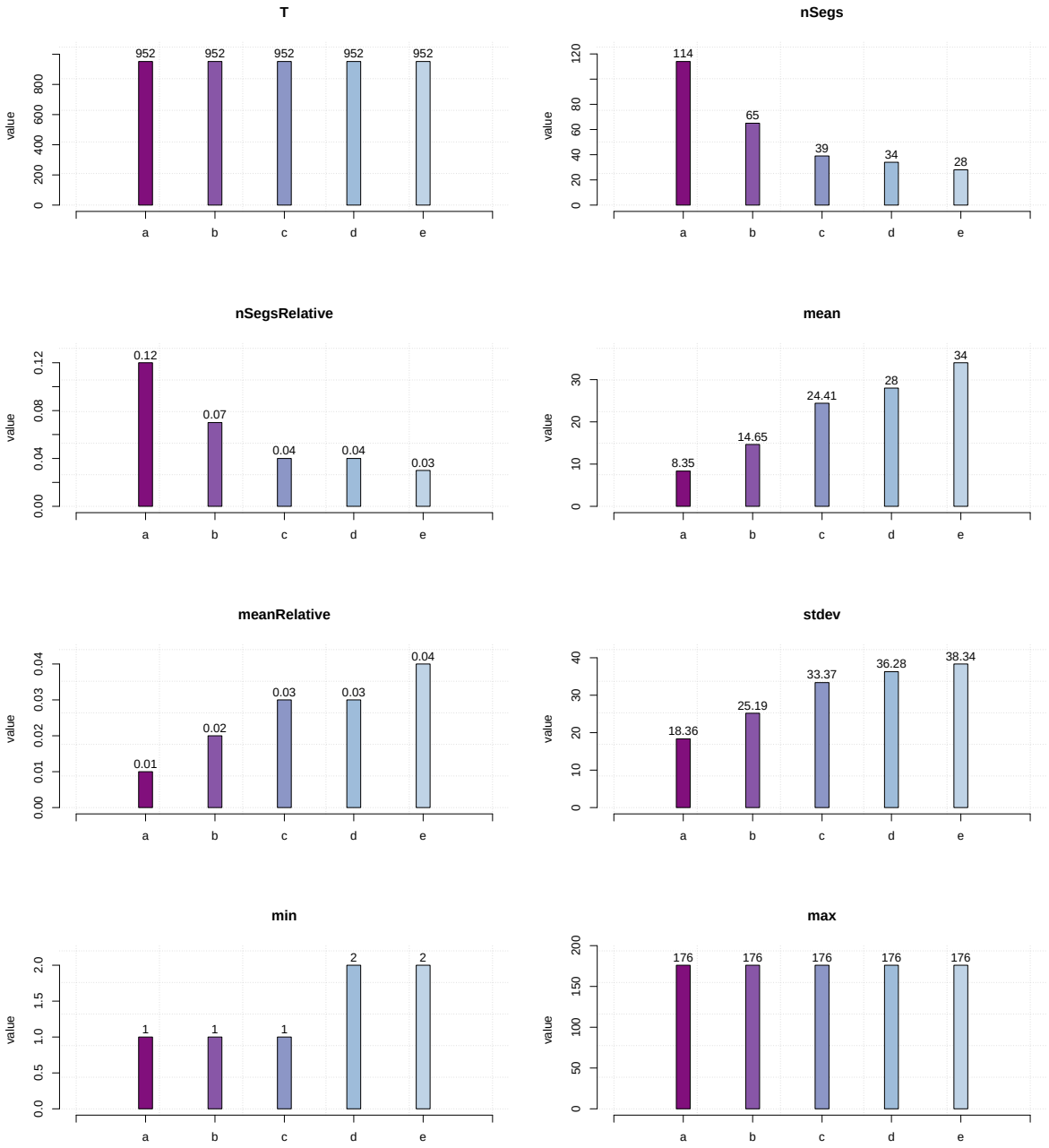


**Figure 11:** *Descriptors for the collection $C_1$, after five segmentations.*

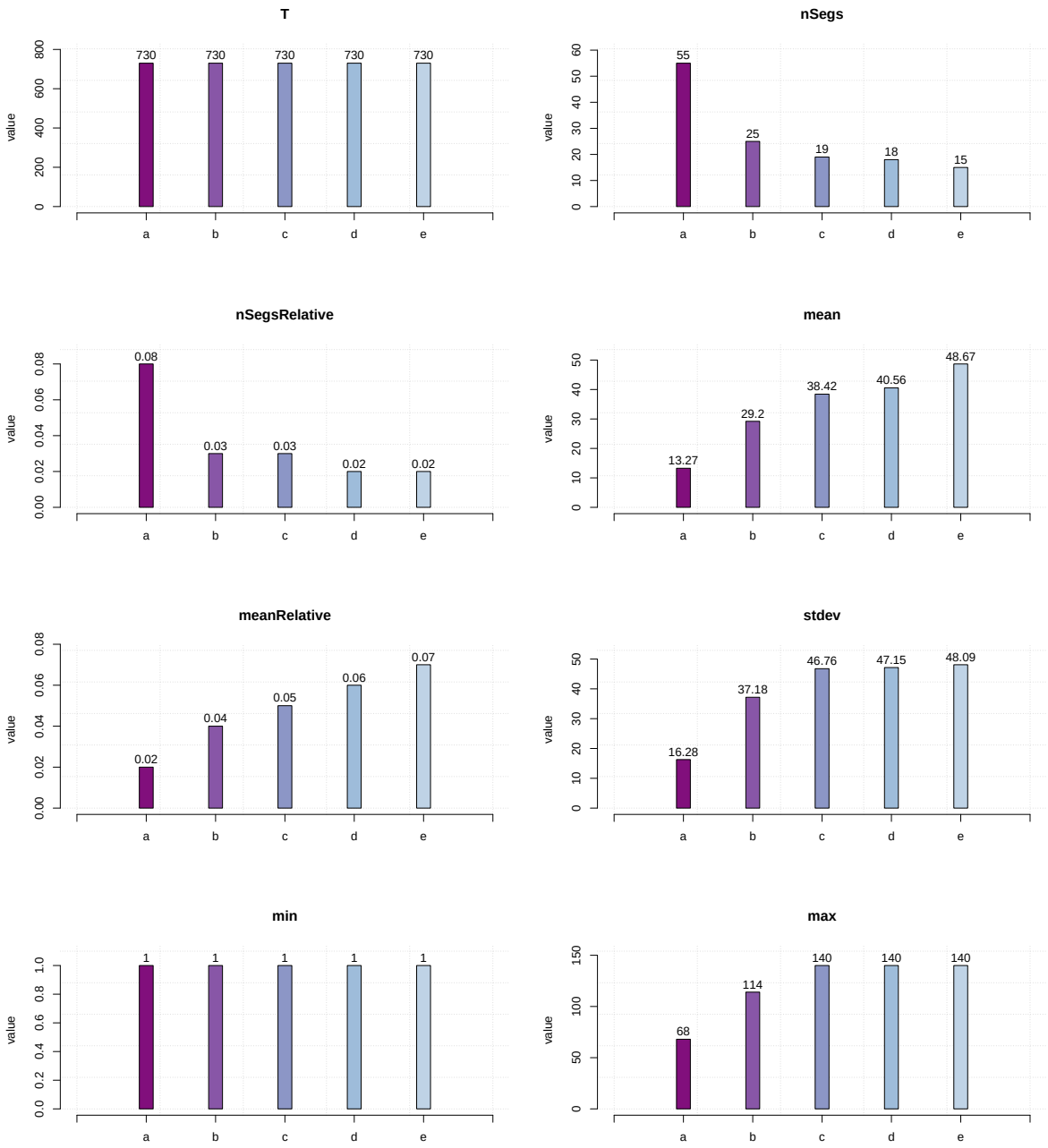


**Figure 12:** *Descriptors for the collection $C_2$, after five segmentations.*

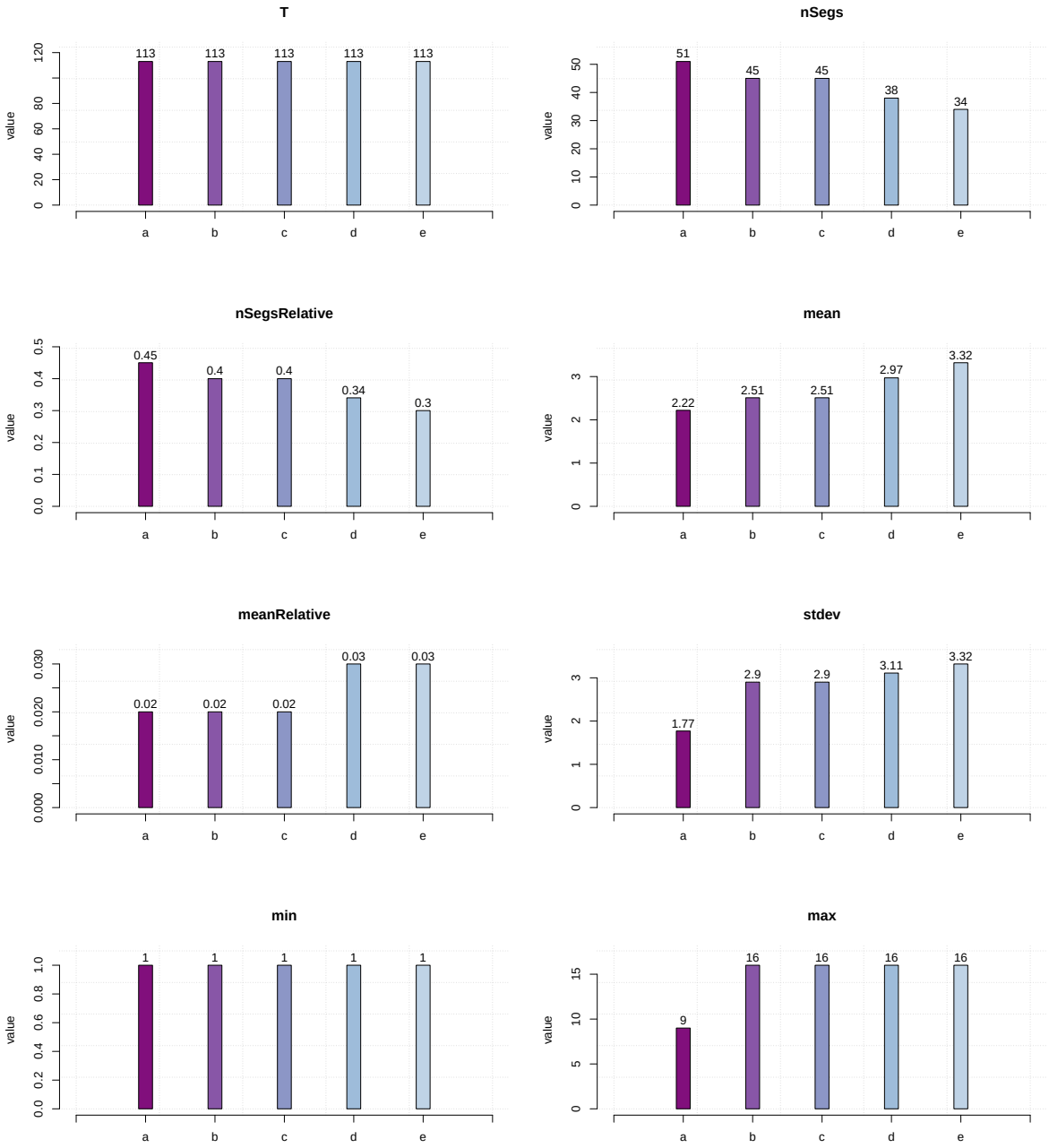


**Figure 13:** *Descriptors for the collection $C_3$, after five segmentations.*

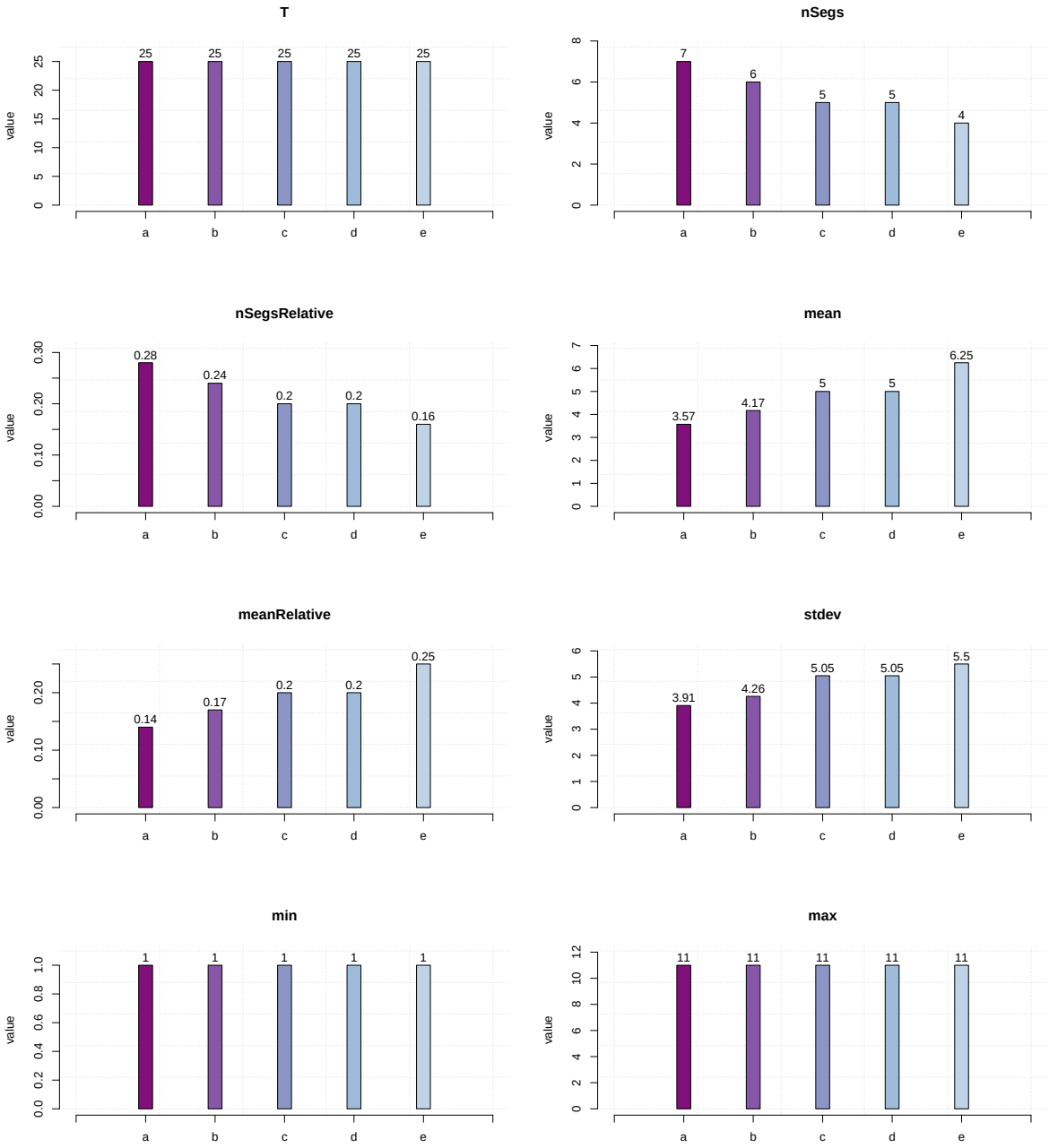


**Figure 14:** *Descriptors for the collection $C_4$, after five segmentations.*

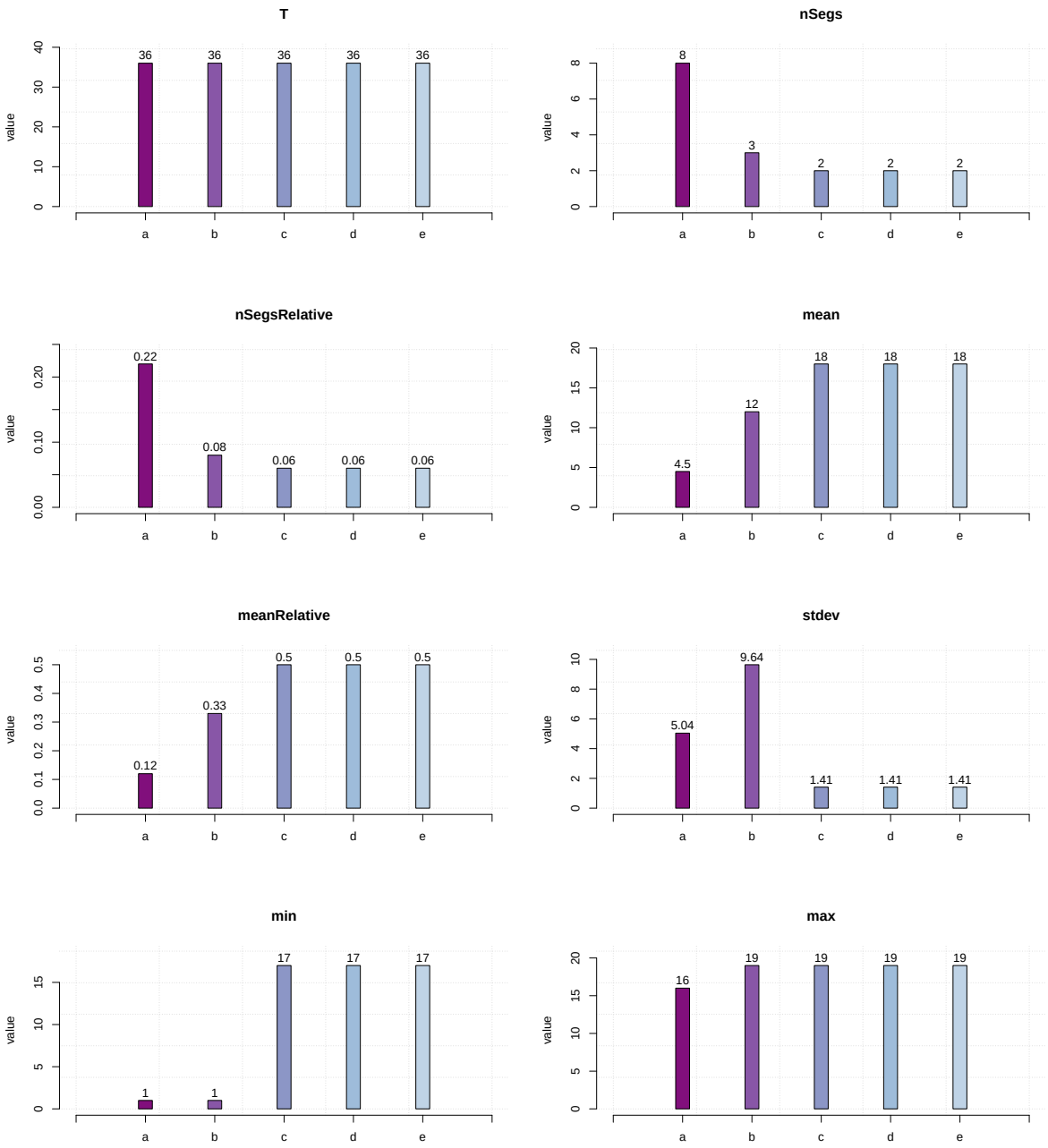


**Figure 15:** *Descriptors for the collection $C_5$, after five segmentations.*

# B Comparison of the segmentations

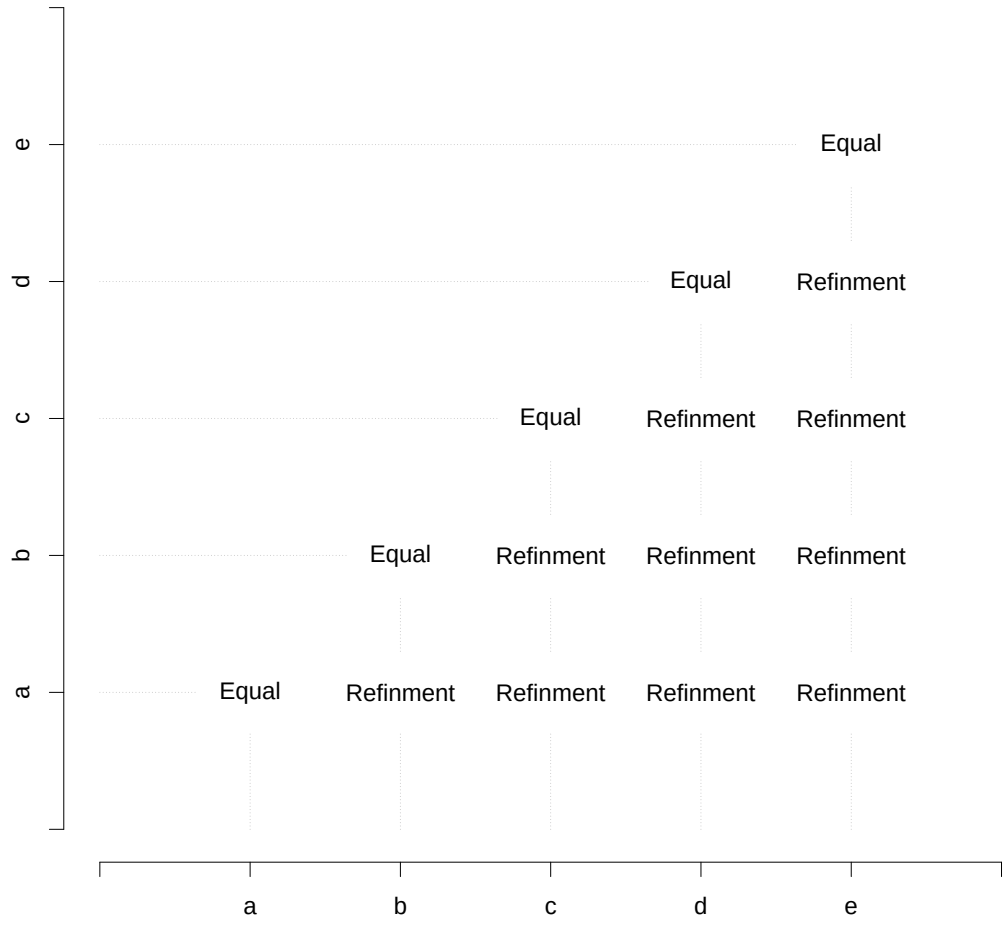


**Figure 16:** *Comparison of the segmentations produced in different runs, for the collection $C_1$.*

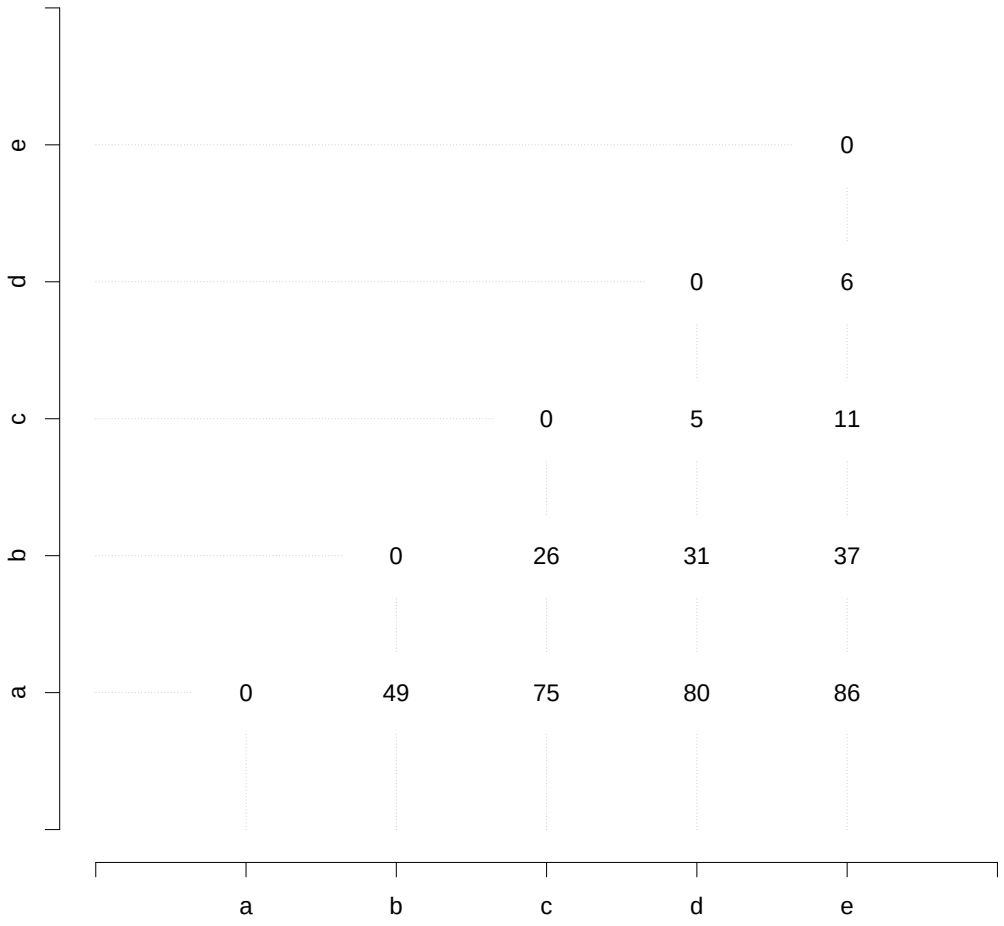


**Figure 17:** *Distance of the segmentations produced in different runs, for the collection* $C_1$.

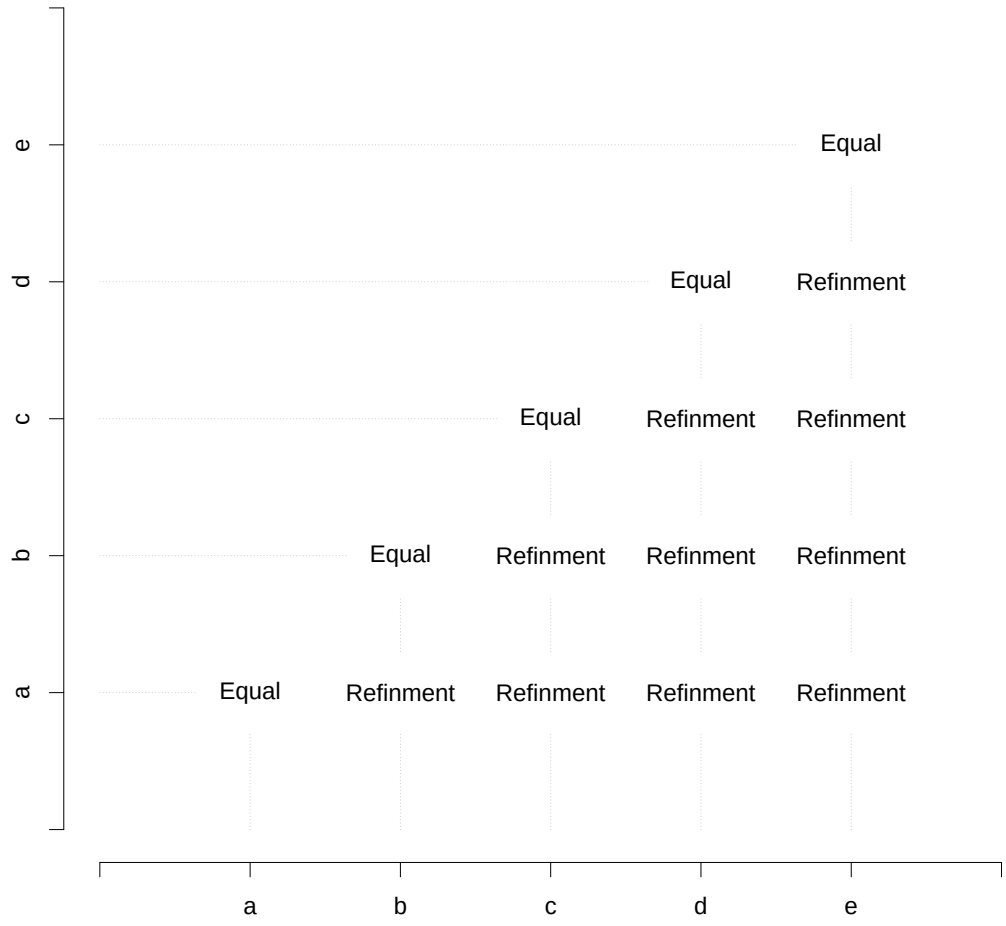


**Figure 18:** *Comparison of the segmentations produced in the different runs, for the collection $C_2$.*

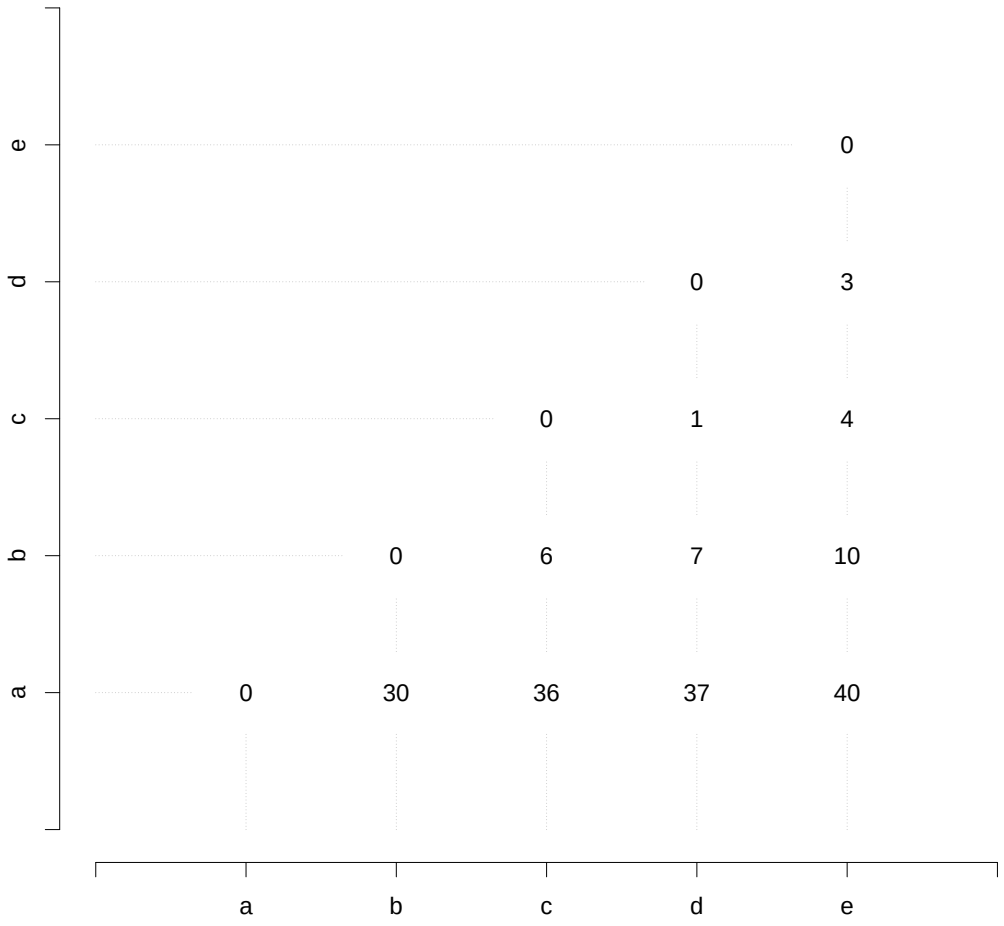


**Figure 19:** *Distance of the segmentations produced in different runs, for the collection $C_2$.*

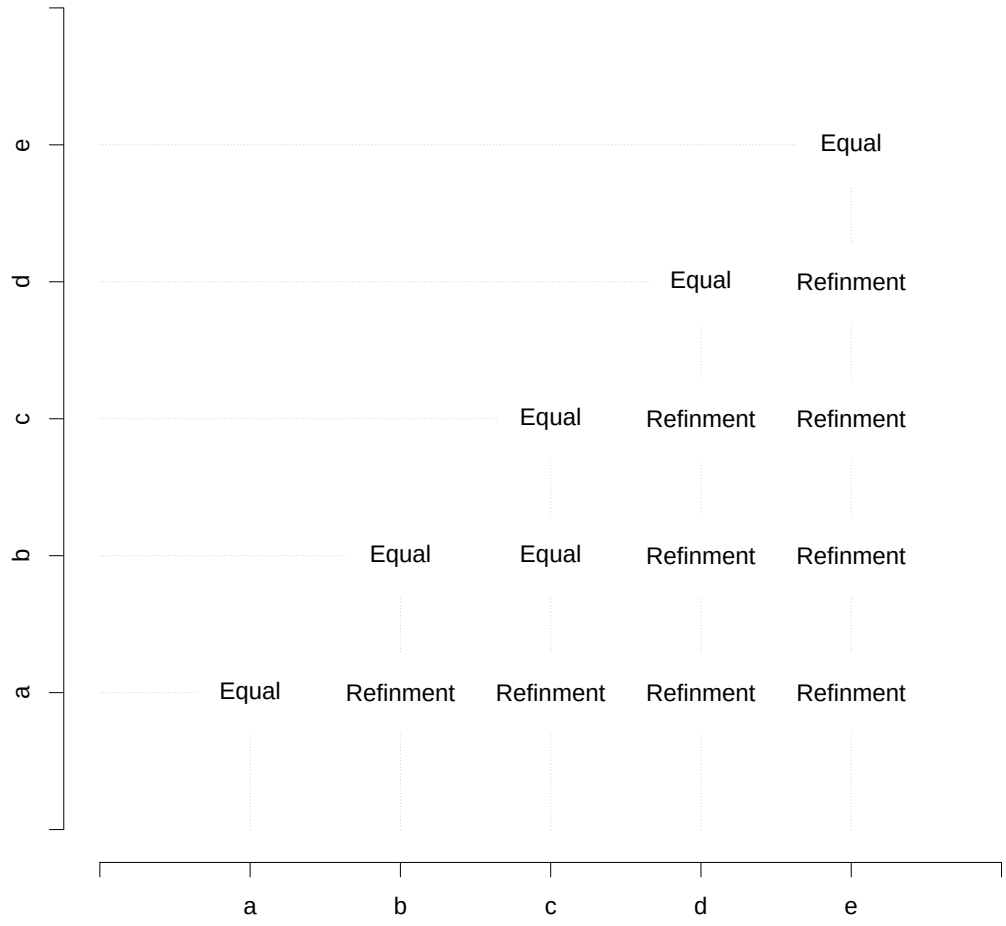


**Figure 20:** *Comparison of the segmentations produced in the different runs, for the collection $C_3$.*

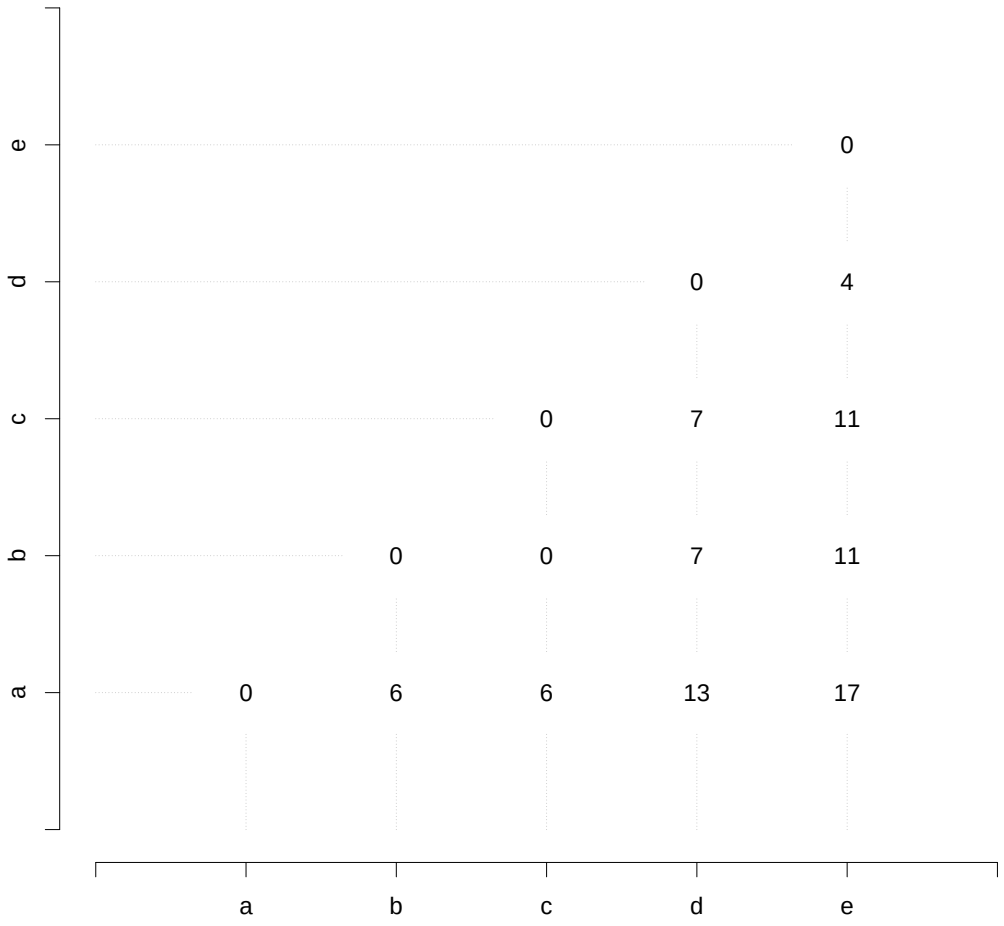


**Figure 21:** *Distance of the segmentations produced in different runs, for the collection $C_3$.*

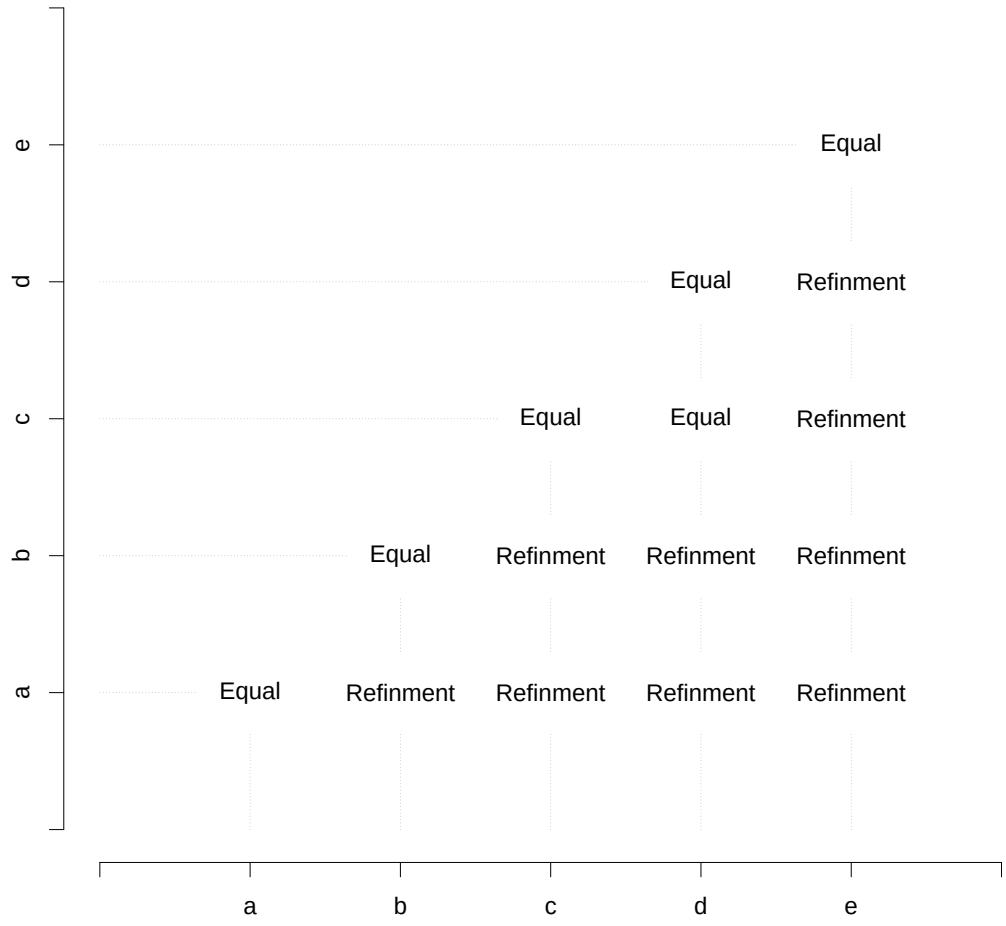


**Figure 22:** *Comparison of the segmentations produced in the different runs, for the collection* $C_4$.

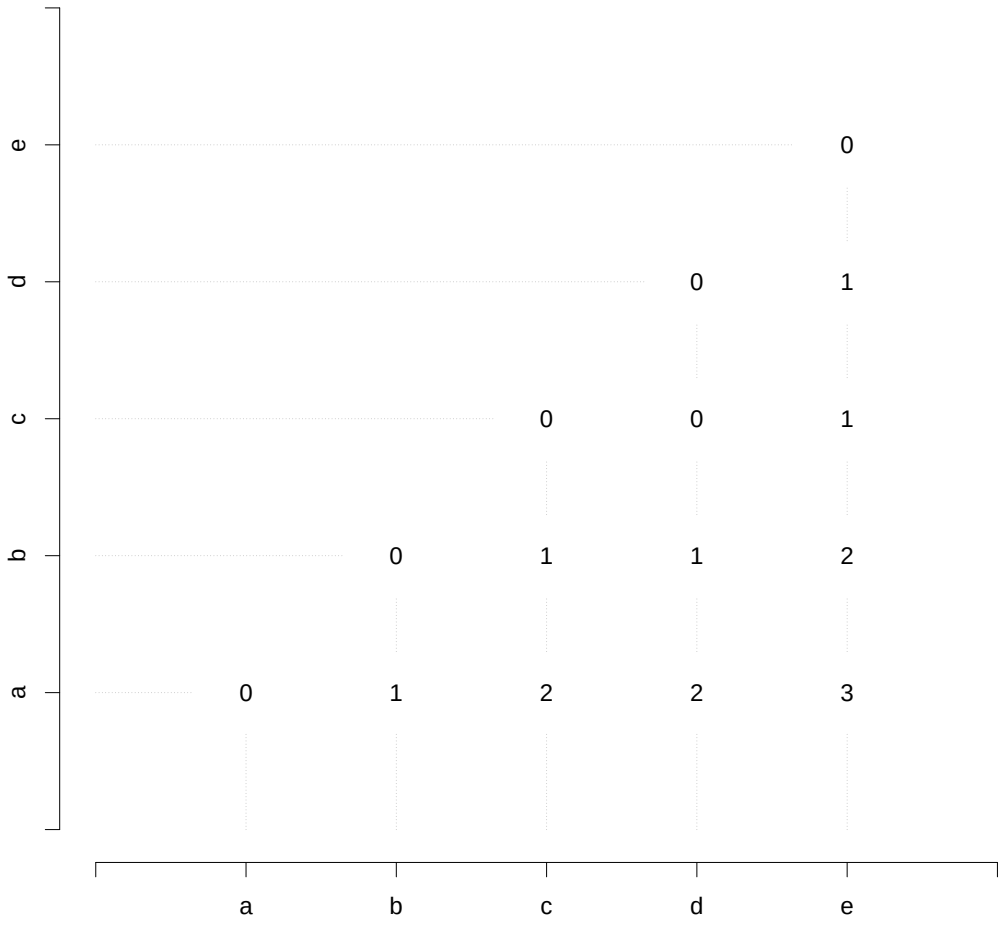


**Figure 23:** *Distance of the segmentations produced in different runs, for the collection $C_4$.*

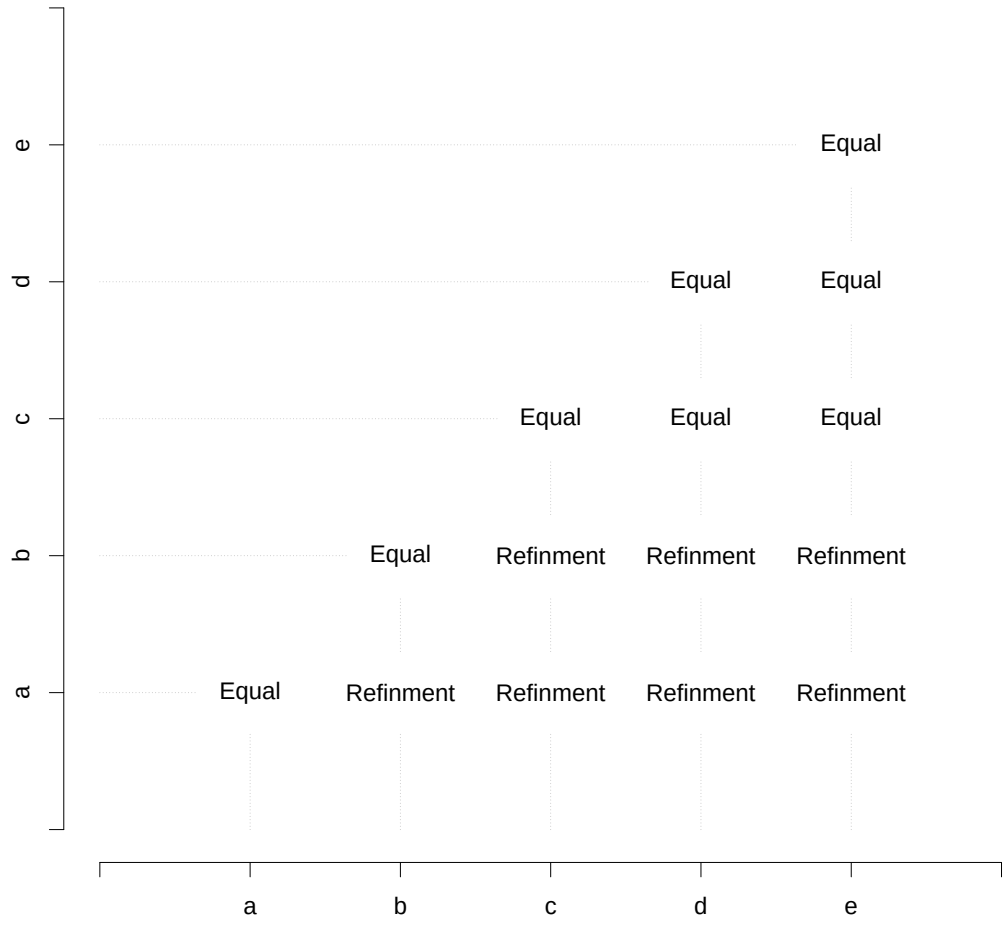


**Figure 24:** *Comparison of the segmentations produced in the different runs, for the collection $C_5$.*

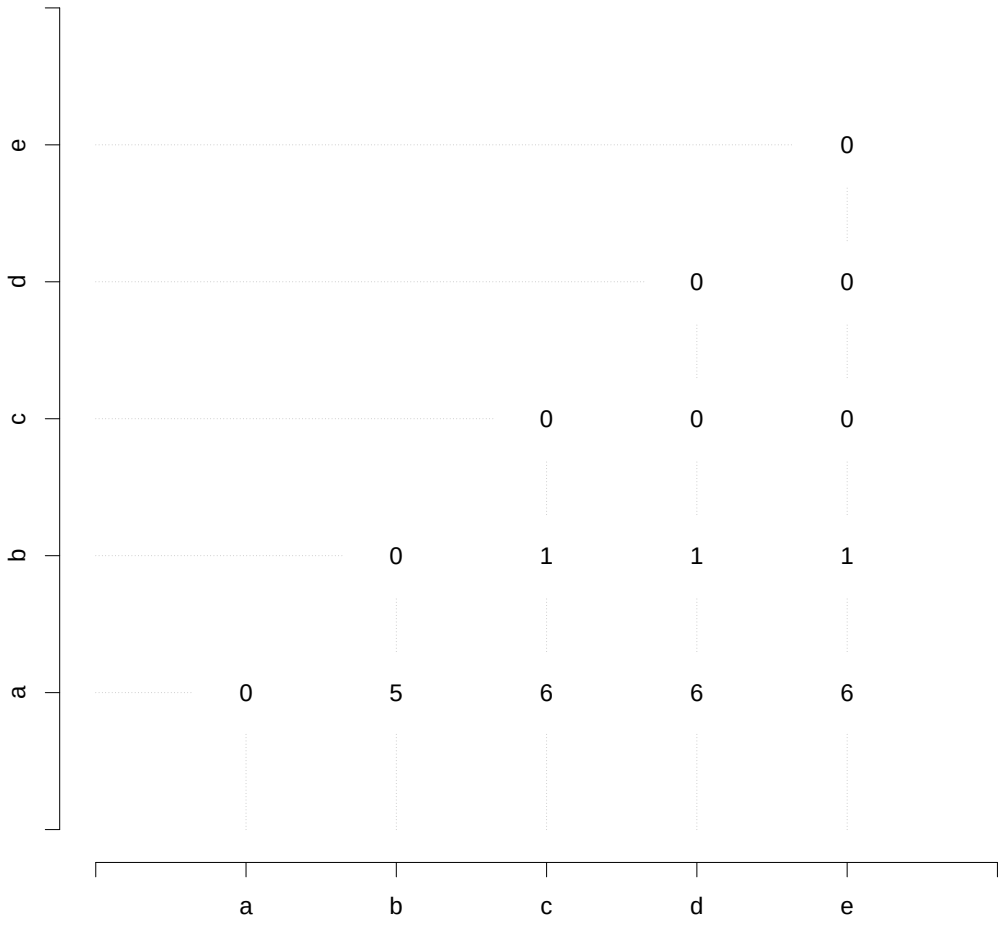


**Figure 25:** *Distance of the segmentations produced in different runs, for the collection $C_5$.*