UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE

"NEWSPAPER MAGNIFICATION PRESERVING ENTRY POINTS AND OPTIMIZING AESTHETICS: A COMPUTATIONAL APPROACH OF LAYOUTING USING AN EVOLUTIONARY ALGORITHM."

SEBASTIÁN ALEJANDRO GALLARDO DÍAZ

TÉSIS PARA OPTAR AL GRADO DE
MAGISTER EN CIENCIAS DE LA INGENIERÍA INFORMÁTICA

Profesor Guía: Dra. María Cristina Riff Rojas
Profesor Correferente:

January - 2023

DEDICATORIA

A mis padres, Richard y Alejandra.

AGRADECIMIENTOS

Comienzo agradeciendo a quienes me dieron la oportunidad de trabajar en este proyecto con gran impacto social: Dorian Mazauric y Pierre Kornprobst, investigadores del centro *Inria Sophia Antipolis - Méditerranée*, quienes continuamente me dieron su *feedback* y toda la ayuda posible para lograr los objetivos propuestos. También a la profesora María Cristina Riff que aportó con su valioso y vasto conocimiento del área, lo cual me ayudó a concretar este trabajo.

Luego, me gustaría agradecer a toda mi familia y amigos por el apoyo constante. Un especial agradecimiento a mis amigos Jonathan, Makarena, Erick, Katherine, Christian, Álvaro, Nicolas, Javier, Mauricio, Aldo, Ignacio, y a todos mis amigos de infancia por las risas y apoyo constante. No me olvido tampoco de mis queridos compañeros de universidad Ignacio, Macarena, Bastián, Ariel y Claudio, tanto por el trabajo realizado en conjunto como el apoyo emocional en los momentos más dificiles. No me olvido de mis amigos virtuales, la comunidad del sitio *Colemono*, quienes apañaron más de una tarde o noche de arduo trabajo, en especial a Sebastián, Cristian y Francisco.

Muchas gracias a todos mis profesores en este largo trayecto, en especial a Cecilia Reyes, quien siempre tuvo completa disposición de ayuda con cada uno de los estudiantes de Informática UTFSM.

A mis queridos padres: Richard y Alejandra, quienes no dudaron un segundo en posponer sus propios intereses por los míos y me permitieron soñar con estudiar en aquel castillo frente al Pacífico.

Finalmente, a Constanza, compañera de aventuras durante toda mi etapa universitaria que aun no acaba, que me apoyó en los momentos más duros y estuvo presente, y es en parte responsable, de todos mis logros tanto académicos como personales. Un beso y abrazo gigantes.

Seamos la pesadilla de los que pretenden arrebatarnos los sueños
Ernesto *Che* Guevara

RESUMEN

Resumen— La prensa escrita ha intentado adaptar su producto principal, el periódico, a las nuevas tecnologías desde el surgimiento de las mismas. La mayoría de los dispositivos de lectura digitales ofrecen herramientas para mejorar la accesibilidad tales como magnificadores (zooming). Magnificación usando pinch and zoom es la principal solución disponible; sin embargo, este método resulta en perder muchos entry points, concepto usado para denominar a los puntos importantes de la página como las imágenes y los titulares. El problema de accesibilidad descrito motiva este trabajo de tesis.

En este trabajo de tesis, se desarrolla una solución para crear nuevos layouts de solo texto (redistribución de artículos en una página) que permite magnificar solo la fuente del texto, garantizando una buena visualización de todos los titulares. Esto requiere explorar nuevas relaciones de aspecto y/o posiciones de cada artículo en la página, ya que se intenta minimizar la cantidad de titulares que requieren más de 3 líneas de texto para el tamaño de fuente. Esto constituye nuestra medida cuantitativa de accesibilidad. De manera adicional, se busca optimiza la estética del resultado.

Se implementa lo anterior con un algoritmo genético. Se realizaron pruebas usando instancias generadas a partir de páginas reales de periódico. Se obtuvieron diferentes layouts donde la fuente fue magnificada con un factor de 2.0. Se puede ver que la calidad de los titulares es globalmente mucho mejor, y la distribución de los artículos en la página genera mejores valores de estética.

Para trabajo futuro se espera generalizar este método prometedor, considerando la complejidad de periódicos reales, con la calidad de experiencia de usuario en mente como principal objetivo.

Palabras Clave— Layout, Diseño, Estética, Periódico, artículo, algoritmo

ABSTRACT

Abstract— The digital era is completely transforming the newspaper industry, offering new user experience for all readers. Most of the reading devices are currently offering new user experience for all readers, including some vision tools, such as magnifiers, that we can use to improve overall accessibility to text and news media. Magnification by pinch and zoom is the primary solution currently available; however, this method results in losing some of the entry points, concept used to name some of the key objects in a newspaper page, like images and headlines. This accessibility problem inspire this thesis.

In this thesis, we designed and implemented a novel solution to create new designs with only-text articles, magnifying only font size, allowing a nice visualization of all headlines. This solution requires the exploration of new aspect ratios and/or positioning for each article, because we want to minimize headlines that need more than three lines for a given font size. This constitutes our accessibility measure. Additionally, another feature of our redesigning method is to optimize the aesthetics.

Our approach consists in a relayouting approach implemented via a genetic-inspired approach. We tested it on realistic newspaper pages. For the case discussed here, we obtained many different layouts where the font was increased by a factor of two. We show that the quality of headlines is globally much better with the new layouts than with the original layout. Future work will tend to generalize this promising approach, accounting for the complexity of real newspapers, with user experience quality as the primary goal.

Keywords— Layout, Newspaper Design, Aesthetic, Newspaper, article, genetic algorithm

GLOSSARY

BPP: *Bin Packing Problem*
SPP: *Strip Packing Problem*
2DSPP: *2D Strip Packing Problem*
2DBPP: *2D Strip Packing Problem*
1DBPP: *1D Bin Packing Problem*
MMKP: *Multidimensional Multiple-Choice Knapsack Problem*
MCKP: *Multiple-Choice Knapsack Problem*
MDKP: *Multidimensional Knapsack Problem*
TS: *Tabu Search*
SA: *Simulated Annealing*
GA: *Genetic Algorithms*

Contents

List of Figures

List of Tables

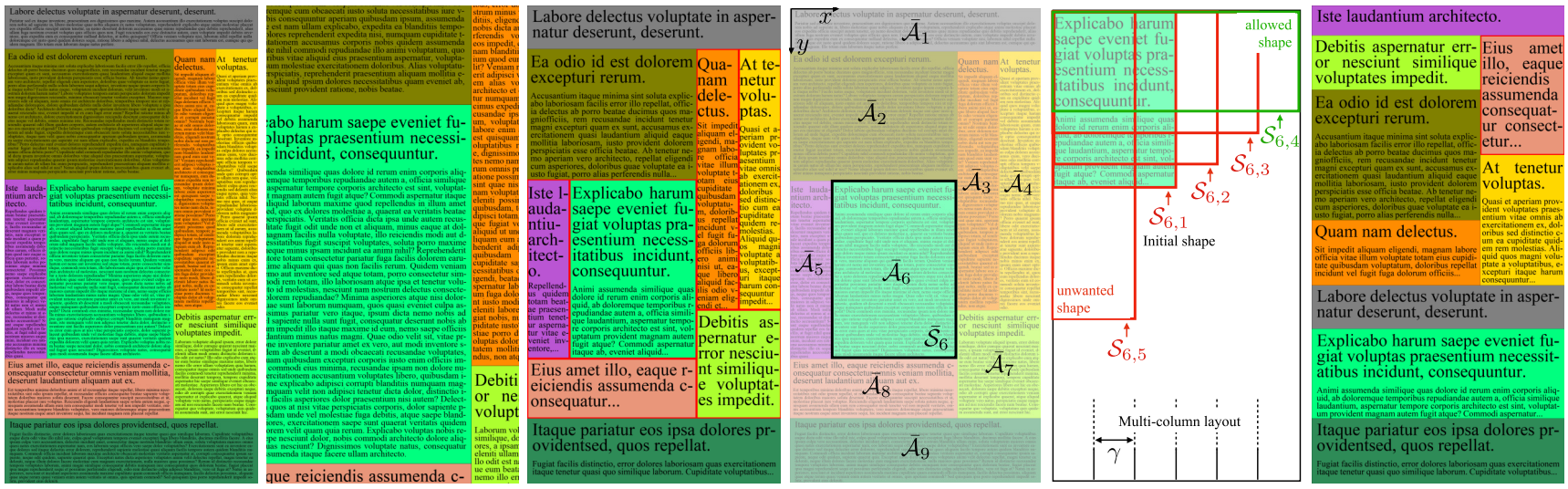
INTRODUCTION

While there has been a lot of studies and debates around print versus online newspapers, print newspapers continue to be a point of reference in terms of quality and user experience [Fortunati *et al.*, 2015]. This attachment we have for print can be explained mainly by the beauty of their design, which provides a strong, consistent, and appealing visual approach to enhance a story's impact and comprehension [Gautier and Gautier, 2017, Errea and Gestalten, 2018] but also promotes reader engagement [Morell, 2016]. However, combining the design and aesthetics from the printed edition and the functionalities of the online edition remains a complex design challenge [Ihlström *et al.*, 2004, Hollander *et al.*, 2011]. In this sense, we need to go beyond the limits of the two main current modes of access to the digital press, which are: (1) access to an electronic version of the paper edition, on a reader or tablet, through digital kiosk applications (access to more or less enhanced PDF files, in general); (2) access to a web version, adaptable to various screen sizes thanks to "responsive design", which essentially gives us the possibility to read articles but missing out on the print experience.

The print experience is indeed very particular: Numerous behavioral studies have shown that a newspaper is not read linearly and exhaustively. In [Garcia and Stark, 1991], the authors have highlighted the existence of *entry points* (e.g., photos, illustrations, headlines) that provide readers with the necessary elements to scan a page and select what they want to read. As such, these entry points are essential in the print user experience. Since online newspapers have been fundamentally a simple transposition of the print newspaper into digital media, one might think that the print experience should the same. Unfortunately, this simple reasoning does not work simply because the print newspapers are not designed to be read on small displays. To convince yourself, let us consider here the case of headlines which will be our focus in this thesis. When the headlines become less readable (because their font is too small), they can no longer play their role as a hook to make us want to read a particular article (see Fig.1(a)). So there is a need for magnification. One classical option in kiosk applications is to zoom in on the content thanks to intuitive touch screen gestures (e.g., pinch-to-zoom). However, this type of magnification has a significant drawback as we quickly lose our positioning inside the page (Fig. 1(b)), which impacts reading comfort and navigation significantly.

In this work, we aim to find a solution to approximate the print experience online, i.e., enable this scanning and selection procedures that define the print reading experience. We present a new approach to transform a print newspaper layout to fit small displays while preserving entry points. In this work we focus our attention on headlines. Our goal will be to guarantee that headlines should be readable (referring to the font size) and usable (referring to the

number of lines to split the headline string into). Concerning this last property, note that most newspaper headlines use one or two lines –sometimes three or four, which is related to the easiness of reading the headline. The challenge is that these two properties are opposite since increasing the font size makes headlines split into more lines (Fig. 1(c)). To tackle this problem, we propose a method that explores alternative shapes for articles to generate new layouts where headlines will be readable and usable. Moreover, we also want to include the notion of aesthetics of the layout, based on some existing aesthetic criteria.



(a) (b) (c) UW=6 (d) (e) (f) UW=1

Figure 1: From print to online newspapers on small displays: (a) Print newspaper page where each article has been colored to highlight the structure. On small displays, headlines become too small to be readable and usable for navigation. Magnification is needed. (b) Pinch-zoom result with a magnification factor of two: Illustrates the common local/global navigation difficulty encountered when reading newspapers via digital kiosk applications. (c) Increasing the font with a factor two, keeping the original layout: Articles with unwanted shapes (UW with $\lambda_{max} = 3$, see are boxed in red; (d) Original page with articles numbered. (e) Alternative shapes for article $\bar{\mathcal{A}}_6$ using the multicolumn grid (suggested at the bottom), where colors indicate whether the shapes are allowed (in green) or unwanted (in red). For article $\bar{\mathcal{A}}_6$ the only allowed shape is the one occupying all page width. (f) Proposed layout showing how the page in (a) has been transformed to preserve headlines quality.

This thesis is organized as follows. Chapter describes the formalities of our problem. Chapter 1.2 presents a brief survey of related problems, like *Strip Packing*. Chapter 2.6 presents our approach's main lines, which is inspired by genetic algorithms. Section 3.4 shows how it works in a realistic case, together with quantitative analysis. Section 4.8 concludes this work by discussing the potential of this approach.

Objectives

General Objective

- Generate new layouts from printed version of newspaper, considering a magnification factor of the font size, in order to approximate online reading to the print reading experience, focusing in headlines.
- Include aesthetics measures to refine the generated layout.

Specific Objectives

- Design an evolutionary algorithm, to obtain new layouts that imitates print reading experience.
- Include aesthetic measures to obtain refined layouts focusing on headlines.

CHAPTER 1
PROBLEM DEFINITION

Let us consider a newspaper page specified by a multi-column layout and denote by γ the width of the columns (Fig. 1(e)). This page is made of a set of N articles $\{\bar{\mathcal{A}}_i\})_{i=1..N}$ defining the original layout $\bar{\mathcal{L}}$ (Fig. 1(d)), which is denoted by the symbol $\bigoplus$:

$$\bar{\mathcal{L}} = \bigoplus_{i=1..N} \bar{\mathcal{A}}_i.$$

Note that bars over letters refer to the original layout. By construction, these articles define a partition of the page. Each article $\bar{\mathcal{A}}_i$ is defined by (i) its shape $\bar{\mathcal{S}}_i$, which has dimensions $(\bar{w}_i, \bar{h}_i)$, defining an aspect ratio $\bar{\lambda}_i = \bar{h}_i/\bar{w}_i$, (ii) its position here defined by the upper-left corner position $(\bar{x}_i, \bar{y}_i)$, and (iii) its content. In this work we assume that the content will always consist in a headline and just body text for the main body.

Then it's important to introduce an important notion of *allowed* versus *unwanted* shapes. Given a shape $\mathcal{S}$, let us first denote by the function $\lambda(\mathcal{S})$ the number of lines a headline is split into (concerning how this number of line is estimated, note that here we chose justified headlines in all cases, but any other choice would work). By definition, a shape $\mathcal{S}$ will be either *allowed* if $\lambda(\mathcal{A}) \leqslant \lambda_{max}$ or *unwanted* otherwise. As shown later, the number of unwanted shapes (UW) will be essential in evaluating a layout since it will provide information about the headline's usability.

Given these definitions, we can define the problem as follows: Consider that a reader wishes to increase the font size by a magnification factor of α (e.g., to compensate for a visual acuity limitation that prevents him or her from reading the headlines correctly), is the original layout satisfactory (i.e., no unwanted shapes) or does it exist alternative better layouts (with less unwanted shapes)?

Answering the first question is easy. If we simply increase font size using the original layout, most headline will naturally flow on many lines (see, e.g., Fig. 1(c) with six unwanted shapes out of nine). So we need to find alternative layouts. Indeed, it is required to discard unwanted shapes, but when we change one shape, this generates a domino effect needing a complete re-layout of the page by exploring alternative shapes for all articles (even if they were not *a priori* impacted by the magnification).

To be more precise, we define notion of alternative shapes. For an article $\bar{\mathcal{A}}_i$, we define N_i alternative shapes $\{\mathcal{S}_{i,j}\}_{j=1,\ldots,N_i}$ that are the shapes having the same area as the initial one, widths being multipliers of γ (adjusting the height in a way that preserves the article's area), and of course fitting into the page. Note that we will use $j = 1$ for the original shape (i.e.,

$\mathcal{S}_{i,1} = \bar{\mathcal{S}}_i$). For example, Fig. 1(e) shows the five possible shapes for $\bar{\mathcal{A}}_6$ (including the original one). Note that the initial position should not be considered to build alternative shapes (they should only fit the page).

Given this set of all possible shapes $\mathcal{S}_{i,i}$ for each article i, the solution space, i.e., the combinations of shapes leading to good alternative layouts (i.e., no overlapping articles) is:

$$\mathcal{E} = \{\{\mathcal{S}_{i,j}\}_{i=1..N} \quad / \quad \exists \quad \mathcal{L} = \bigoplus_{i=1..N} \mathcal{A}_{i,j}\},$$

where $\mathcal{A}_{i,i}$ is derived from the original article $\bar{\mathcal{A}}_i$ and is defined by (i) the alternative shape $\mathcal{S}_{i,j}$, (ii) a new position $(x_{i,j}, y_{i,j})$, (iii) a content that is cut because of the magnification (normally concerns only body text for reasonable magnification factors). Note that we target a digital experience so that the full body text will still be accessible once an article is chosen.

Now the problem becomes to find the best layout, i.e., the best combination of shapes for each article, with a minimal number of unwanted shapes, thus leading to the best alternative layout $\mathcal{L} \in \mathcal{E}$. For this, we define an energy function E:

$$\max_{\mathcal{L}=\bigoplus \mathcal{A}_{i,j} \,/\, \{\mathcal{S}_{i,j}\} \in \mathcal{E}} E(\mathcal{L}) = \Omega(H(\mathcal{L}), F(\mathcal{L}))$$

where term $H(.)$ is related to the number of lines of each headings in chosen shape configurations. It penalizes strongly shapes that elicit headlines flowing on high number of lines ($> \lambda_{max}$). The term $F(.)$ is related to the aesthetics of the layout. It is based on three criteria: alignment, regularity and balance (taking some inspiration from [David Chek Ling Ngo, 2000, Harrington *et al.*, 2004].).

1.1 Mathematical Model

In this section a mathematical model is presented.

The model definition is based on this proposed by Baldi et al.[Baldi *et al.*, 2012] for the knapsack problem in multiple dimensions.

Variables

- z_{ij}: 1 if $\mathcal{S}_{i,j}$ is chosen for article $\bar{\mathcal{A}}_i$, 0 otherwise
- (x_{ij}, y_{ij}): bottom-left corner coordinates of $\mathcal{S}_{i,j}$.
- a_{ij}^{kl}: 1 if $\mathcal{S}_{i,j}$ is packed before $\mathcal{S}_{k,l}$ in x-axis. In other words, if the right side of $\mathcal{S}_{i,j}$ has a x-coordinate lesser than the left side of $\mathcal{S}_{k,l}$.

- a_{ij}^{kl}: 1 if $\mathcal{S}_{i,j}$ is packed before $\mathcal{S}_{k,l}$ in y-axis. In other words, if the top side of $\mathcal{S}_{i,j}$ has a y-coordinate lesser than the bottom side of $\mathcal{S}_{k,l}$.

Parameters

- (W, H): dimensions of newspaper page
- γ: Width of the columns of the multicolumn layout imposed over $\mathcal{L}$.
- (w_{ij}, h_{ij}): dimensions of $\mathcal{S}_{i,j}$.
- M: $\sim \infty$.
- N: Number of articles
- N_i: Number of alternative shapes for $\bar{\mathcal{A}}_i$.

Constraints

- Don't exceed page dimensions

$$x_{ij} + w_{ij} \leqslant W \qquad \forall j \in \{1, 2, ..., N_i\} \quad \forall i \in \{1, 2, ..., P\} \tag{1}$$
$$y_{ij} + h_{ij} \leqslant H \qquad \forall j \in \{1, 2, ..., N_i\} \quad \forall i \in \{1, 2, ..., P\} \tag{2}$$

(1) constrains width to be less than page width , and (2) the same for the heights.

- No *overlapping*

$$a_{ij}^{kl} + a_{kl}^{ij} + b_{ij}^{kl} + b_{kl}^{ij} \geqslant z_{ij} + z_{kl} - 1 \quad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{3}$$

$$x_{ij} + w_{ij} \leqslant x_{kl} + M(1 - a_{ij}^{kl}) \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{4}$$

$$x_{kl} + w_{kl} \leqslant x_{ij} + M(1 - a_{kl}^{ij}) \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{5}$$

$$y_{ij} + h_{ij} \leqslant y_{kl} + M(1 - b_{ij}^{kl}) \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{6}$$

$$y_{kl} + h_{kl} \leqslant y_{ij} + M(1 - b_{kl}^{ij}) \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{7}$$

(3) constrains overlapping between 2 pieces (*overlapping*): following a and b definitions, $a_{ij}^{kl} = a_{kl}^{ij} = b_{ij}^{kl} = b_{kl}^{ij} = 0$ shows, if both shapes were packed, that we have *overlapping*

between them. (4) constraints that, if s_{ij} is packed before s_{kl}, $x_{ij} + w_{ij}$ doesn't exceed x_{kl}. (5) follows the same principle, but when s_{kl} is packed before s_{ij}. Then, (6) y (7) follows the same idea in y-axis.

- If s_{ij} is not packed, $(x_{ij}, y_{ij}) = (0, 0)$

$$x_{ij} \leqslant M z_{ij} \qquad \forall i \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \tag{8}$$
$$y_{ij} \leqslant M z_{ij} \qquad \forall i \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \tag{9}$$

- If s_{ij} is not packed, $a_{ij}^{kl} = 0$ y $b_{ij}^{kl} = 0$

$$a_{ij}^{kl} \leqslant z_{ij} \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{10}$$
$$a_{ij}^{kl} \leqslant z_{kl} \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{11}$$
$$b_{ij}^{kl} \leqslant z_{ij} \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{12}$$
$$b_{ij}^{kl} \leqslant z_{kl} \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\} \tag{13}$$

- Only one shape per article

$$\sum_{j=1}^{N_i} z_{ij} = 1 \qquad \forall i \in \{1, 2, ..., P\} \tag{14}$$

- Variable Domain

$$z_{ij}, a_{ij}^{kl}, b_{ij}^{kl} \in \{0, 1\} \qquad \forall i, k \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\} \quad \forall l \in \{1, 2, ..., N_k\}$$
$$x_{ij} \in \{0, \delta_v, 2\delta_v, ..., (N_{strips} - 1)\delta_v\} \qquad \forall i \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\}$$
$$y_{ij} \in \{0, \delta_h, 2\delta_h, ..., (N_{strips} - 1)\delta_h\} \qquad \forall i \in \{1, 2, ..., P\} \quad \forall j \in \{1, 2, ..., N_i\}$$

Objective Function

$$\max_{\mathcal{L}=\bigoplus \mathcal{A}_{i,j} \,/\, \{\mathcal{S}_{i,j}\} \in \mathcal{E}} E(\mathcal{L}) = H(\mathcal{L}) + F(\mathcal{L})$$

1.2 Chapter Conclusions

One of the differences of the problem presented in this chapter respect to others like *Strip Packing* is that pieces (articles in this case) can have any width and height, as soon as size is preserved. To describe it precisely and in a discrete way, an horizontal grid was imposed, and all possible *alternative shapes* were previously generated and listed. With this, the problem becomes a mixture between *Multidimensional Multiple-Choice Knapsack Problem*

and *Strip/Bin Packing*. Another aspect that differentiate this problem is that corresponds to a *single-and-closed container packing*.

It is important to mention that even if it is presented as a packing problem, accessibility (related to magnification, entry points and number of lines, as explained) and aesthetics are the main objective of any method to solve it. That's why the objective function is presented as a function of $H(\mathcal{L})$ and $F(\mathcal{L})$ terms.

CHAPTER 2
RELATED WORK

2.1 Packing Problems

2D Packing Problems are combinatorial optimization problems with a lot of applications, like cutting of raw-material or item packing. In all applications, the focus is positioning of figures with known dimensions inside bigger containers.

If the objective is to minimize the number of used containers to pack all the pieces, then is an instance of a *Bin Packing Problem*. On the other side, if there is only one container, with infinite height and fixed width, and the objective is to pack all pieces minimizing total height of the strip, then is an instance of a *Strip Packing Problem* (SPP). Additionally, is interesting to study a variant of the knapsack problem; *Multidimensional Multiple-Choice Knapsack Problems* where we want to choose one piece of each type or group, to put inside a multiple dimension knapsack. The packing problem definition of the problem in this work is based on a combination of this problem and a packing problem.

Packing problems can be further characterized, for example, using Wäscher et al. typology [Wäscher *et al.*, 2007]. Not only the dimensionality of the problem and the geometry can be taken into account, but further characteristics regarding the way how the rectangles are laid out on the strip can be considered for the problem categorization (Fig. 2).

- *Offline/Online*: In the online version, rectangles are input one by one without any information regarding the following rectangle. Once the rectangle is placed on the strip there is no possibility of repositioning it. Not knowing which will be the next rectangle arises for instance in scheduling problems, when a sequence of tasks or jobs have to be processed in a given number of machines but the tasks are not known in advance. However, the most frequent in the literature (and in real-world applications) is the offline case. In this situation it is possible to define criteria for the order by which the rectangles will be placed on the strip, as length, width, area or perimeter, once all characteristics of the rectangles are known in advance.[Oliveira *et al.*, 2016a].
- *Regular/Irregular*: Usually, the pieces are rectangular-shaped with width and height. However, in practical contexts there are irregular items like polygons, circles or other types.
- *Dimensionality*: If the problem is with 1D, 2D or 3D items or pieces.

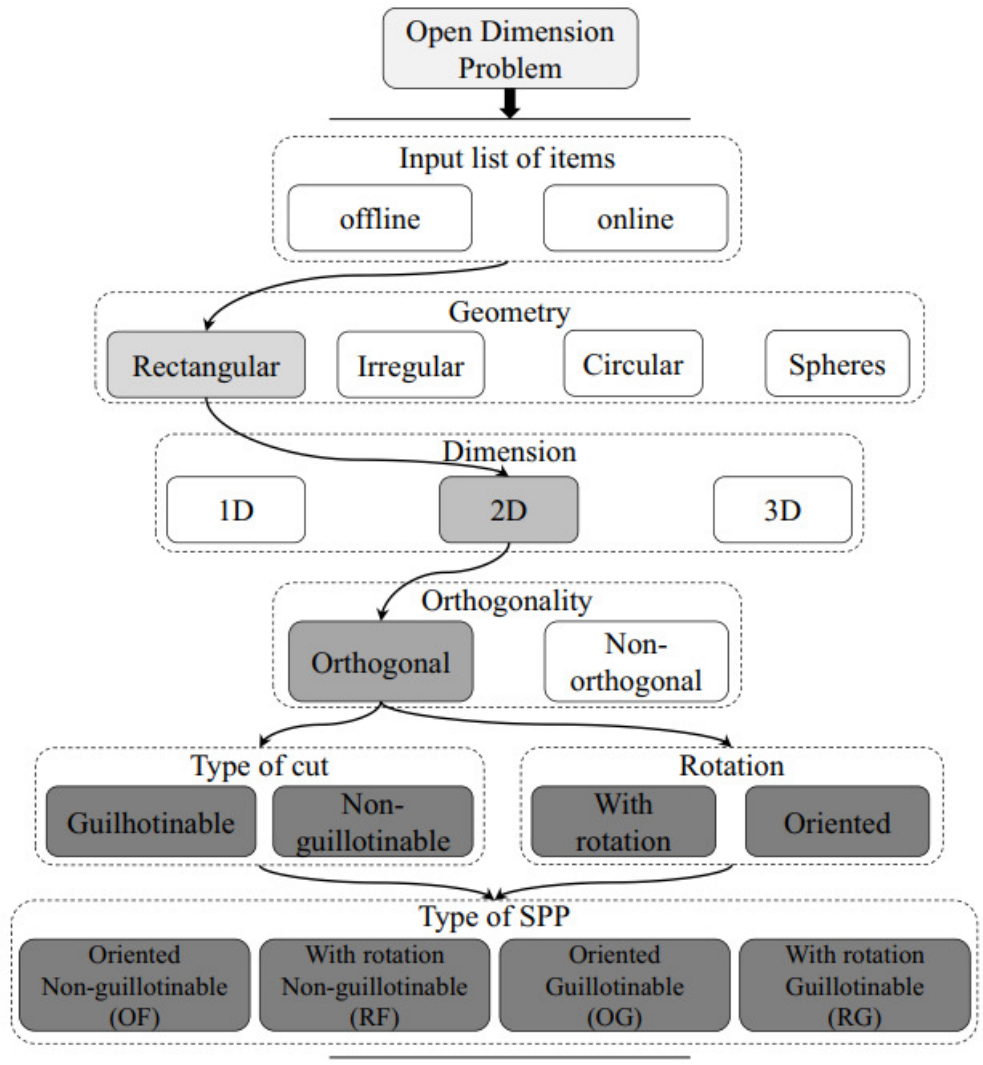


Figure 2: Classification proposed by Wäscher et al.[Wäscher *et al.*, 2007]

- *Orthogonality*: In orthogonal packing the edges of the rectangles are parallel to the edges of the container.

- *Guillotinable*: a cutting pattern is guillotinable if it is possible to obtain the requested rectangles by cutting from edge to edge, either the strip's edges or the edges of the sub-rectangles generated by previous cuts. In fact the name arises from the applications where the cutting tool is a guillotine [Oliveira *et al.*, 2016a]. Also, if it is enough to cut the container horizontally or vertically in n levels, then it is *one stage guillotinable*. if it is necessary to rotate after the first level of cuts, and apply a second stage of cuts, then it is *two stage guillotinable* (and so on for *n-stage guillotinable* [Imahori and Yagiura, 2010]). See Fig. 3.

  Patterns that do not have this property are designed as non-guillotinable patterns.

- *With/without rotation*: If it is allowed to rotate 90 degrees any item before packing, then it is a packing problem with rotation. In other case, it is a packing problem without rotation.

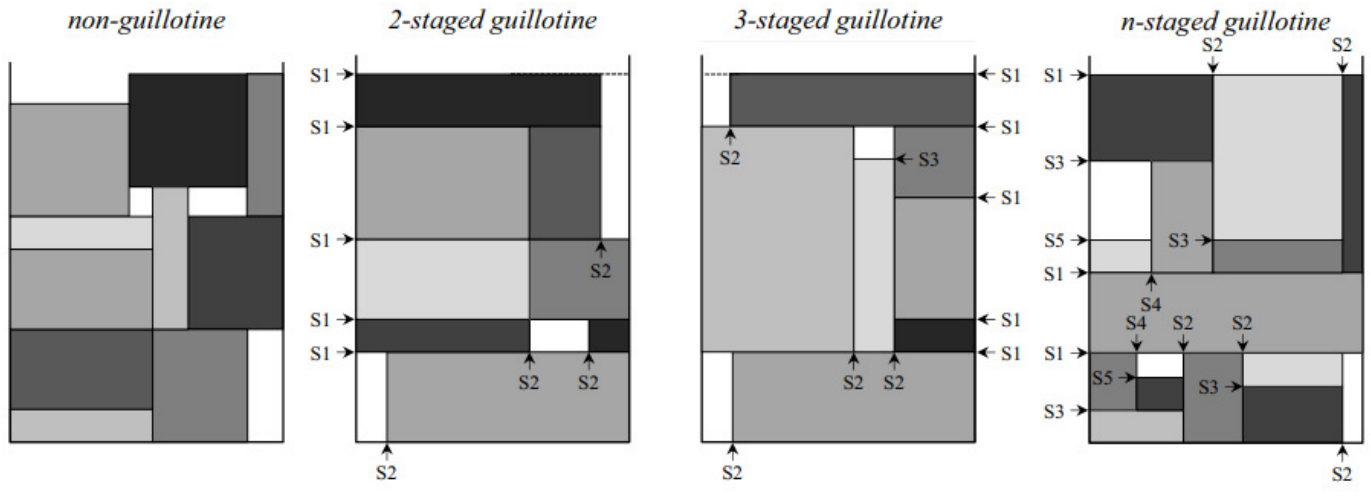


Figure 3: Examples of guillotinable cuts[Oliveira *et al.*, 2016a]

2.2 Strip Packing Problem (2DSPP)

The Strip Packing Problem (SPP) aims to pack a set of n *items* (in the scope of this work, rectangles with width w_i and height h_i) inside a larger object , the container, with fixed width and unlimited height, with the objective of minimizing the free dimension of the large object. The small items cannot overlap each other and must be completely inside the large object. It is an Open Dimension Problem according to Wäscher et al. typology [Wäscher *et al.*, 2007]. It is similar to Bin Packing Problem, which aims to minimize the number of fixed dimension containers to pack all pieces. Also, both of them are *NP-hard*, because there are generalizations in more dimensions of the 1D-Bin packing problem and the knapsack problem.

2.2.1 Programming Model for the 2DSPP

Riff et al. [Riff *et al.*, 2009] propose the following programming model.

For a set of N rectangular*items*:

- x_i, y_i: coordinates of the *bottom left* corner of item $i \in N$.
- h_i, w_i: Parameters. Height and width of rectangular item i.
- W: Parámeter. Fixed width of the *strip*.

The objective is

$$\min H = \max_i (y_i + h_i)$$

subject to

$$x_i + w_i \leqslant W, \qquad \forall i \in N \tag{15}$$

$$y_i + h_i \leqslant H, \qquad \forall i \in N \tag{16}$$

$$\begin{aligned} x_i + w_i \leqslant x_j \vee x_j + w_j \leqslant x_i \qquad & \vee \\ y_i + h_i \leqslant y_j \vee y_j + h_j \leqslant y_i, \qquad & \forall i, j \in N, i \neq j \end{aligned} \tag{17}$$

$$x_i + y_i \geqslant 0, \qquad \forall i \in N \tag{18}$$

(15) and (16) define restrictions to keep each item inside the strip. Also, (17) imposes *non-overlapping*. Finally, (18) allows only positive positions.

2.2.2 Linear Programming methods

Lodi et al.[Lodi *et al.*, 2002] propose a linear programming model for a simpler version of the problem, called 2D Linear Strip Packing, where pieces are packed by levels, in an iterative way:

$$\min \sum_{i=1}^{n} h_i y_i$$

s.a

$$\sum_{j=1}^{j-1} x_{ij} + y_j = 1, \qquad \forall j = \{1, ..., n\} \tag{19}$$

$$\sum_{j=i+1}^{n} w_j x_{ij} \leqslant (W - w_i) y_i, \qquad \forall i = \{1, ..., n-1\} \tag{20}$$

$$y_i, x_{ij} \in \{0, 1\} \qquad \forall i, j \tag{21}$$

This model allows to solve the problem using solvers.

Many authors proposed exact methods using *Branch and Bound* or Dynamic Programming techniques. However, in many instances it is focus in *perfect packing* instances (with no

empty spaces) and low number of pieces (less than 30), and still obtaining long execution times.

Given that issue, heuristic methods are the most-used tool to solve *Strip Packing*.

2.2.3 Positioning-based heuristics

Positioning-based heuristics are the oldest in the SPP literature, and most common in the early works. They are rather flexible, allowing to incorporate the most usual constraints of the problem. The basic mechanism behind positioning-based heuristics is the identification of the free space on the strip that is most suitable for a given piece.

- *Bottom Left*: Baker et al. [B.S. *et al.*, 1980] introduced this heuristic which tries to identify an empty space for a specific piece. Just as suggested by the name, tries to fit the piece in an initial position and proceed to move the piece to the bottom and the left, alternating between these moves until find final position. The main advantage of the bottom-left strategy is being very fast (With complexity $O(n^2)$).

- *Bottom Left Fill*: Chazelle [B., 1983] approach search just as BL, but allows to look in any free corner, even holes formed by pieces around (See Fig. 4). This increases the search space, leading to (in general) worse execution times, but allows to fill holes presents in the container. The implementations of the bottom-left-fill heuristic places the rectangle on a position that is not feasible, because it overlaps other rectangles that had been already placed (e.g. at the bottom-left corner of the strip), and then moves the rectangle right and up until a feasible position is found.

- *improved Bottom Left*: Liu and Teng [D. and H., 1999] proposed *improved BL*, which suposses a BL-improvement, gives always priority to the down movement, i.e. when it is not possible to move the rectangle downwards it moves it to the left, but just the distance necessary to make again a downward movement.

- *Priority Heuristic*: Most of the positioning-based heuristics pack the pieces in a given order, which can impact the quality of the final solution. In order to decrease this dependence from the rectangle input order, Zhang et al. [Zhang *et al.*, 2016] have developed a recursive heuristic previously proposed by the same authors[D. *et al.*, 2006]. This recursive algorithm successively divides the strip into rectangles and sub-rectangles and when choosing which rectangle to place in a given empty rectangle, this choice does not depend just on the rectangle input sequence but on how well the rectangle fits the space.

Research is still active in this kind of heuristics, like Wauters et al.[Wauters *et al.*, 2013] that proposes "Improved Deepest Bottom Left Fill (iDBLF)".

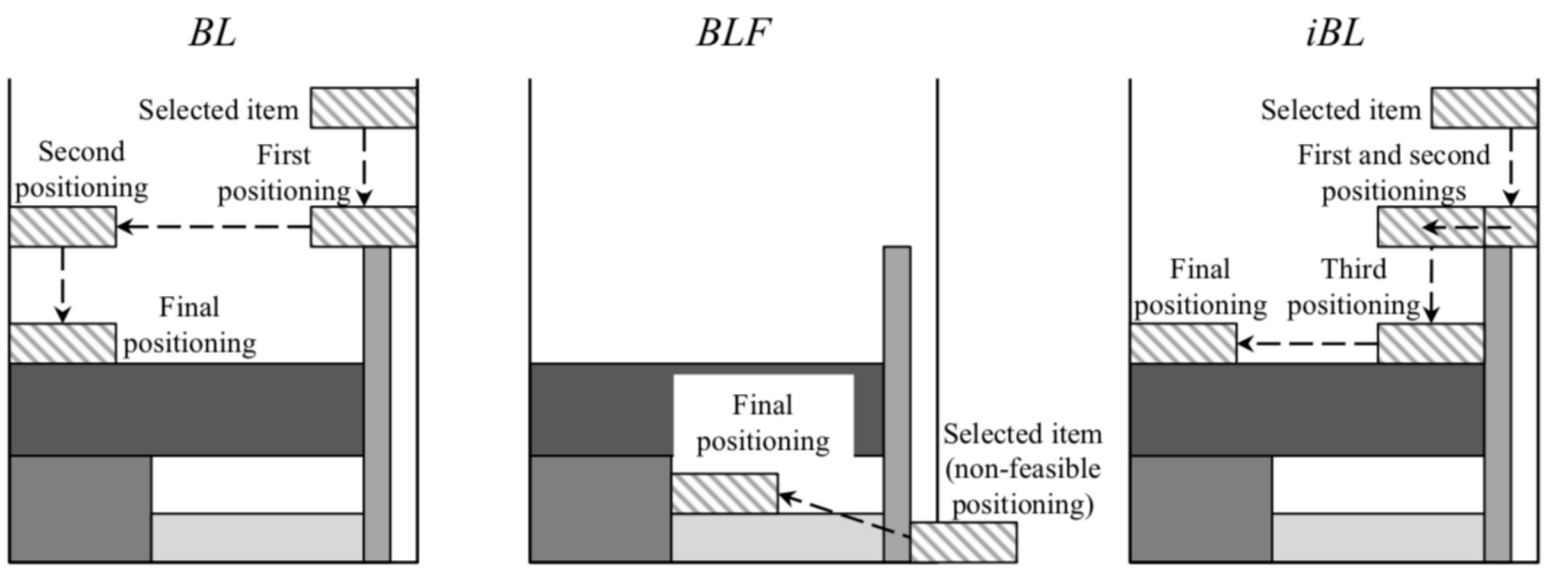


Figure 4: BL, iBL and BLF [Oliveira *et al.*, 2016b]

2.2.4 Fitness-based heuristics

Fitness-based heuristics are focused on the free spaces, i.e. the best-fit between the rectangle to place and the empty spaces.

- *Best Fit (BF)*: This was the first heuristic based on this mechanism. Proposed by Burke et al. [E.K. *et al.*, 2004], the selection of the rectangle to place is dynamic and depends on the space to be filled. The lowest free space is selected and the first choice will be a rectangle that perfectly fits this space. If there is no rectangle with this characteristic, the largest rectangle that fits the space is selected and placed according to one of three different policies: left-most policy (at the left of the empty space), next-to-the-tallest policy , and next-to-the-shortest policy (the latest depends on previously placed items). And efficient implementation of this, as proposed by Imahori et al. [Imahori and Yagiura, 2010], decomposes the "skyline" formed by the already placed rectangles into line segments and stores them in a heap structure using their y-coordinate as the key to access the memory structure.

- *Bidirectional Best-Fit (BBF)*: Improvement of BF proposed by Asik and Ozcan[B. and Özcan E., 2009]. In this work, not only the spaces between already placed rectangles are considered as feasible, but also what the author calls vertical spaces, i.e. spaces on the top of previously placed rectangles and with no side neighbor rectangles (See Fig. 5).

- *Fast Layer-Based Heuristic (FH)*: Usually rectangles do not fit completely the space, so many approaches tend to generate many small spaces that, for large instances, deteriorate significantly the efficiency. To solve this, Zhang and Leung.[Leung and Zhang, 2011] approach aims to keep the layout skyline as flat as possible. To do this, the quality of the relationship between a free space and a rectangle to be placed is designated as the fitness value. This measure takes values between 0 and a maximum of 4, when there is a perfect match between the space and the rectangle (See Fig. 6).

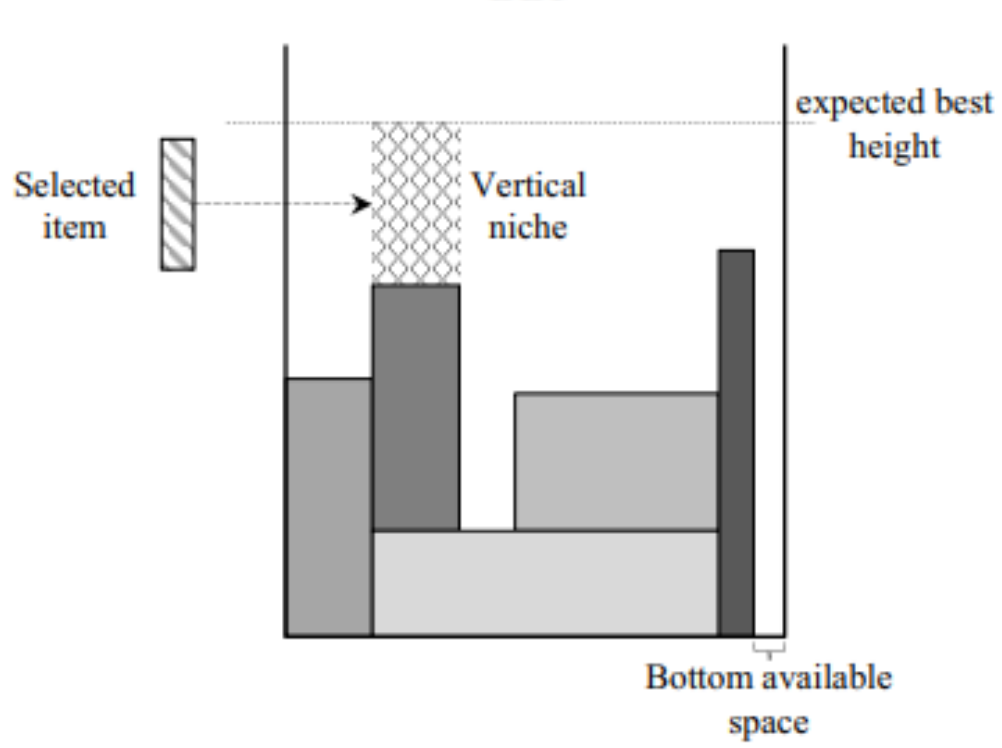


Figure 5: Example of BBF heuristic [Oliveira *et al.*, 2016a]

There are some recent works on improving this scoring methodology([Chen *et al.*, 2015], [Yang *et al.*, 2013]).

2.2.5 Level-based heuristics

The central idea of these heuristics is the placement of the rectangles in levels. They address specific real-world problems, like supermarket shelves.

- *Next-Fit Decreasing Height (NFDH)*: This heuristic[E. *et al.*, 1980] sorts the rectangles by non-increasing height and places them one by one on the currently open level and at the leftmost admissible position. It closes a level when it does not have enough space

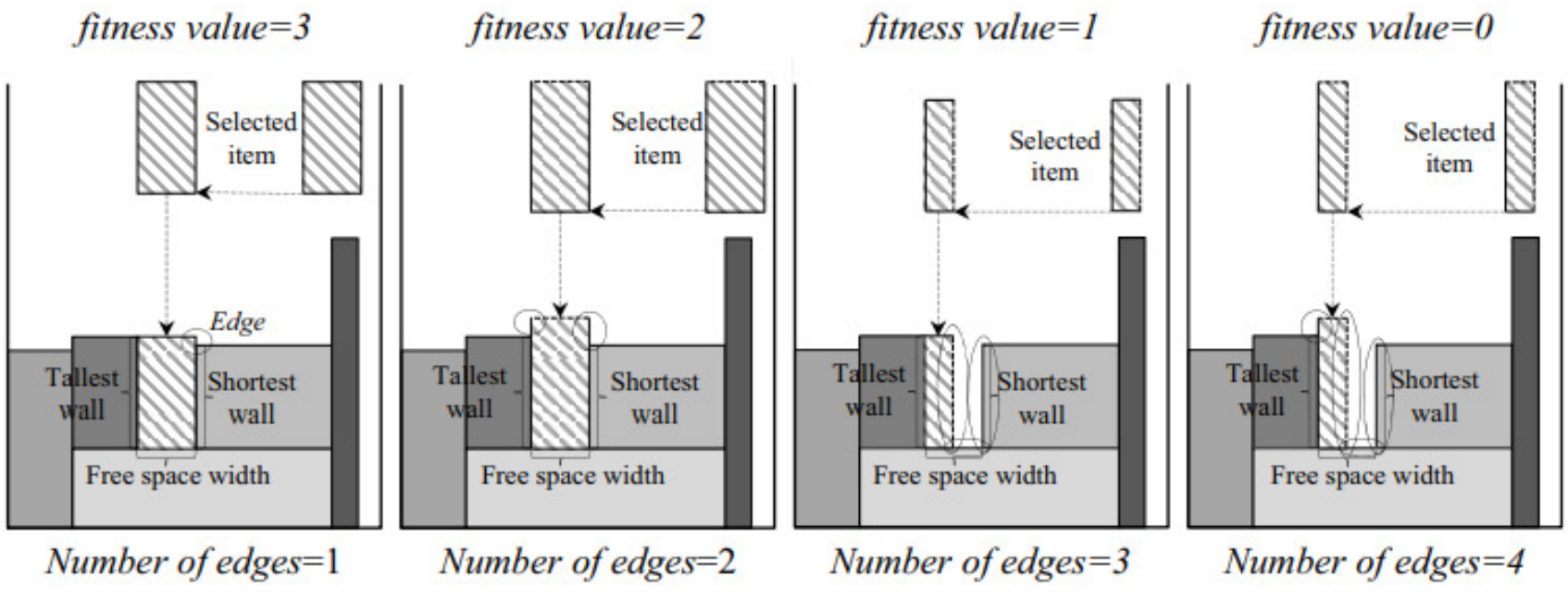


Figure 6: Example of FH heuristic [Oliveira *et al.*, 2016a]

to accommodate the candidate rectangle, opening a new level above the previous one. (Fig 7). A disadvantage of this heuristic is that free spaces of closed levels cannot be later on used by smaller rectangles.

- *First-Fit Decreasing Height (FFDH)*: , proposed by the same authors[E. *et al.*, 1980] allows to place each rectangle at the lowest possible level as long as it fits (it doesn't close levels).

- *Best-Fit Decreasing Height (BFDH)*: An improvement of FFDH proposed by Berkey and Wang[Berkey and Wang, 1987]. In this approach the rectangle is not placed at the lowest level where it fits, but at the level, among those where it fits, for which the unused horizontal space is minimum

- *Size Alternating Stack (SAS)*: The previous heuristics sort the rectangles by non-increasing height, and in instances where consecutive rectangles, which most probably will be placed on the same level, have very different heights lead to a very poor performance of these approaches as the taller rectangles will define a height for the level that can not be afterwards met by the shorter ones. To address this problem, Ntene[Ntene and van Vuuren, 2009] proposed the following: The rectangles are divided in two lists. List L_1 consists of all rectangles that are strictly taller than wider (narrow rectangles), and list L_2 has the rectangles that have a width equal or greater than the height (wide rectangles). The first list is ordered by decreasing height while the second list is ordered by decreasing width. The main idea behind the size alternating stack heuristic is to alternate between narrow and wide rectangles. Each level is initiated by comparing the heights of the first rectangles of each list and the tallest one is selected for placement, hence defining the level height. The following rectangle is taken from

the list that did not open the level and additionally rectangles from this list are placed on the top of each other, forming a column, until the level height is reached. Only afterwards a new rectangle is taken from the alternate list.

Research on level-oriented heuristics is still active, as shows the recent paper Buchwald and Scheithauer[Buchwald and Scheithauer, 2016] where an improved version of the FFDH heuristic is proposed.

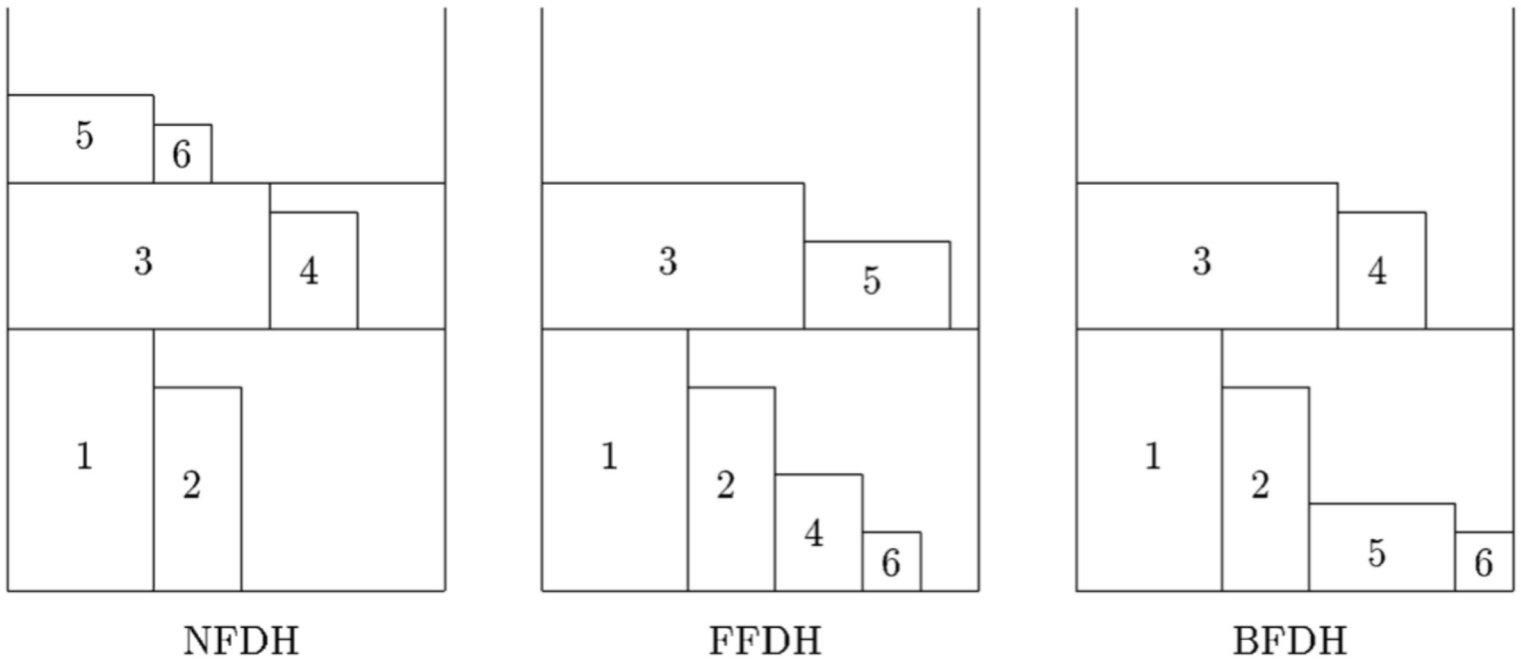


Figure 7: NFDH, FFDH and BFDH [A. *et al.*, 2002]

2.2.6 Improvement methods and metaheuristics

Improvement methods start with an initial solution, i.e. a complete solution obtained either by a constructive heuristic or randomly generated. This initial solution is then improved by applying small consecutive changes. They rely on metaheuristics to perform this changes, like Tabu Search, Simulated Annealing or Genetic Algorithms.

In the survey presented by Oliveira et al.[Oliveira *et al.*, 2016a], different improvement heuristics are classified in **search over sequences** and **direct search over the layout itself**.

Search over sequences

Heuristics that search over sequences resort to modification operators that are common to other problems that rely on sequences to codify their solutions: in other words, this type of methods codify the layouts in some data structure (like an array), and perform modifications operators like swapping or inserting to construct new layouts (after decoding using a specific method). The operators can change the ordering to place elements, rotate them, etc.

Thr first work using search over sequences and metaheuristics is Jakobs [Jakobs, 1996], with genetic algorithms being used as a search strategy and each sequence being transformed into a layout by the bottom-left heuristic. More recent applications of genetic algorithms to search over sequences can be found ([Hadjiconstantinou and Iori, 2007], [Burke *et al.*, 2010], [Thomas and Chaudhari, 2014])

Another strategy followed by Burke et al. (2009) [E.K. *et al.*, 2004] used simulated annealing to improve a partial initial solution generated by the best-fit heuristic. A first set of rectangles is placed by the best-fit heuristic and these rectangles are not moved during the search procedure. The remaining rectangles are placed by the bottom-left-fill heuristic, according to several sequences that are generated during the search, trying to use the holes between the initially placed rectangles.

An original approach in searching over sequences was proposed by Burke et al.[Burke *et al.*, 2011]: the *Squeaky Wheel Heuristic*. As all the others, changes the current sequence to generate the incumbent sequence, but incorporates a learning factor to impact the solution in the next iteration. It uses the best-fit heuristic as a decoder of the rectangle sequence, but the sequence is generated taking into account penalties assigned to each rectangle. A target for the height is defined (in this work it is the so called area lower bound – the sum of the areas of the rectangles divided by the width of the strip), and when a rectangle exceeds the expected height a penalty is assigned to it. The more penalized rectangles are the first ones to place in the next iteration. (See Fig. 8)

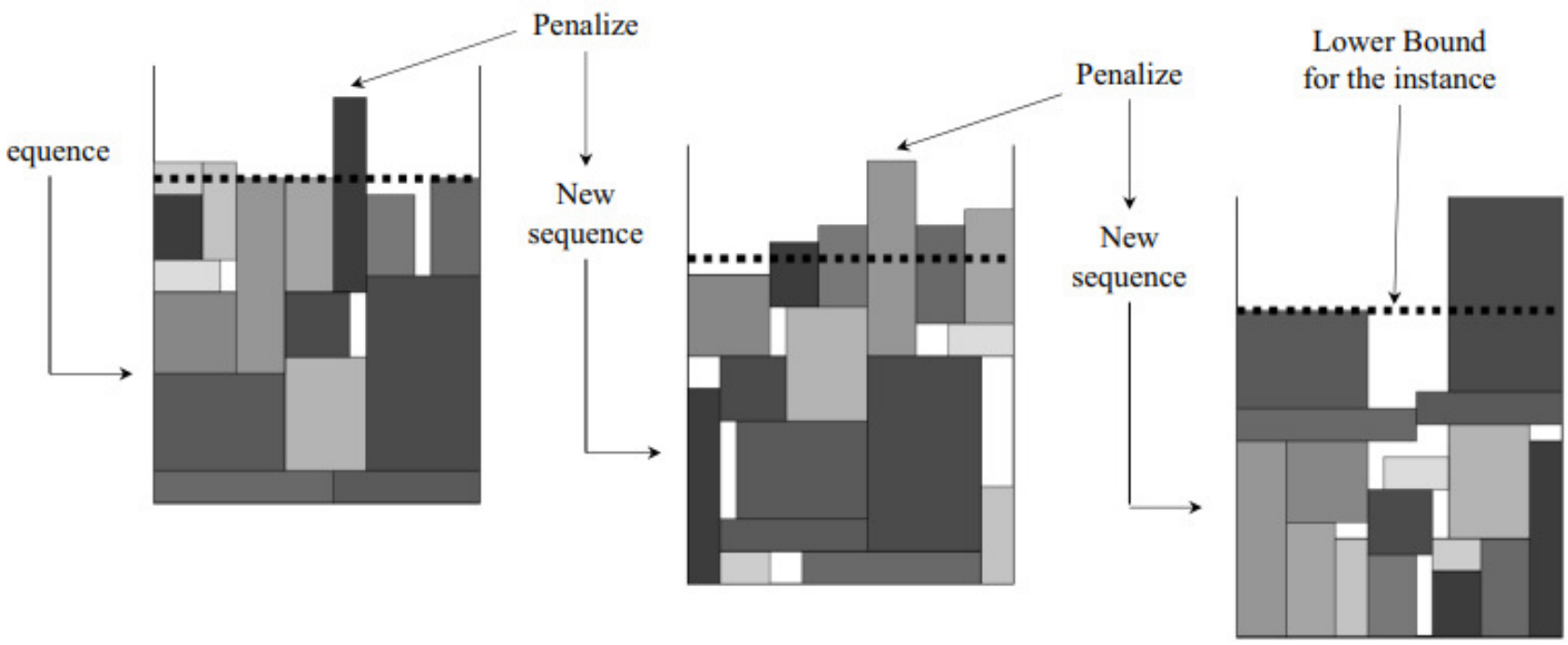


Figure 8: Squeaky wheel heuristic example [Oliveira *et al.*, 2016a]

Search over Layout

The main work based on this strategy is Alvarez-Valdes et al.[Alvarez-Valdes *et al.*, 2008],

which is a GRASP algorithm (random greedy procedure) and therefore has two phases: a constructive phase and an improvement phase. The constructive heuristic uses a fitness function, which assigns to each piece a fitness value based on how well the piece occupies the space, and then one of the pieces in the restricted candidate list is chosen to be the following one to be placed. The improvement phase is run over the actual layouts, and tries to remove a certain number of pieces to re-pack them. To do so, the authors proposed 4 different operators:

1. From a solution with a height H the last k% rectangles of the solution are removed, an artificial height of $H1$ is imposed to the strip and the solution completed with the deterministic constructive heuristic, if it's possible.

2. All rectangles that define the value H of the current solution (i.e. the rectangles whose top edge is at height H) are removed from the solution and placed on some of the empty spaces, lower on the strip. If the rectangle exceeds the dimensions of the empty space, the overlapping rectangles are deleted and placed again using the deterministic constructive heuristic (See Fig. 9).

3. The last k% rectangles that have been placed in the constructive phase are removed and placed again with the deterministic constructive heuristic (See Fig. 10).

4. Similar to the previous operator, but in this case all pieces with their top edge exceeding a height are removed, with$0 < \lambda < 1$ (See Fig. 11).

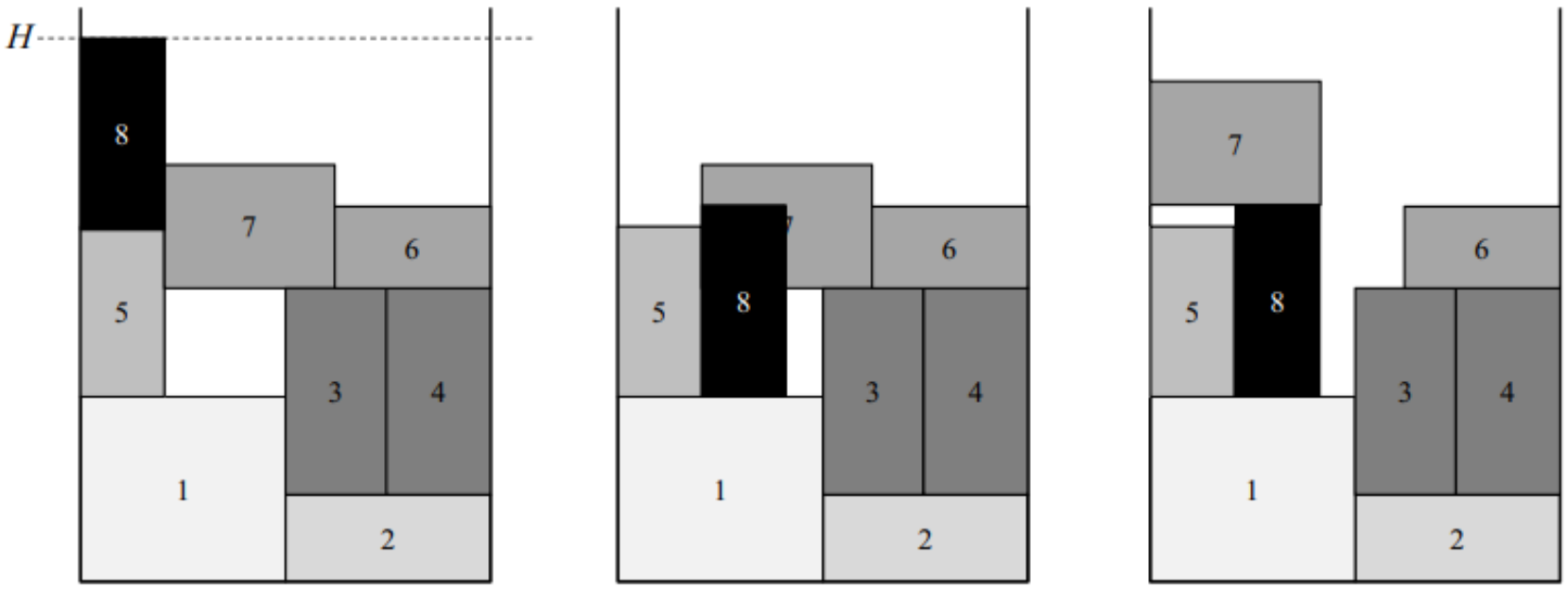


Figure 9: GRASP using improvement strategy #2 [Oliveira *et al.*, 2016a]

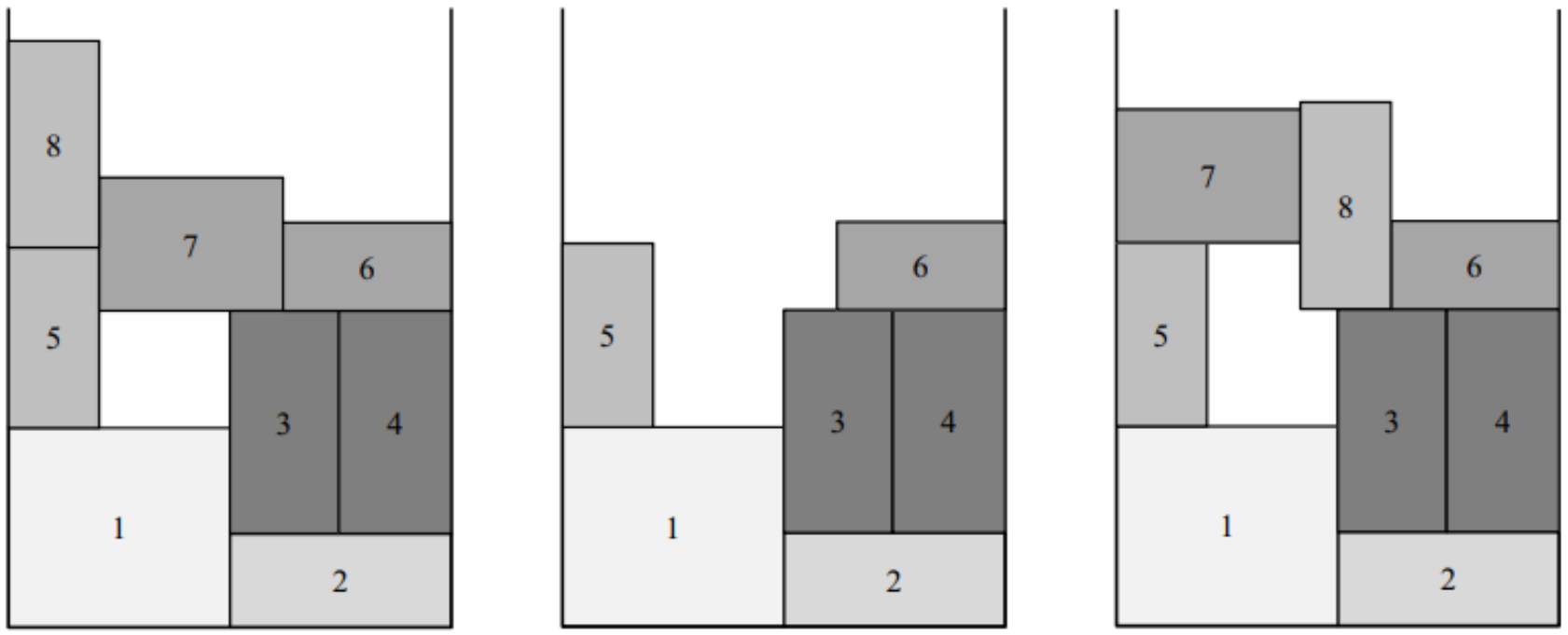


Figure 10: GRASP using improvement strategy #3 [Oliveira *et al.*, 2016a]

2.3 Multidimensional Multiple-Choice Knapsack Problem

Finally, it is necessary to mention the *Multidimensional Multiple-Choice Knapsack* (MMKP), which is a generalization of the classical knapsack problem. This variant is a combination of *Multidimensional Knapsack Problem* (MDKP) and *Multiple-Choice Knapsack Problem* (MCKP):

- The first one is a multidimensional version of the knapsack problem, where there are a vector of weights that cannot exceed a threshold vector, in an element-wise way.
- The second one considers group of elements, where a "one per group" constraint inside the knapsack is imposed.

2.4 Aesthetic Measures

An important aspect in newspaper design is aesthetics. It exists many proposal to measure aesthetics, which is naturally subjective; some classical aspects considered are balance, equilibrum or symmetry.

Ngo et al.[David Chek Ling Ngo, 2000] propose 13 aesthetic measures, focusing in screen design. We roughly describe these measures in the following sections, using examples. In each figure we add a number between 0 and 1, where 0 is the worst value for that measure,

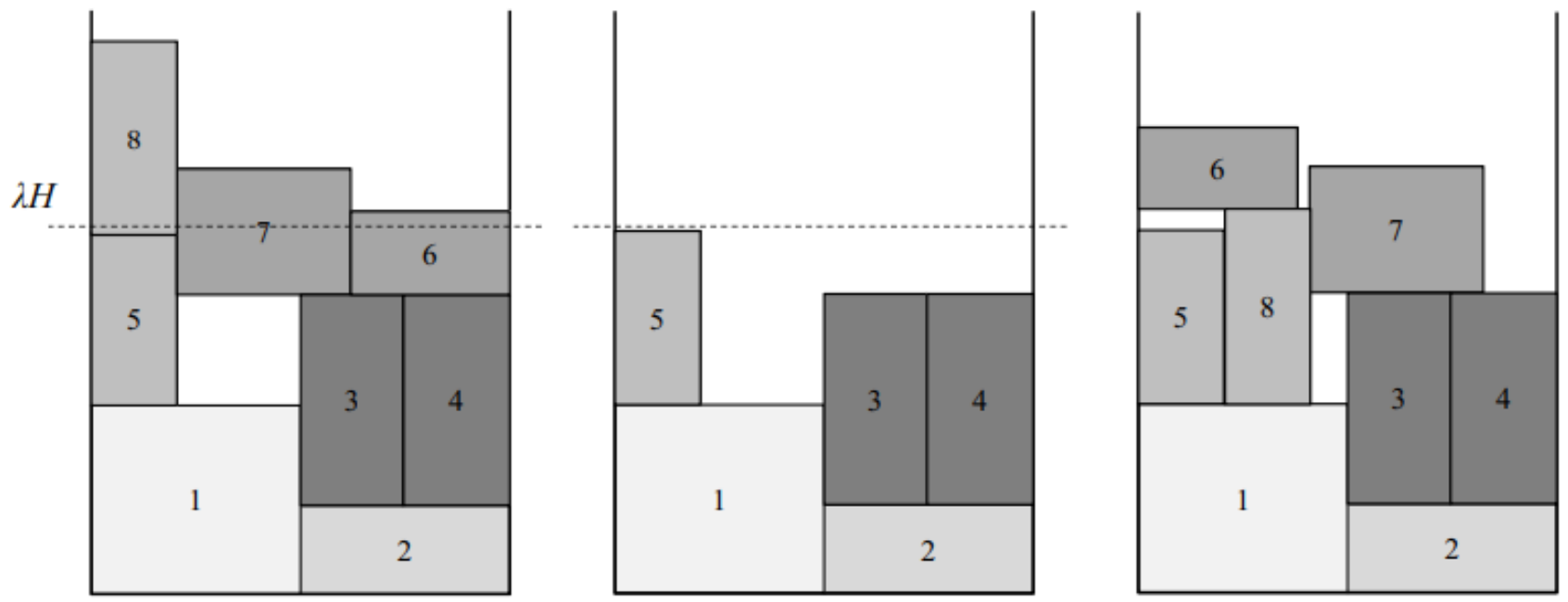


Figure 11: GRASP using improvement strategy #4 [Oliveira *et al.*, 2016a]

and 1 a perfect value related to each measure. This is calculated using equations defined by Ngo et al.

Balance

Distribution of optical weight in a picture. Based on "Larger objects are heavier, whereas small objects are lighter" notion. Is computed as the difference between total weighting of components on each side of the horizontal and vertical axis. See Fig.12: in the left figure, we see how the optical weight is almost exactly and the center of the image. In the right figure, the optical weight is slightly moved to the top, because the object at the bottom left corner is too small in comparison to the others.

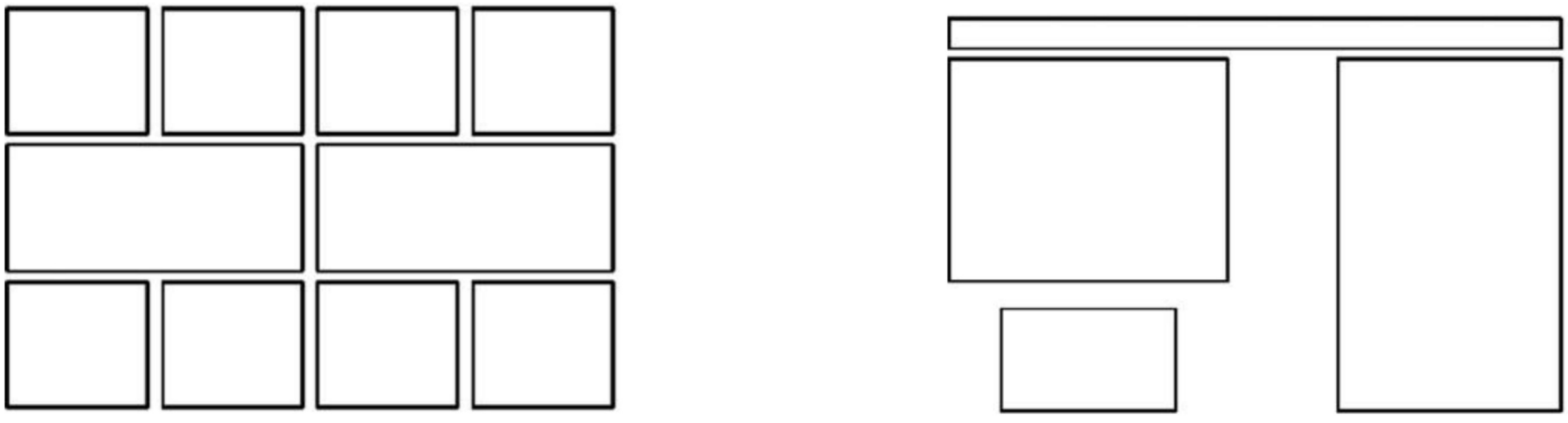


Figure 12: Balance = 0.9965 (left) vs Balance = 0.63771 (right) [David Chek Ling Ngo, 2000]

Equilibrum

Computed as the difference between **the center of mass** of the displayed elements and the physical center of the paper. See fig 13: the center of mass (calculated using the position together with the size of all articles) is exactly at the center of the image.

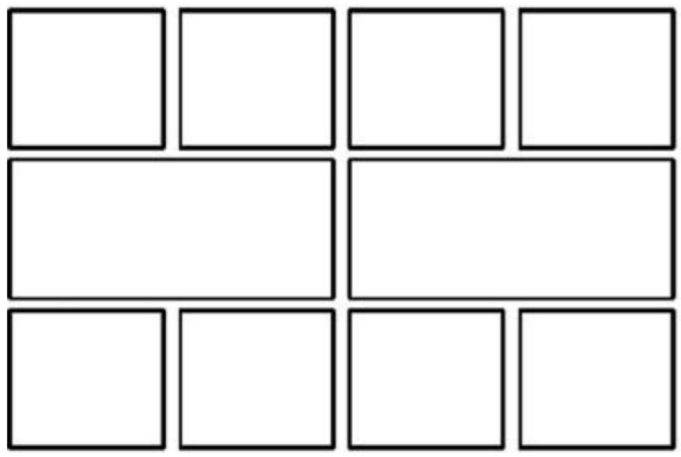


Figure 13: Equilibrium = 1.0 [David Chek Ling Ngo, 2000]

Symmetry

Based on the notion of **Axial duplication**; that means, an unit on one side of the center line is exactly replicated on the other side. This considers three axis:

- **Vertical Symmetry** refers to symmetry of a vertical axis.
- **Horizontal Symmetry** refers to symmetry of an horizontal axis.
- **Radial Symmetry** refers to symmetry of a central point.

See fig. 14: right figure has slightly bad symmetry because it is not symmetrical respect any axis.

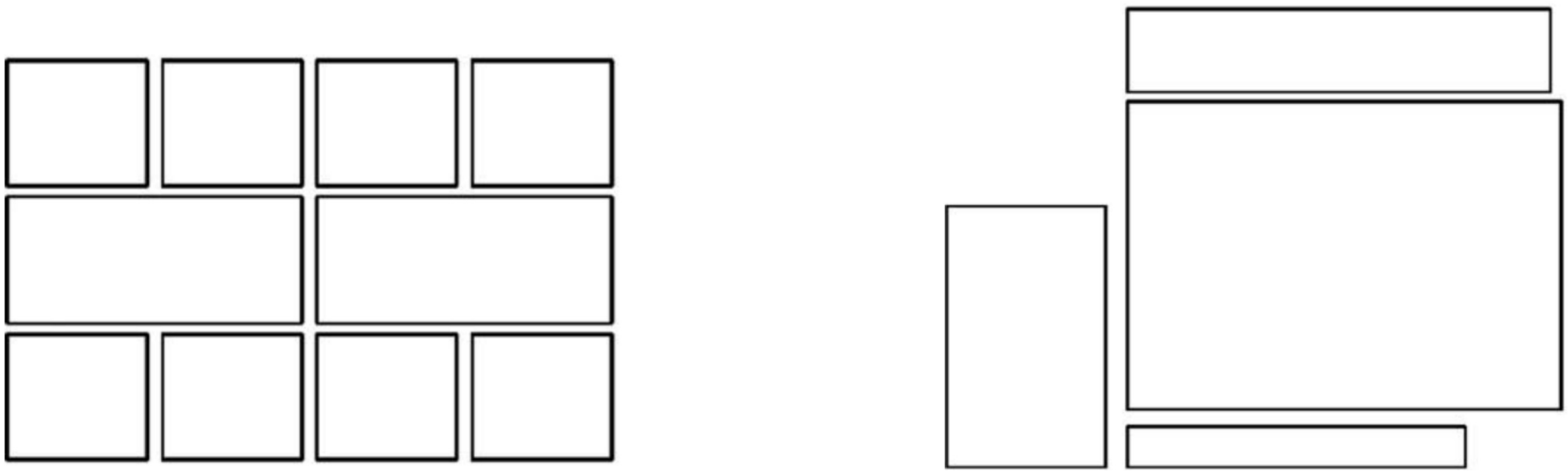


Figure 14: Symmetry = 0.9985 (left) vs Symmetry = 0.44697 (right) [David Chek Ling Ngo, 2000]

Sequence

Sequence in design refers to the arrangement of objects in a layout in a way that facilitates the movement of the eye through the information displayed. Normally the eye, trained by reading, starts from the upper left and moves back and forth across the display to the lower right (almost as a book).

See fig. 15: we see that left figure is presented almost as a regular grid of images, something nice to follow. On the other hand, righr figure doesn't have this distribution and it is difficult to follow as a book.

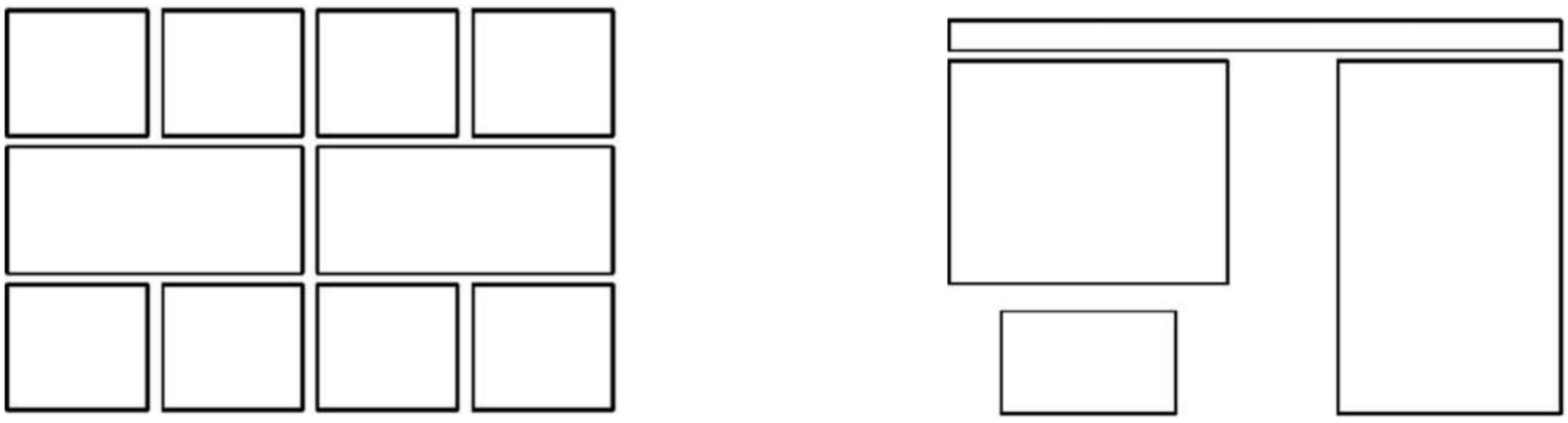


Figure 15: Sequence = 1.0 (left) vs Sequence = 0.25 (right) [David Chek Ling Ngo, 2000]

Cohesion

In design, similar aspect ratios promote cohesion. The aspect ratio of a visual field should stay the same during the scanning of a display. This measure the relative ratio of the object's

aspect ratio relative to the paper aspect ratio. See fig. 16: In this figure we can observe that differences between aspect ratios, and page's aspect ratio gives lower levels of cohesion measure.

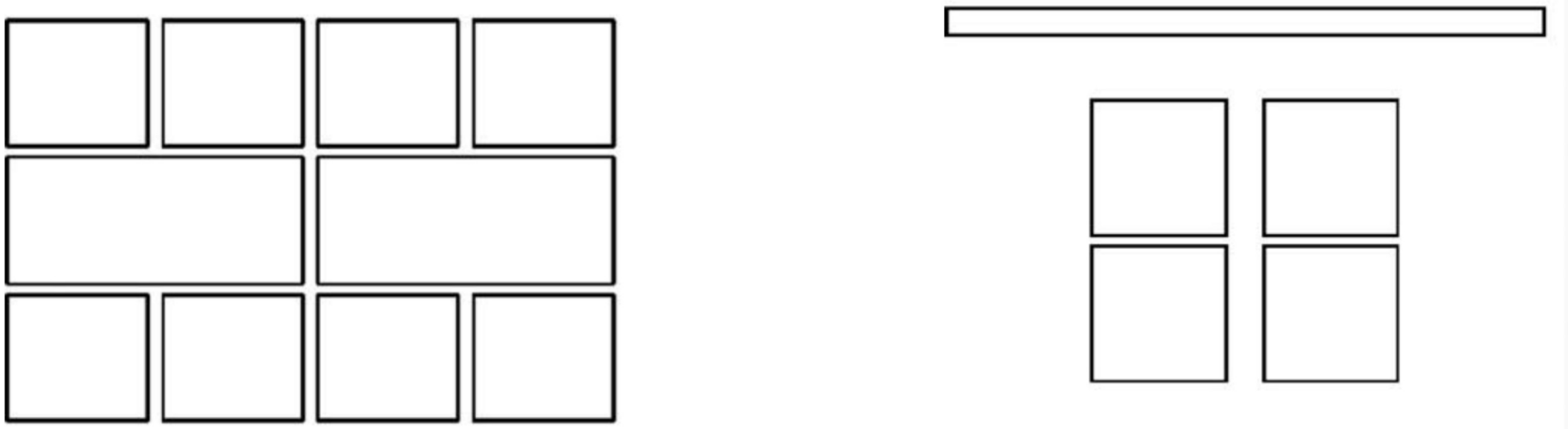


Figure 16: Cohesion = 0.8 (left) vs Cohesion = 0.71 (right) [David Chek Ling Ngo, 2000]

Unity

This measures how much the elements seem to belong together, to dovetail so completely that they are seen as one thing. In other words, if we can see all objects in a compact way that looks like one big object. See fig. 17: In this case, we see that one example looks like *one big image* in comparison to the right one, where the left block is separated from the rest.

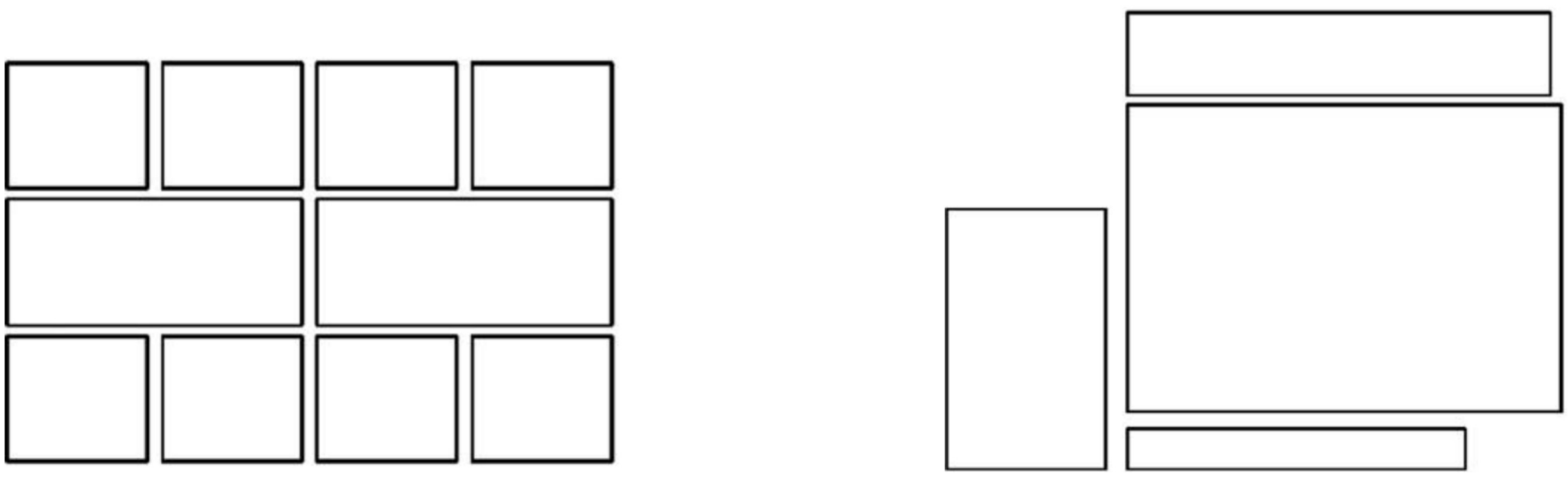


Figure 17: Unity = 0.87668 (left) vs Unity = 0.14543 (right) [David Chek Ling Ngo, 2000]

Proportion

Aesthetically pleasing proportions should be considered for major components of the screen, including articles and headings. This measure the difference between the objects aspect ratio and other ideal/*golden* aspect ratio, between the following (ratio between parentheses):

- Square (1:1)
- Square root of two (1:1.1414)
- Golden Ratio (1:1.618)
- Square root of three (1:1.732)
- Double Square (1:2)

See fig. 18: this example has a good proportion value because all objects except one are perfect squares.

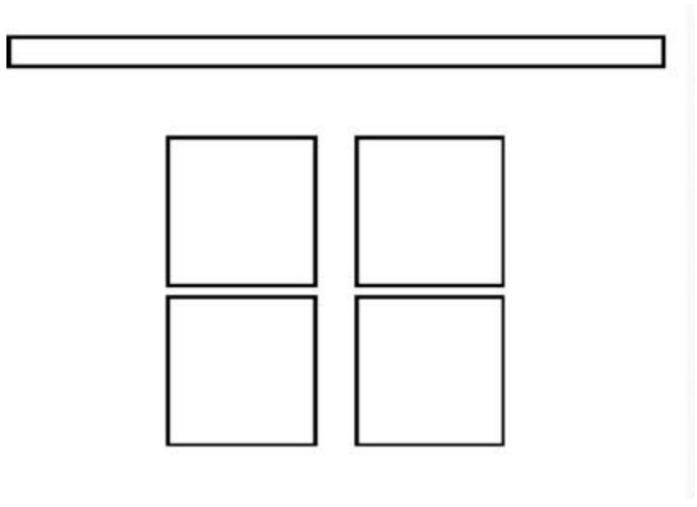


Figure 18: Proportion = 0.89779 [David Chek Ling Ngo, 2000]

Simplicity

Simplicity is achieved by minimizing the number of elements on the page and the alignment points (that means, align the object sides horizontally and vertically). See fig. 19: left layout has a better simplicity value than the right layout because it has less objects, and are aligned at the left and the right side of the page (except the smallest one). On the other hand, right figure has a very low simplicity value because it has a lot of objects and quite bad aligned.

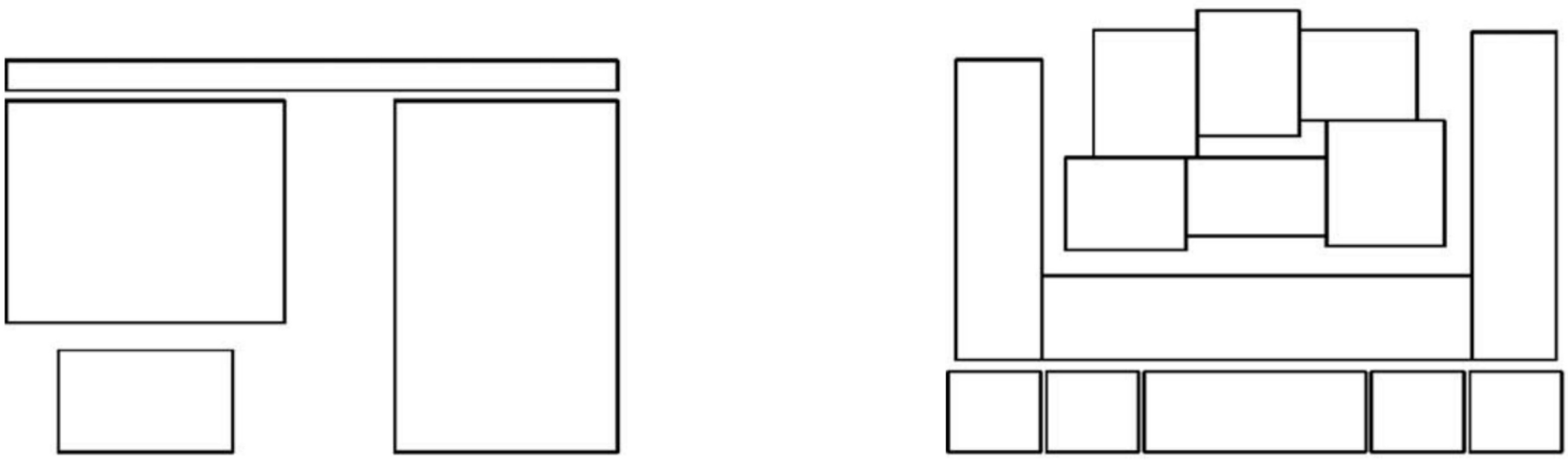


Figure 19: Simplicity = 0.3 (left) vs Simplicity = 0.08333 (right) [David Chek Ling Ngo, 2000]

Density

Density is area covered with objects. It exists different density levels.A suitable level is somewhat between 40% and 50% [Hays, 2018]. See fig. 20: right figure has a lot of empty spaces in comparison to the left one.

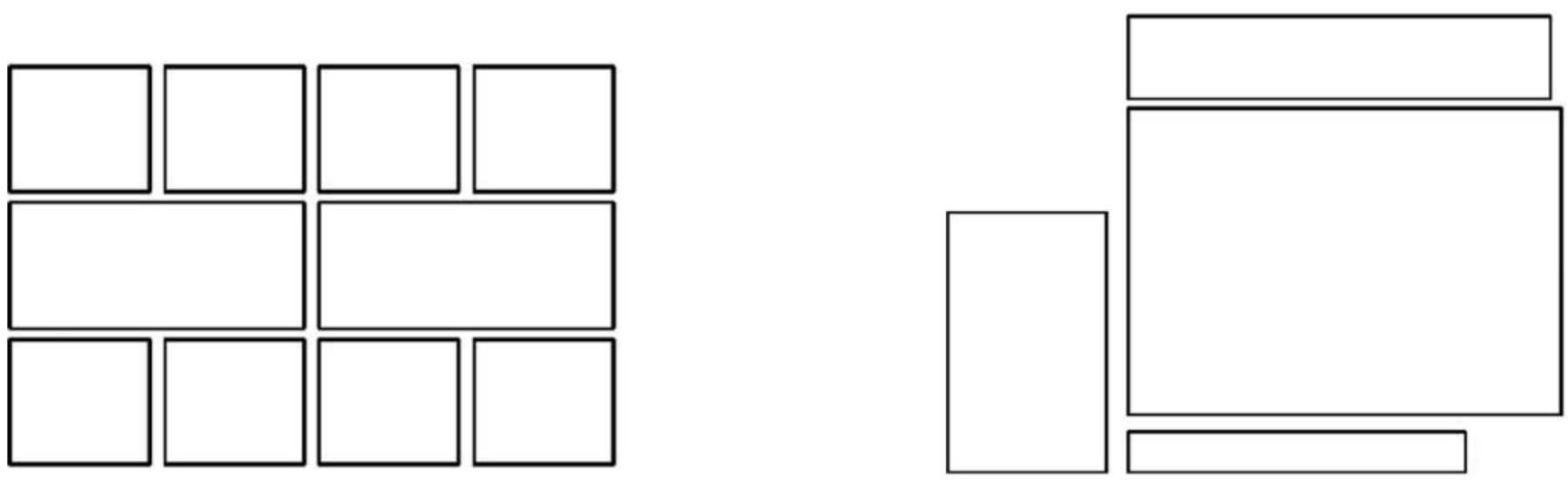


Figure 20: Density = 0.82188 (left) vs Density = 0.40792 (right) [David Chek Ling Ngo, 2000]

Regularity

Regularity is achieved by establishing standard and consistently spaced horizontal and vertical alignment points, and minimizing those points. In other words, if the objects are aligned almost as a regular grid of blocks. See fig. 26: left image is distributed nicely. On the other hand, right figure has a lot of horizontal and vertical alignment points, so the regularity value drops considerably.

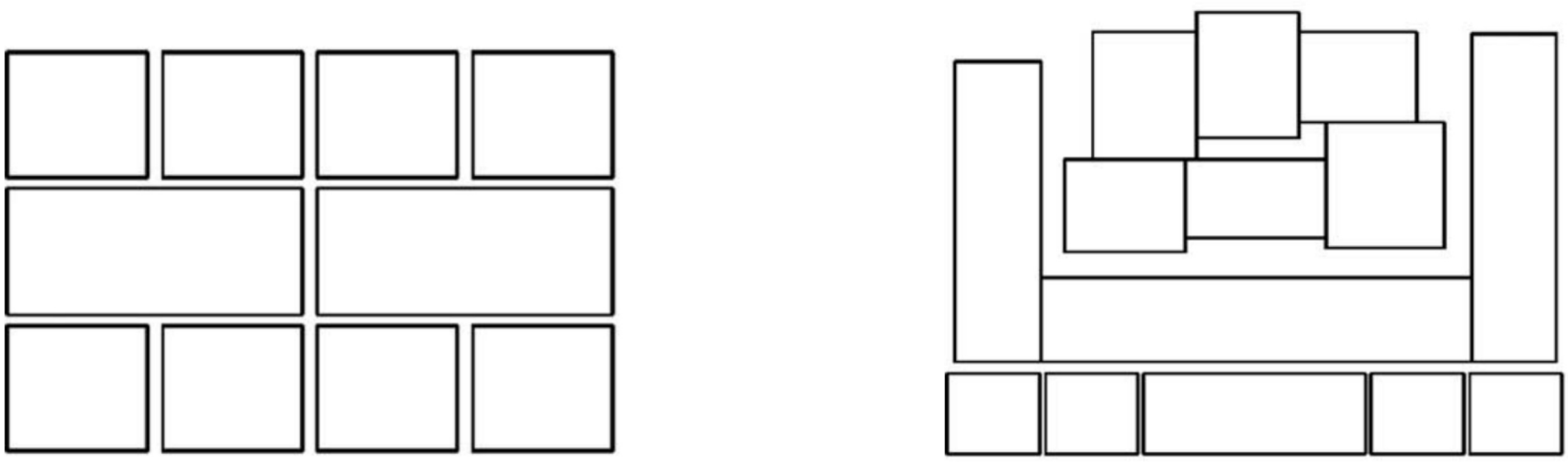


Figure 21: Regularity = 0.79722 (left) vs Regularity = 0.31868 (right) [David Chek Ling Ngo, 2000]

Economy

Economy is the careful and discrete use of display elements to get the message across as simple as possible. This is achieved by using as few different object sizes as possible. See fig. 22: left figure has similar objects sizes except the top rectangular one. On the other hand, right figure has a lot of different object sizes.

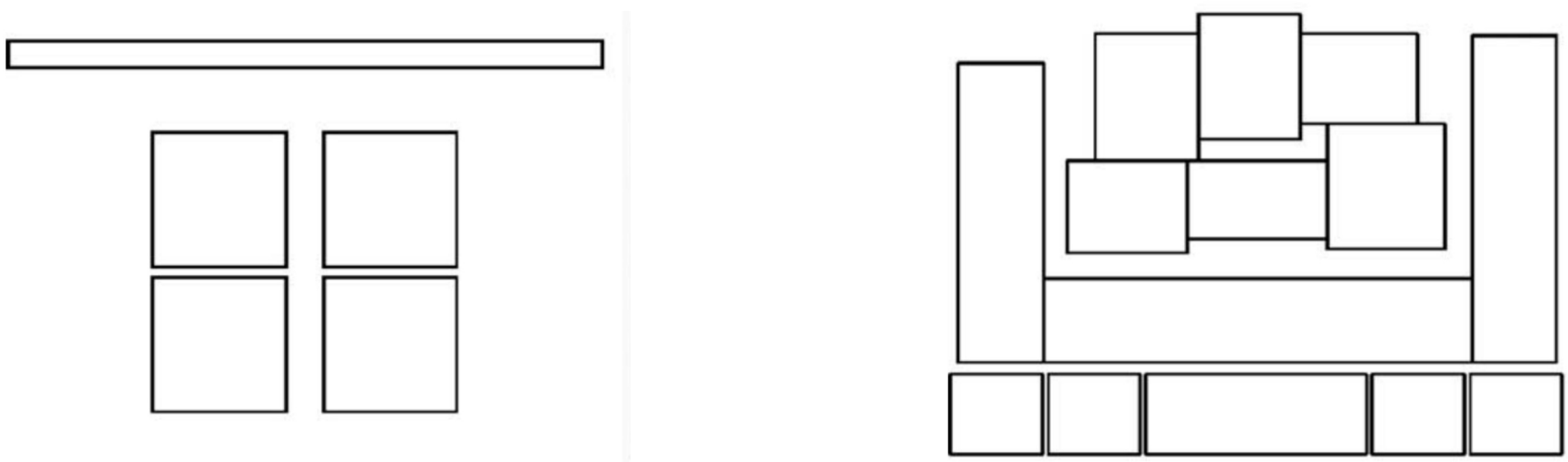


Figure 22: Economy = 0.5 (left) vs Economy = 0.09091 (right) [David Chek Ling Ngo, 2000]

Homogeneity

Given 4 quadrants (lower left, lower right, upper left and upper right); it measures number of objects belonging to each quadrant, and compares that with a perfect distribution (for n objects, $\frac{n}{4}$ per cuadrant).

See fig. 23: left figure has a perfect homogeneity, because all 4 quadrants has equal number

of objects. Right figure has different number of objects in bottom right quadrant and the other quadrants.

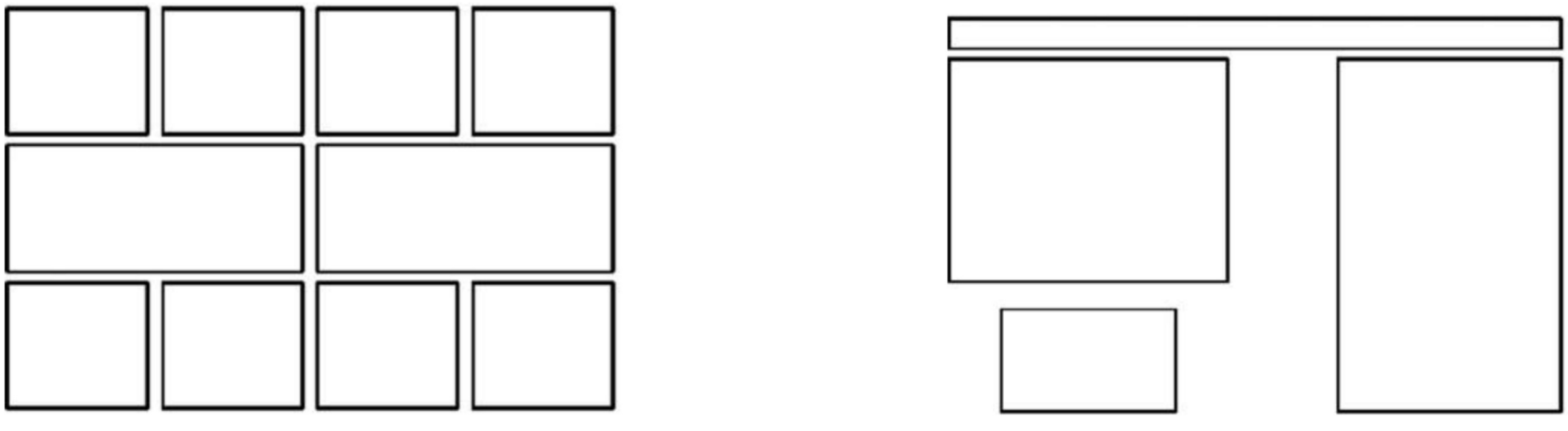


Figure 23: Homogeneity = 1.0 (left) vs Homogeneity = 0.00463 (right) [David Chek Ling Ngo, 2000]

Rhythm

Refers to regular patterns of **changes** in the elements. Rhythm is accomplished through variation of arrangement, dimension, number and form of the elements.

See fig. 24: left figure has regular changes of alignment points, and size of the figures. On the other hand, right figure has an abrupt change in distribution, related to the bottom left object.

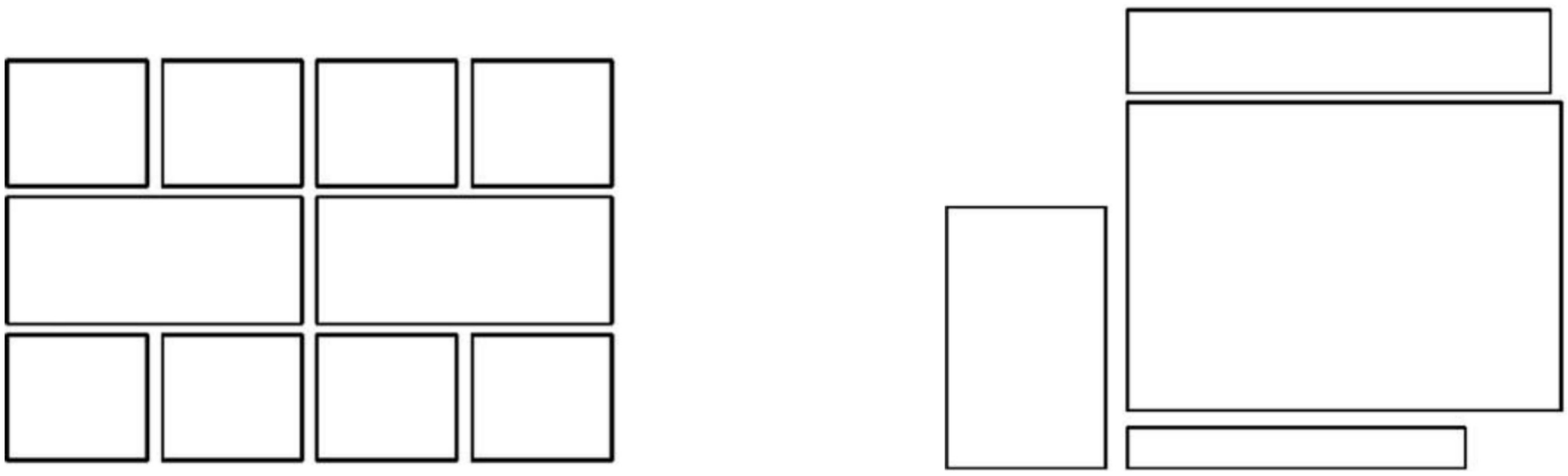


Figure 24: Rhythm = 0.99840 (left) vs Rhythm = 0.46527 (right) [David Chek Ling Ngo, 2000]

2.5 Newspaper Layouting

There exists many works about tools and methods in newspaper research.

First, we want to make a point on the use of genetic algorithms for layouting (Brabrand 2008 [Brabrand, 2008]; Purvis, 2002 [Purvis, 2002]; Geigel-Loui et al., 2000 [Geigel and Loui, 2000]). Genetic approaches can incorporate soft constraints (aesthetic) in chromosomes and/or fitness function, and hard constraints (overlapping, margin, etc.), or incorporate aspects other that position like fonts and content variation inside chromosomes. Also, in general, it doesn't need a lot of specific low level constraints, making the creation process simpler, and seems appropriate for such artistic tasks since, unlike other more brute force algorithms, the genetic algorithm does not attempt to mimic or model any particular process by which solutions are created. Instead, solutions are generated randomly and evaluated after the fact. On the other side, systematic approaches (like Branch and Bound methods) might be too large to solve in real life efficiency requirements.

On the other hand, some authors describes the notion of **spatial and abstract constraints** (Lok et al., 2001 [Lok and Feiner, 2001], Kivela et al., 2011 [**?**]) in constraint based layouting methods. Spatial Constraints describes position or size constraints of the components related to other componentes to the page itself (e.g. TEXT1 BELOW HEADLINER). On the other hand, abstract constraints describe high-level relationship between two components (e.g. TEXT1 IS IMPORTANT). In general, abstract constraints leads to more flexible distributions that spatial, giving a larger space of designs.

Another key concept relevant for our work, is the use of a grid to construct layouts (Jacobs-Li-etal, 2003 [Jacobs *et al.*, 2003], Strecker-Albayrak, 2010 [Strecker and Albayrak, 2010]). In grid-based layouting, the space is divided on different empty blocks, to fit content. We can also use **templates**; a template defines a particular set of content (example: 2 images, 1 body, 1 headliner) and its position on the newspaper page.

2.6 Chapter Summary

In this chapter a brief review of related problems, solving techniques and other topics like newspaper layouting and aesthetic measures was made. Problems like *Bin Packing* or *Strip Packing* had a lot of solving approaches, focusing mainly on heuristics (approximation algorithms) because exact and complete methods have usually long execution times. Some of the heuristics still used and improved these days are *Bottom Left* (and its variants) and *Best Fit*, included in some metaheuristics approaches like SA or Genetic approaches as a way to decode chromosomes.

Also, there is a brief mention to another related problem (Multidimensional Multiple Choice Knapsack Problem), that has some constraints in common with the problem presented in this thesis. Finally, a brief summary of aesthetics and a way to quantify them is presented, together with some algorithmic approximations to *layouting*. These provides measures and ideas that inspire our work in this thesis.

CHAPTER 3
PROPOSED SOLUTION

In this chapter we explain our method to solve the problem introduced before. First a detailed definition of the evaluation function is presented, followed by a description of the evolutionary algorithm to generate layouts.

3.1 Line Splitting

$H(.)$ uses number of lines needed to write the headline in a certain shape. To count this, and include it inside the optimization criteria, we specify first how the algorithm decides to split a headline (or any sentence) in lines.

- Font used in this work is Times New Roman.
- The width of the between-word spaces (shown as multiples of the normal space width) must be within 0.80 and 1.25. Indeed, the spaces between the words on each line are adjusted in order to achieve left-right justification so that the sentence snugly fits into the sentence bounding box (see Fig. 25). However, each space may be narrowed to no less than 80% or widened by no more than 125% of the width of a space. By this we avoid excessively loose or tight spacing between the words on each line which could otherwise impact the legibility of the sentence. So, this rule gives the line splitting criteria: if the words are too close between each others (i.e each space in a single line is narrowed to less than 80% of the normal size), the sentence is splitted in two (or more, to fulfill this constraint)
- The width of each line of text is, of course, given by the width of the bounding box that is, in this case, the width of each article's shape.

3.2 Evaluation Function

In this section we define all terms inside the evaluation function:

$$\max_{\mathcal{L}=\bigoplus \mathcal{A}_{i,j} \,/\, \{\mathcal{S}_{i,j}\} \in \mathcal{E}} E(\mathcal{L}) = H(\mathcal{L}) + F(\mathcal{L})$$

Figure 25: Example of a sentence splitted in 3 lines to fulfill the requirements of the line splitting criteria.

First we define $H(.)$. Then, how to calculate $F(.)$.

3.2.1 Definition of $H(.)$

The goal of this term is to penalize shapes with headline split exceeding λ_{max} lines. We define $H(.)$ as:

$$H(\mathcal{L}) = e^{-\sum_i f(\lambda(\mathcal{A}_{i,j}))},$$

with

$$f(s) = \begin{cases} 0 & \text{if } s \leqslant \lambda_{max} \\ \eta(s - \lambda_{max}) & \text{otherwise,} \end{cases}$$

where η is a parameter and s is the number of lines of an article.

3.2.2 Definition of $F(.)$

The goal of this term is to maximize the aesthetics of the layouts. We took inspiration from [Harrington *et al.*, 2004, David Chek Ling Ngo, 2000] and adapted the measures therein to better fit our requirements. We select three criteria from [Harrington *et al.*, 2004, David Chek Ling Ngo, 2000] that are the most relevant for our problem, namely alignment, regularity, and balance. We define $F(.)$ as:

$$F(\mathcal{L}) = A(\mathcal{L}) + R(\mathcal{L}) + B(\mathcal{L}),$$

where:

Alignment term $A(.)$ By alignments we refer to the alignment of left, right, top and bottom edges together (i.e., how the left edges of all shapes are aligned?). We started from the measure proposed in [Harrington *et al.*, 2004] and generalized it to account for all edges. To explain it better, let's see Fig. 3.2.2: Here, sub-figure 3.2.2a has many articles aligned respecting left edges, something not present in 3.2.2c. Also, 3.2.2a has 3 articles aligned respecting right edges, something that also isn't present in 3.2.2c. However, 3.2.2c has better *vertical* alignment (top and bottom) than 3.2.2a.

Let us focus on horizontal alignment of left edges to explain the method, that can be applied analogously to right, top and bottom edges. First we sort in an increasing order all the left edge x-coordinates of each shape to obtain a list $\{x_1, ..x_N\}$. For two consecutive x-coordinates (x_i, x_{i+1}), we define the cost:

$$g(x_i, x_{i+1}) = \frac{A}{A + |x_{i+1} - x_i|},$$

where A is a parameter. If the left edges of articles $\mathcal{A}_{i,.}$ and $\mathcal{A}_{i+1,.}$ are aligned, then $g(i, i+1) = 1$. Considering all consecutive edges, we can define the horizontal alignment measure for left edges by:

$$A_l = \sum_{i=1}^{N-1} g(x_i, x_{i+1}). \tag{22}$$

Analogously, we can define an alignment measure for the right edges (A_r), top edges (A_t) and bottom edges (A_b). Then, we define $A(\mathcal{L})$ as:

$$A(\mathcal{L}) = \frac{A_l + A_r}{2} + \frac{A_t + A_b}{2} \tag{23}$$

which domain is $[0, 1]$

Regularity term $R(.)$ Regularity suggest that it is best to have articles distributed in a regular fashion, close to a regular grid of blocks. For example, it is better if rows and columns of a table have relatively the same heights and widths. We use the same approach of Harrington's [Harrington *et al.*, 2004] idea. Let us focus on horizontal alignment of left vertical edges to explain the method; first, we obtain and sort all alignment points $\{x_1, ..x_N\}$ in the same way explained for $A(.)$. Now we are interested in changes of the alignment points, not in the points itself, and also we are not interested when $x_i = x_{i+1}$ because we are trying to see the pattern of changes between different alignment points . **So now, we measure the distance $z_{i,i+1}$ between 2 consecutive and different points**, and save this values for the next step. In Fig. 26 we explain this approach using an histogram to illustrate what are we measuring.

Now that we saved the distance $z_{i,i+1}$, we follow the same strategy explained in the definition of $A(.)$, but using this $z_{i,i+1}$ values instead of the left edge x-coordinates. That means:

Sort in an increasing order all the $z_{i,i+1}$ values. Then, for two consecutive $z_{i,i+1}$ values $(z_{i,i+1}, z_{i+1,i+2})$, we define the cost:

$$f(z_{i,i+1}, z_{i+1,i+2}) = \frac{R}{R + |z_{i+1,i+2} - z_{i,i+1}|},$$

where R is a parameter. The maximal value is 1 when $z_{i,i+1}$ and $z_{i+1,i+2}$ have the same value. Considering all $z_{i,i+1}$, we can define the horizontal regularity measure by:

$$R_h = \sum_{i=1}^{Z-2} f(z_{i,i+1}, z_{i+1,i+2}). \tag{24}$$

Where Z is the number of $z_{i,i+1}$ values.

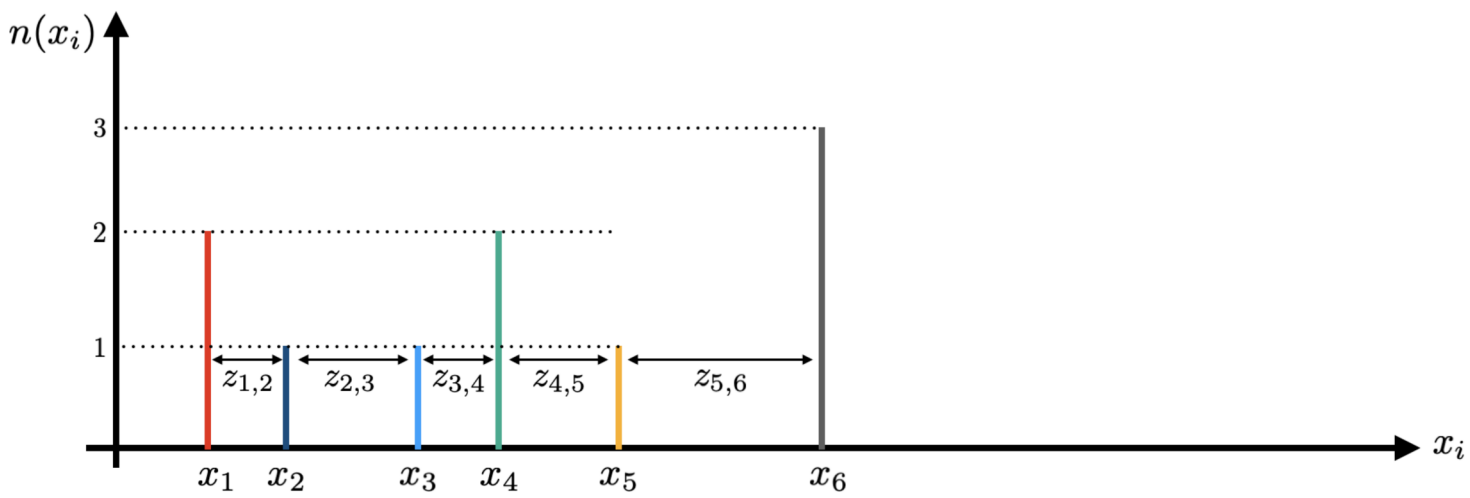


Figure 26: Illustration of the regularity measure, using an histogram just for visualization: Each entry in the histogram show the value of an alignment point. Then, the distance between 2 consecutive bars corresponds to $z_{i,i+1}$.

Analogously, with horizontal regularity (computed using left-edge alignment points) R_h and vertical regularity (computed starting from a top-edge alignment points) R_v, compute the overall regularity measure:

$$R = \frac{R_h + R_v}{2} \tag{25}$$

which domain is $[0, 1]$

Balance term $B(.)$ Balance can be defined as the distribution of optical weight in the newspaper page. Here we took the same definition as in [David Chek Ling Ngo, 2000]. Let us consider first the balance between the left and right sides of the page. Given the vertical

line $\lceil$ splitting the page in two equal parts ($\mathcal{D}_l$ and $\mathcal{D}_r$), we first define the left weight by doing a weighted sum of the areas of all articles $\mathcal{A}_{i,.}$ whose centers $(x^C_{i,j}, y^C_{i,j})$ belong to $\mathcal{D}_l$, i.e.,

$$w_l = \sum_{\mathcal{A}_{i,j}/(x^C_{i,j}, y^C_{i,j}) \in \mathcal{D}_l} a\left(\mathcal{A}_{i,j}\right) \delta_i,$$

where δ_i denotes the distance between $(x^C_{i,j}, y^C_{i,j})$ and line $\lceil$, and $a\left(\mathcal{A}_{i,j}\right)$ corresponds to article's area. Doing the same for the right-hand side, we can estimate a weight w_r and then define the balance between the left and right sides of the page

$$B_{lr} = \frac{|w_l - w_r|}{\max(|w_l|, |w_r|)},$$

which is a scalar value in $[0, 1]$. The exact same procedure can be applied to estimate the balance between the top and bottom sides of the page (B_{tb}), leading to our final balance term:

$$B(\mathcal{L}) = 1 - \frac{B_{lr} + B_{tb}}{2},$$

which is a measure in $[0, 1]$.

3.3 Evolutionary Algorithm

We propose an evolutionary algorithm (i) we can work with many layouts at the same time that convexity constraints are not required (ii) it is a technique adapted to artistic tasks because they do not attempt to mimic or model any particular process by which solutions are created[Geigel and Loui, 2000].

Given a newspaper page (defining page width, height, and a set of articles with their respective dimensions and coordinates), size of the multi-column grid γ, and the desired magnification factor, the algorithm works as follows:

1. Generate and classify alternative shapes for each article as allowed or unwanted.
2. Generate M random chromosomes using an initial population algorithm (see Sec. 3.3.4). Each one of the M chromosomes must be feasible.
3. Then, in each one of the N generations:
   (a) Apply crossover with probability p_c (see Sec. 3.3.7), mutation 1 with probability p_{m1} (see Sec. 3.3.8) and/or mutation 2 with probability p_{m2} (see Sec. 3.3.9). If any of this operators is applied, first select parents chromosomes using the selection procedure (see Sec. 3.3.6). Save the result for the next iteration, and continue until we fill $M - 1$ slots for the next generation.

(a) (b) (c)

Figure 27: Aesthetic terms illustrations. We can see that (a) has nice alignment (almost all the articles are aligned to the left of the page), but not so great balance (related to the optical center). On the other hand, (c) has a high value of balance but acceptable alignment. (b) is a good trade-off between these two aesthetic measure. Notice that even if (c) is a good layout related to balance, is not the best in terms of $H(.)$

(b) Additionally, save the best chromosome of all for the next generation (based on the evaluation function), to complete the population of M chromosomes.

3.3.1 Generation of alternative shapes

The first step is generate, for each article in the input, a list of alternative shapes $\{\mathcal{S}_{i,j}\}$ (different for each article). The algorithm do this following the definition in chapter 1, and Fig. 1e. Each possible shape has an index that is going to be used by the chromosome.

3.3.2 Chromosome Representation

Now, we need to define the basic data structure of a EA, a chromosome that encodes, in our case, a single generated layout.

See Fig. 28: Each chromosome contains two lists and an additional parameter:

1. a *shape* list of the used shape for each article. This list has one value for each article, indicating the index of the shape inside the corresponding list of alternative shapes. For example, $[3, 1, 2, 1]$ indicates that article #1 use shape #3 ($\mathcal{S}_{1,3}$) in this solution, article #2 use shape #1 ($\mathcal{S}_{2,1}$), article #3 use shape #2 ($\mathcal{S}_{3,2}$), and article #4 use shape #1 ($\mathcal{S}_{4,1}$).

2. a *permutation* list that defines the order of the articles to be packed in the newspaper page,

3. the chosen heuristic for packing (From a pool of alternatives)

We can decode or *render* the layout by choosing the shapes for each article from the list specified by the chromosome, and start packing inside the page using the chosen heuristic, in the order given by the permutation list. Then, we fill each article with the corresponding headline and body text with the magnification factor applied.

3.3.3 Pool of alternative heuristics

Now we define possible heuristics that can be used by each chromosome. The implementations were suggested by Jylänki's work in 2010[Jylänki, 2010].

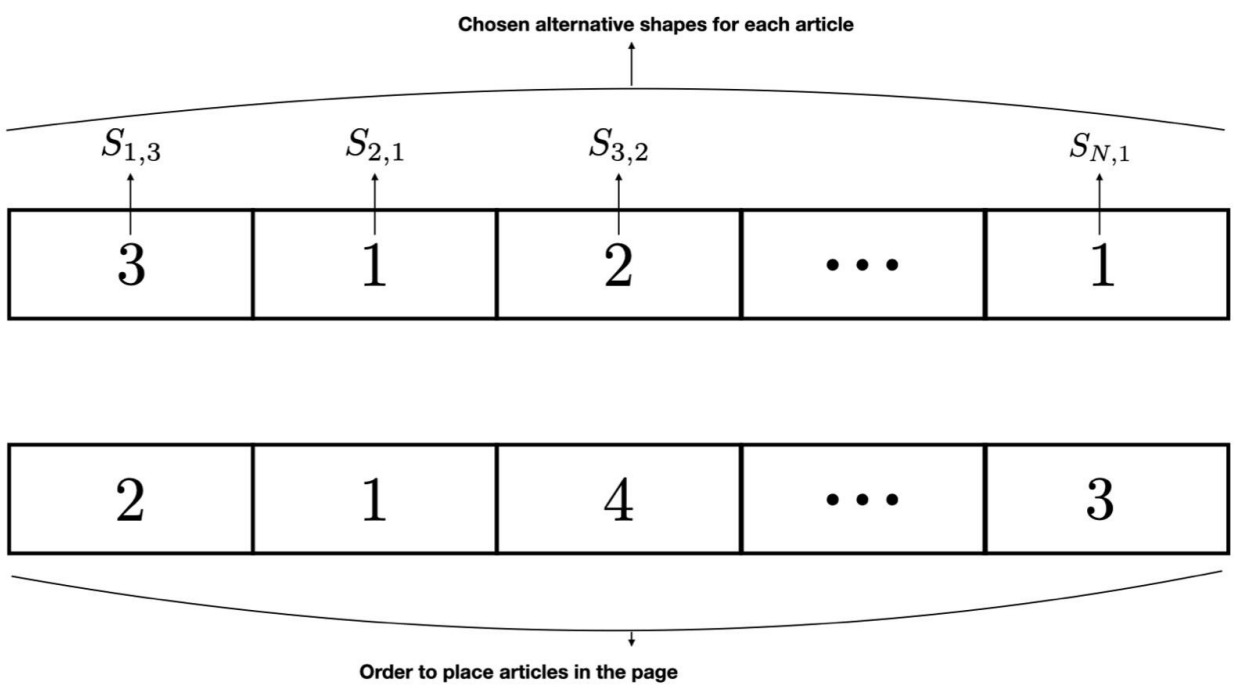


Figure 28: Example of the chromosome representation

Improved Bottom Left We implemented improved-BL (explained in Fig. 4) keeping track of the skyline, which is composed by edges formed by the topmost edges of already packed rectangles (See Fig. 29). The idea is place each piece, in the order given by the corresponding chromosome, in the most-bottom-left corner of the skyline.

Bottom Left Fill We implemented BLF (explained in Fig. 4) using a different data structure than iBL, because it is necessary to keep track of the "holes" that BLF fill with pieces. This heuristic uses Maximal Rectangles strategy, which basically stores a list of free rectangles that represents the free area of the bin, and pack each consecutive item in the in the most-bottom-left free rectangle that fits the piece (See Fig. 30).

Best Fit We considered and implemented Best Fit heuristic using the same Maximal Rectangles implementation than BLF. However, in this case, the algorithm packs each consecutive item inside the free rectangle that minimize the wasted area between the rectangle and the piece. If there is a tie, minimize width of the free rectangle to break it.

3.3.4 Initial Population Algorithm

The algorithm generates M random initial solutions. Each one of them must be feasible.

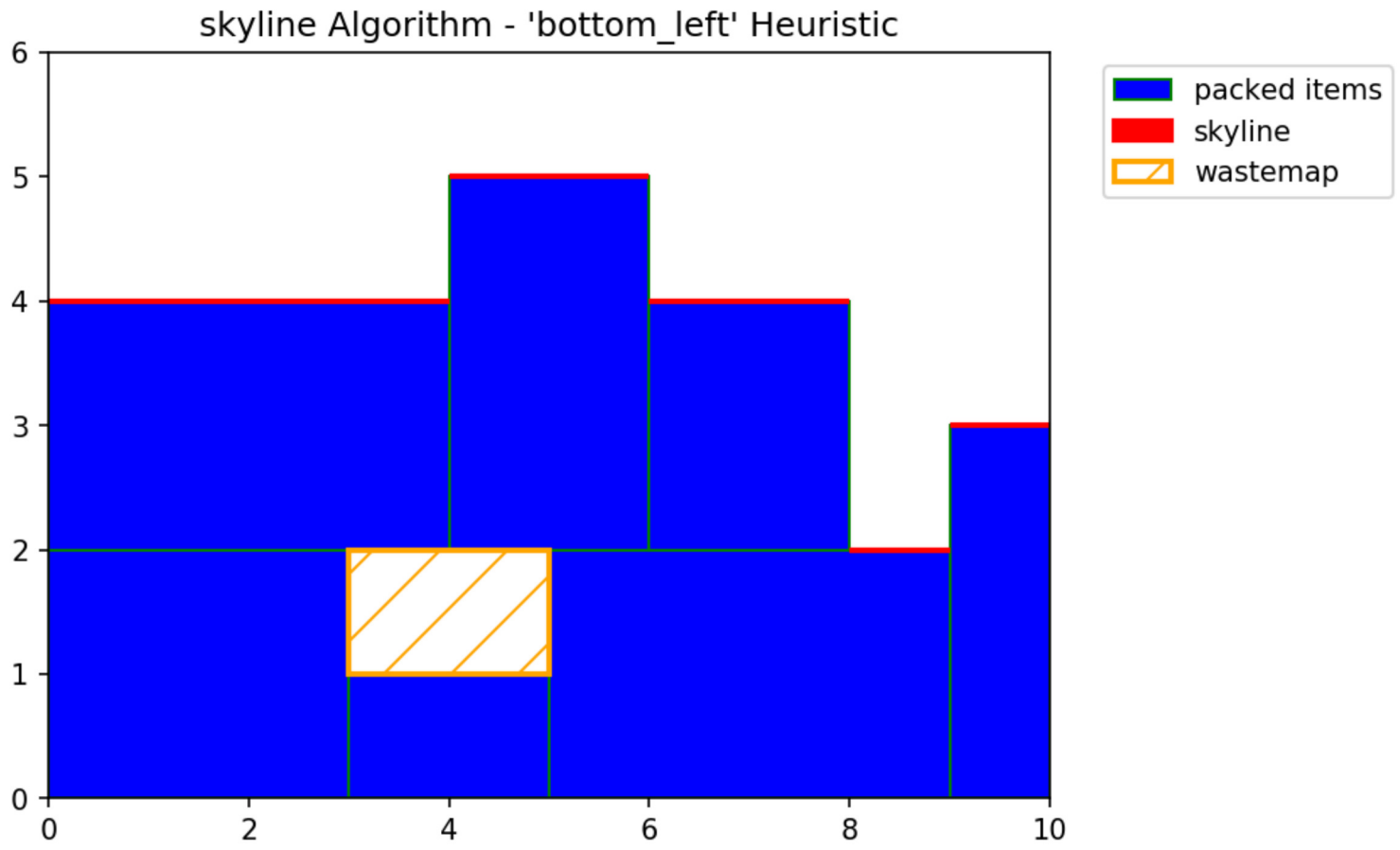


Figure 29: improved Bottom Left implementation using Skyline. [Jylänki., 2019]

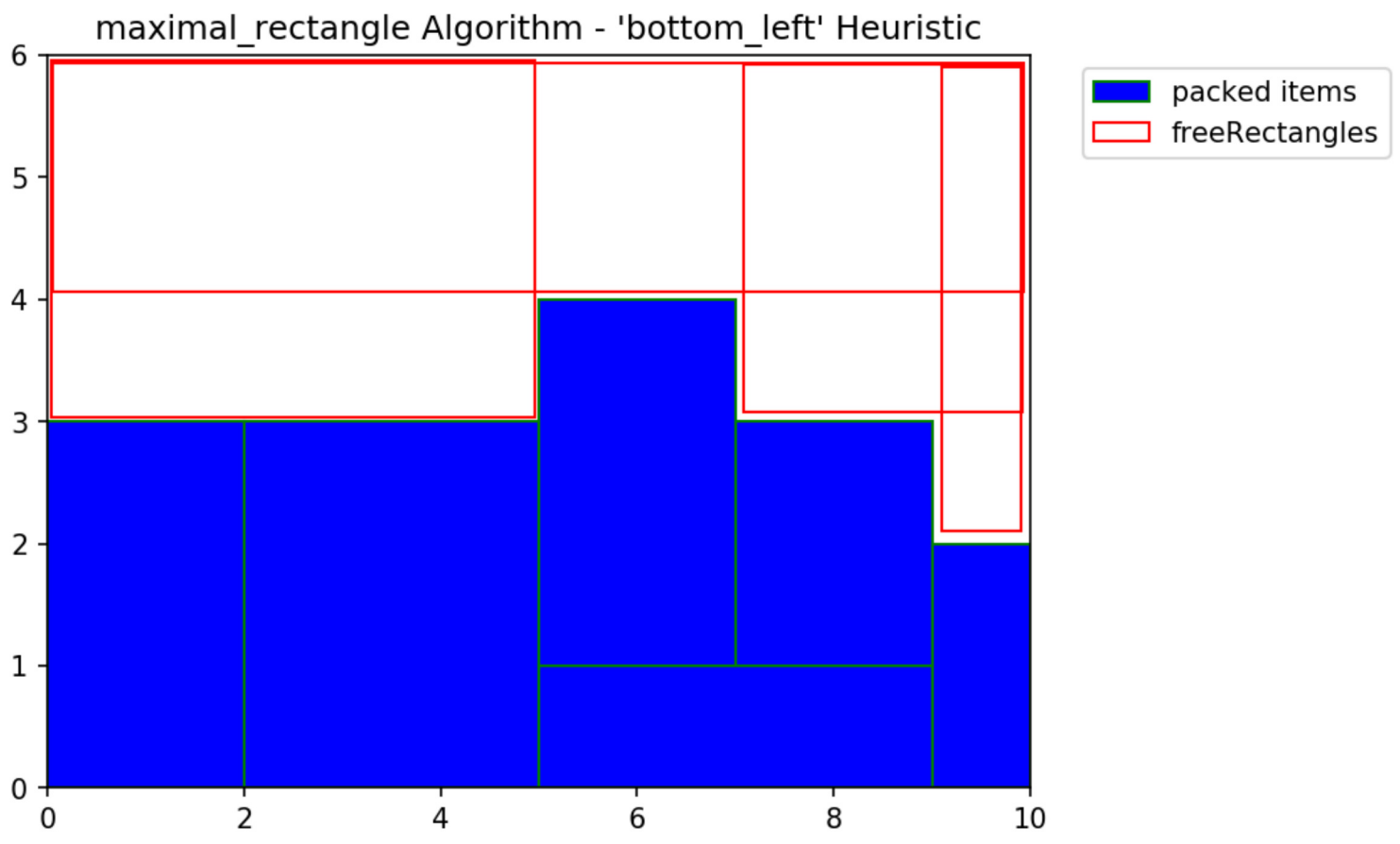


Figure 30: Bottom Left Fill implementation using Maximal Rectangles. [Jylänki., 2019]

1. Given all possible shapes $\mathcal{S}_{i,j}$ of each article $\bar{\mathcal{A}}_i$. We assume that $\{\{\mathcal{S}_{i,j}\}_{j=1,\ldots,N_i}\}$ are already classified as allowed or unwanted.

2. Generate a random chromosome:

   - random shapes for each article $\bar{\mathcal{A}}_i$ (all from the list of *allowed* shapes). Pick from the list of *unwanted* shapes only if there aren't *allowed* shapes for a given block.
   - random permutation and heuristic.

3. Render the layout. If it is feasible (i.e fits in the page), go to step 5.

4. If it is not feasible, try with a different random chromosome, changing the shapes, ordering and/or heuristic. Go back to step 2.

5. Save the layout as a chromosome.

3.3.5 Evaluation Function

The evaluation function is defined as:

$$\max_{\mathcal{L}=\bigoplus \mathcal{A}_{i,j} \,/\, \{\mathcal{S}_{i,j}\} \in \mathcal{E}} E(\mathcal{L}) = H(\mathcal{L}) + F(\mathcal{L})$$

This is used to measure the overall quality, and to save the best chromosome for the next generation.

3.3.6 Selection Procedure: The Roulette wheel and mini-Tournament using a Decision Tree

We use 2 different evaluation criteria in our evaluation function , based on $H(.)$ and $F(.)$. Both of them measure a different part of the chromosome representation:

1. We use $H(.)$ to measure the quality of the chosen shapes, because these impacts directly on the number of lines needed to write the content of each article. However, and because we want to polish the criteria even more, we don't use the function directly, but a criteria represented by a Decision Tree that we present in the following sections.

2. We use $F(.)$ to measure the quality of the permutation list, because the packing order gives the placement of each article, and aesthetic measures were created to check the quality of that placement.

Simply using $E(\mathcal{L})$ in a selection procedure, like a roulette wheel or a tournament might not be enough, because one term could predominate over the other. So the question is: **How can we include our evaluation function in the evolutionary algorithm?**

We propose a 2-step selection procedure: First, we use a roulette wheel to select 2 candidates, using $F(.)$. Then, we select between these 2 candidates with a mini-tournament, using a Decision Tree based on $H(.)$ function.

With this procedure, the algorithm choose a single parent to continue with a certain operator (crossover or mutation). In case of crossover, we use this selection procedure twice, to obtain 2 parents.

Roulette wheel selection The implementation is an standard roulette wheel selection: this procedure choose one chromosome of M, in a way that chromosomes that have better fitness value are more probable to choose than others.

Its probability $p(\mathcal{L}_i)$ of being selected is

$$p(\mathcal{L}_i) = \frac{F(\mathcal{L}_i)}{\sum_{j=1}^{M} F(\mathcal{L}_j)} \tag{26}$$

Obviously, in each selection step, we run this roulette twice, to select 2 candidates for the next step.

Decision Tree based on $H(.)$ We wanted to polish the $H(.)$ function with a more complex criteria, using a Decision Tree that will decide between 2 candidates $\mathcal{L}_1$ and $\mathcal{L}_2$, giving a single solution to the corresponding genetic operator.

Why using this instead of $H(.)$ directly?: because it polish even more the idea of *excess*; instead of simply counting the total number of lines above the threshold, like $H(.)$, this decision tree decides with a more complex criteria, choosing between 2 candidates taking into account if there are some articles with too many lines (even if the total sum is equal or similar), and even *playing* with this excess to make a better decision.

First, we calculate the *excess* of each article for both candidates: that is, calculate and store $\max\{\lambda(\mathcal{A}_{i,j}) - \lambda_{max}, 0\}$ for each article $\mathcal{A}_{i,j}$ in both proposed solution. Then, we classify each excess in 4 categories:

- *Null*: 0 lines of excess
- *Low*: 1 or 2 lines of excess

- *Mid*:3 or 4 lines of excess
- *High*: 5+ lines of excess

Finally, select between both solutions following the decision tree in Fig. XX. Note that, when it says $\mathcal{L}_1$ *Low* it means that $\mathcal{L}_1$ has at least one article with Low excess.

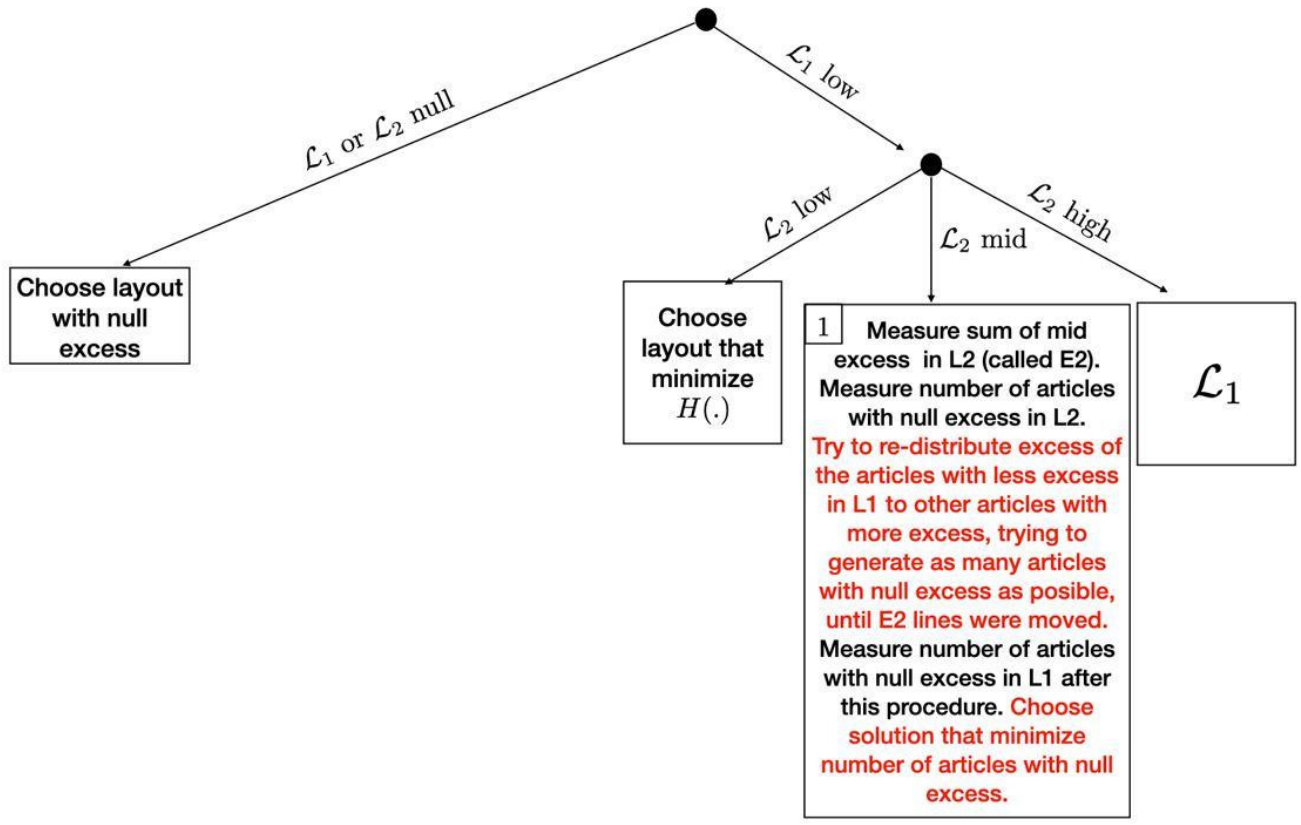


Figure 31: Decision tree. Part 1

3.3.7 Crossover Operator

The crossover operator works as follows:

To generate a child it inherits the shapes list of the first parent, and the permutation list/heuristic of the second parent. Also, generate another child with the opposite: shapes list of the second parent and permutation/heuristic of the first one.

Only feasible chromosomes are selected; if it is not, if the first child is unfeasible, the first parent is selected to the new generation, and/or if the second child is unfeasible, the second parent is selected to the new generation.

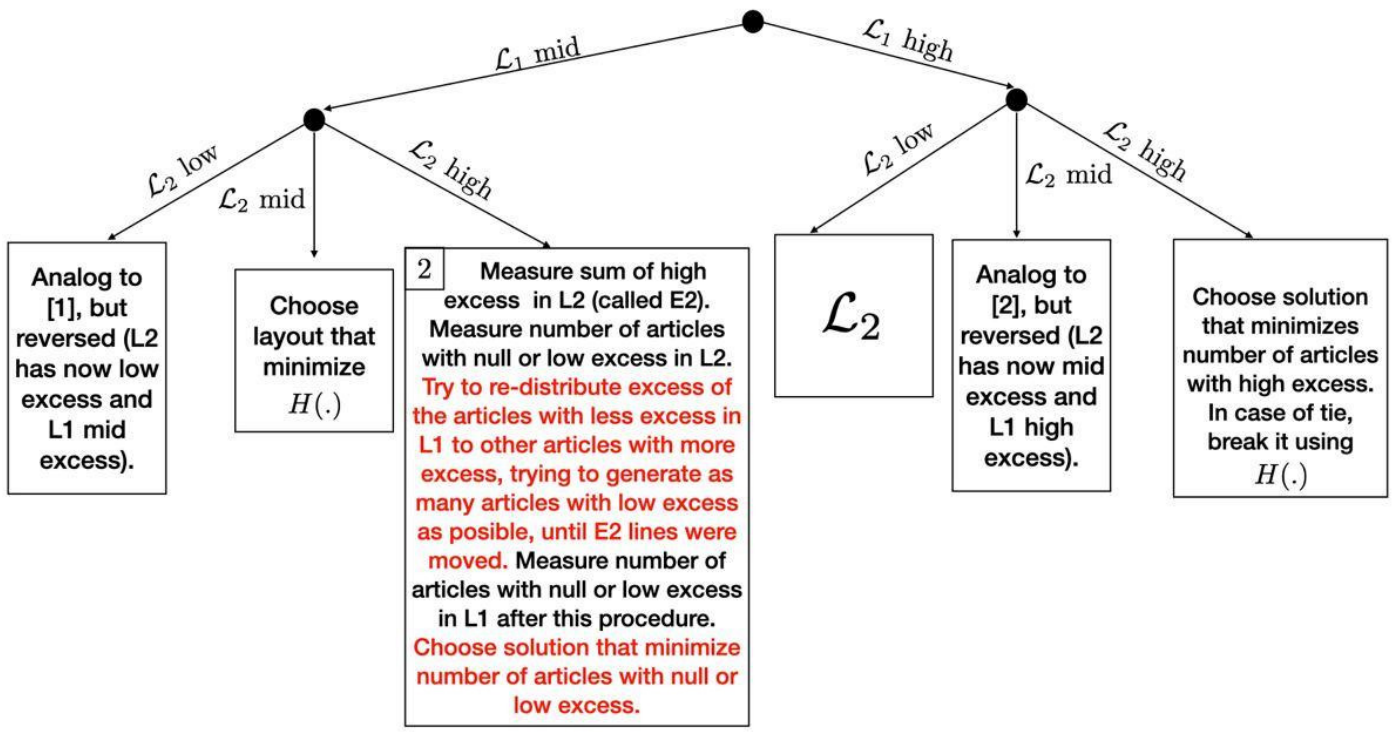


Figure 32: Decision tree. Part 2

3.3.8 Mutation 1: for permutations

To mutate the permutation array: generate two random numbers p, q, and swap the numbers of indices p and q.

Again, only feasible chromosome is selected. If the child is not feasible, the parent is selected to the new generation

3.3.9 Mutation 2: for heuristic

To mutate the heuristic: Simply change it to another random one from the pool of possible heuristics.

Only feasible chromosome is selected. If the child is not feasible, the parent is selected to the new generation

3.4 Chapter Conclusions

$H(.)$ takes into account accesibility related to number of lines, so it is one of the terms that the E.A is trying to optimize and uses it in the selection procedure. This impacts which shape to choose for each article, but not the placement inside the page. $F(.)$, on the other hand,

measures global layout quality, and it is the other term that the E.A is trying to optimize. In this case, impacts only the placement, but not the shapes directly. This shows one of the good things of this genetic design: each term quantify a different aspect of the solution, and are used in different parts of the selection procedure.

Also, another thing to mention is that even if we can use $H(.)$ directly, a decision tree step was included because it polish even more the idea of *excess*; instead of simply counting the total number of lines above the threshold, this decision tree decides with a more complex criteria, choosing between 2 candidates taking into account if there are some articles with too many lines (even if the sum is equal or similar), and even *playing* with this excess to make a better decision.

Another thing to mention is the initial generation of chromosomes; it is important to stick to feasible examples always, because the geometrical constraints make the process of going from unfeasible to feasible very difficult. Also, small changes in some shapes can move the chromosome to an unfeasible space of solutions very fast. That's why the design generate *set of shapes* at the initial generation, and then it doesn't make individual changes, but decides to use the whole array in crossover/mutation operators, performing small changes in the permutation array only, that it is easier to *play* with.

CHAPTER 4
VALIDATION OF SOLUTION

4.1 Parameters

We fix the following parameters for all the experiments below

- Problem parameters
  - Magnification factor α: 2.0
  - Width of the column-grid γ: It depends of the input, should be indicated by the newspaper layout. In all the instances used in this experiment, it equals to 387.
  - λ_{max}: 3 lines.
- $H(.)$ parameter η: 0.1
- Aesthetic Measures parameters
  - $A = B = 100$
- Genetic Algorithm
  - Number of Generations: 200
  - Size of Population: 20
  - p_c: 0.5
  - p_{m1}: 0.5
  - p_{m2}: 0.5
- Experiment Set Up
  - 7 instances
  - 18 seeds per instance

4.2 Implementation and Machine

Proposed Genetic framework was implemented in C++ programming language, using modular functions that allows to set parameters for the optimization criteria and the genetic procedure itself (like population, generations, etc.)

Tests were run in an Intel Core i5 machine of 3.0 GHz.

4.3 Instances

The proposed approach was tested using 7 different instances. Each one of them belongs to real newspaper front pages of the American newspaper *New York Times*. Each one of them was manually annotated, that means, marking each article, images and sections, using a tiny software developed by BIOVISION team members at INRIA, modified by the author of this thesis. This allows to save each frontpage as a text file that specifies the size of the newspaper, number of articles/images/sections, and coordinates together with dimensions of each one of them.

Additionally, we generate another file specifying the content of each annotated article in the newspaper page; to avoid copyright issues and considering that this work considers text-only articles, the content file is generated with random *lorem-ipsum* headline text (with a random number of lines between 1 and $\lambda_{max} = 3$ in our tests), and random body text. The idea is to have less than the number of lines threshold $\lambda_{max} = 3$ for a non-magnified front page, and check the results with a given magnification factor.

4.4 Output and Rendering of solutions

The output of the genetic algorithm is a population of proposed layout that have the same structure of the input instances, specifying the position and dimensions of each article.

Additionally, a tool to render the output of the genetic algorithm was developed. This tool takes the content file specified in the input, and the output file of the genetic method, and renders a PNG image with the layout.

Observation about $H(.)$ and the Decision Tree

Something important to remember is that we use $H(.)$ in some boxplots to analyze solutions, but inside the algorithm it is used indirectly by the Decision Tree.

4.5 First Experiment: Only Aesthetic vs Only Headline vs Weighted Sum Roulette Wheel

In our first experiment, we want to analyze the results for different evaluation functions inside the roulette wheel. As we explained in the corresponding section, the selection procedure has 2 steps, with a roulette wheel in the first part with a certain function to assign the portion's size of the wheel for each candidate solution.

We test 3 different roulettes:

1. Using $H(.)$ function only.
2. Using $F(.)$ function only.
3. Using a weighted sum between these two: $0.3H(.)+0.7F(.)$. We assign more importance to $F(.)$ term because the Decision Tree used in the second step of the selection process already takes into account number of lines and excess, and the roulette is the only place that $F(.)$ is considered.

We run the algorithm (for all 7 instances, 18 seeds per instance, as explained before) for each type of roulette. In all cases, we used an offline version of the method, that means, using a fixed heuristic instead of making it part of each chromosome. The chosen heuristic was *Bottom Left Fill*.

4.5.1 Visual Comparison

First, we want to have images showing how the algorithm works: We can see these results in Fig. 33 to Fig. 39; it is clear that our method generate layouts that minimizes the excess of lines (all articles with more than 3 lines are marked with red edges). In some cases, we see that the generated layout is simply a permutation of the same shapes described by the input inside the page, but is not the general case (as we can see in Fig. 35 or Fig. 38). This happens because the algorithm couldn't find a better distribution in terms of number of lines, something common in these instances because the geometrical constraints are hard to manage when changing shapes for each article, because is trying to fit blocks inside a closed container instead an open container as SPP or BPP.

On the other hand, in terms of aesthetics, there are a lot of changes between solutions using different roulettes, seeing better balance using an aesthetic only roulette (For example, Fig. 37 or Fig. 39).

Finally, something interesting is that instance *nyt8* generate different results in terms of shapes between different runs, for the same roulette (Fig. 40), something uncommon in comparison to the other instances that found an optimal set of shapes (equal or different than the input) and *plays* with the permutation to improve aesthetics. However, as we can see in the boxplots analyzed below, these different solutions are all equivalents in terms of $H(.)$. This proves that the algorithm is capable to obtain different solutions in terms of shapes and distribution in the page, that also optimize $H(.)$ (We will see below what happens also with $F(.)$).

(a) (b) (c) (d)

Figure 33: Comparison of best individuals for instance nyt1. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

4.5.2 Headline Value Comparison

Then, we obtain boxplots of $H(.)$ value to analyze variance of the algorithm respecting this criteria in different runs. We have the results in Fig. 41 to Fig. 47.

In all instances and roulettes, all the population converges to the best individual found in the initial generation, in terms of shapes, for all seeds (with *nyt8* having different in shapes but equivalent in $H(.)$ as we can see in Fig. 47). This first shows that there are no differences in terms of $H(.)$ between different roulettes.

However, almost all solutions are different in terms of distribution inside the page (we will analyze the impact of this in aesthetic terms). This is not unexpected considering that individual changes in any shape for any article might generate unfeasible solutions, because of the geometrical constraints; that's why we design the genetic operators to exploit set of

(a) (b) (c) (d)

Figure 34: Comparison of best individuals for instance nyt2. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

(a) (b) (c) (d)

Figure 35: Comparison of best individuals for instance nyt3. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

Figure 36: Comparison of best individuals for instance nyt5. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

Figure 37: Comparison of best individuals for instance nyt6. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

Figure 38: Comparison of best individuals for instance nyt7. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

Figure 39: Comparison of best individuals for instance nyt8. (a) Original Input. (b) Headline Roulette. (c) Aesthetic Roulette. (d) Headline+Aesthetic Roulette.

(a) (b) (c) (d)

Figure 40: Comparison of best individuals for instance nyt8 for different runs, but same roulette (Aesthetic). We can see that for this instance it is possible to find different set of shapes that have the same Headline Value, so there are equivalent in this sense. (a) Original Input. (b) Solution. (c) A different solution with the same Headline Value but different shapes. (d) Another possible solution with the same Headline Value with different shapes.

shapes found at the initial generation of solutions, so it is expected to converge to the same set of shapes in most of the cases. This behavior is shown in Fig. 48 and Fig. 49 (it is just one example); in the former one, we can see that the best solution was found in the initial generation. In the latter one, it is clear that the worst individual converges quickly to the best one.

Also, let's see Fig. 50; we can see that in this case, the best solution it is not the one with the highest $H(.)$ value, but the best one **considering the Decision Tree**.

4.5.3 Aesthetic Value Comparison

All the previous graphs and analysis were considering only shapes and $H(.)$. The differences in the placement of the each article inside the page is controlled by $F(.)$, so let us analyze this.

Fig. 51 to Fig. 57 shows boxplots between different roulettes. These shows that the use of aesthetic measures inside the roulette improves the aesthetics of the solutions. Also, minimize the number of outliers in some cases. Finally, the use of a weighted sum obtains very similar results. However, we prefer aesthetics-only roulette because the algorithm's architecture is cleaner.

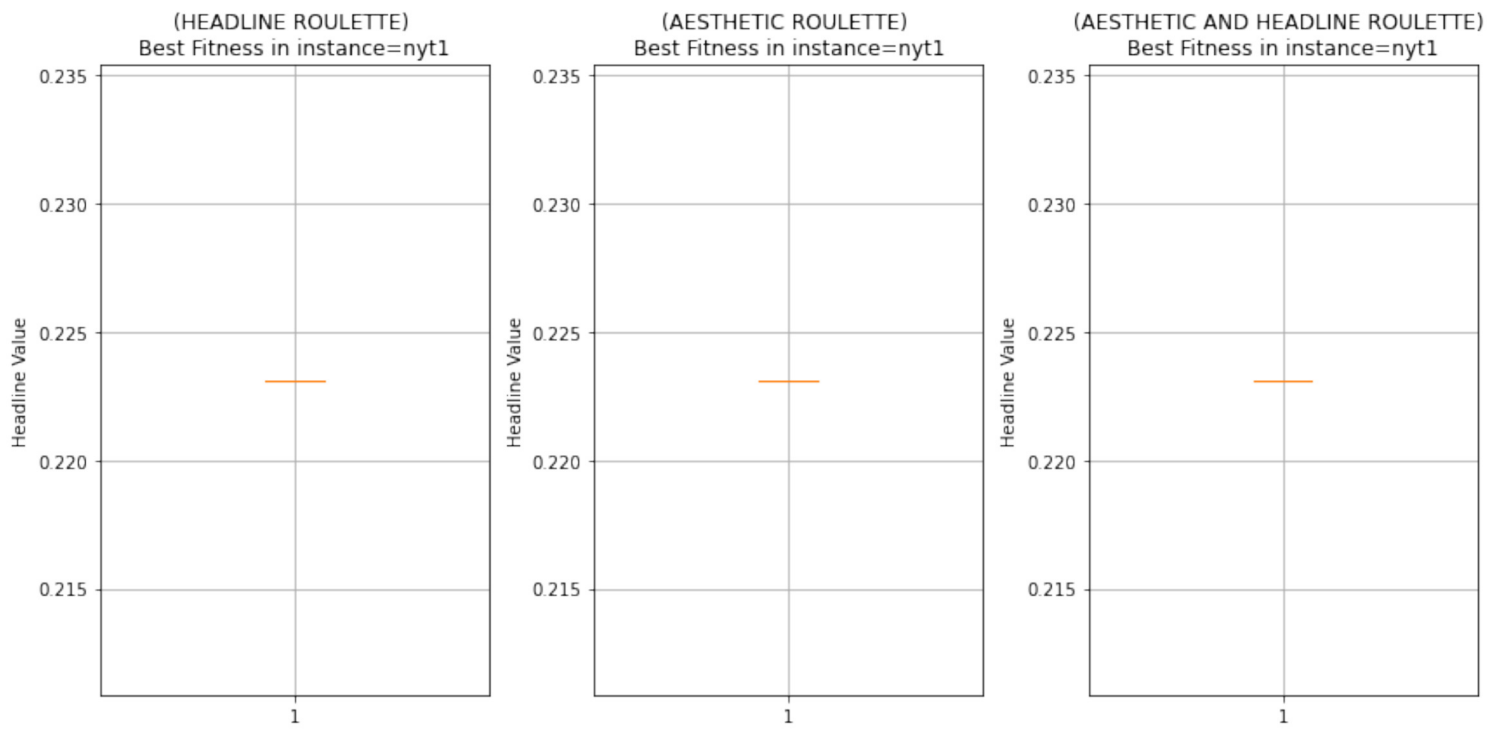


Figure 41: Exp 1: Comparison of Headline Value for instance nyt1

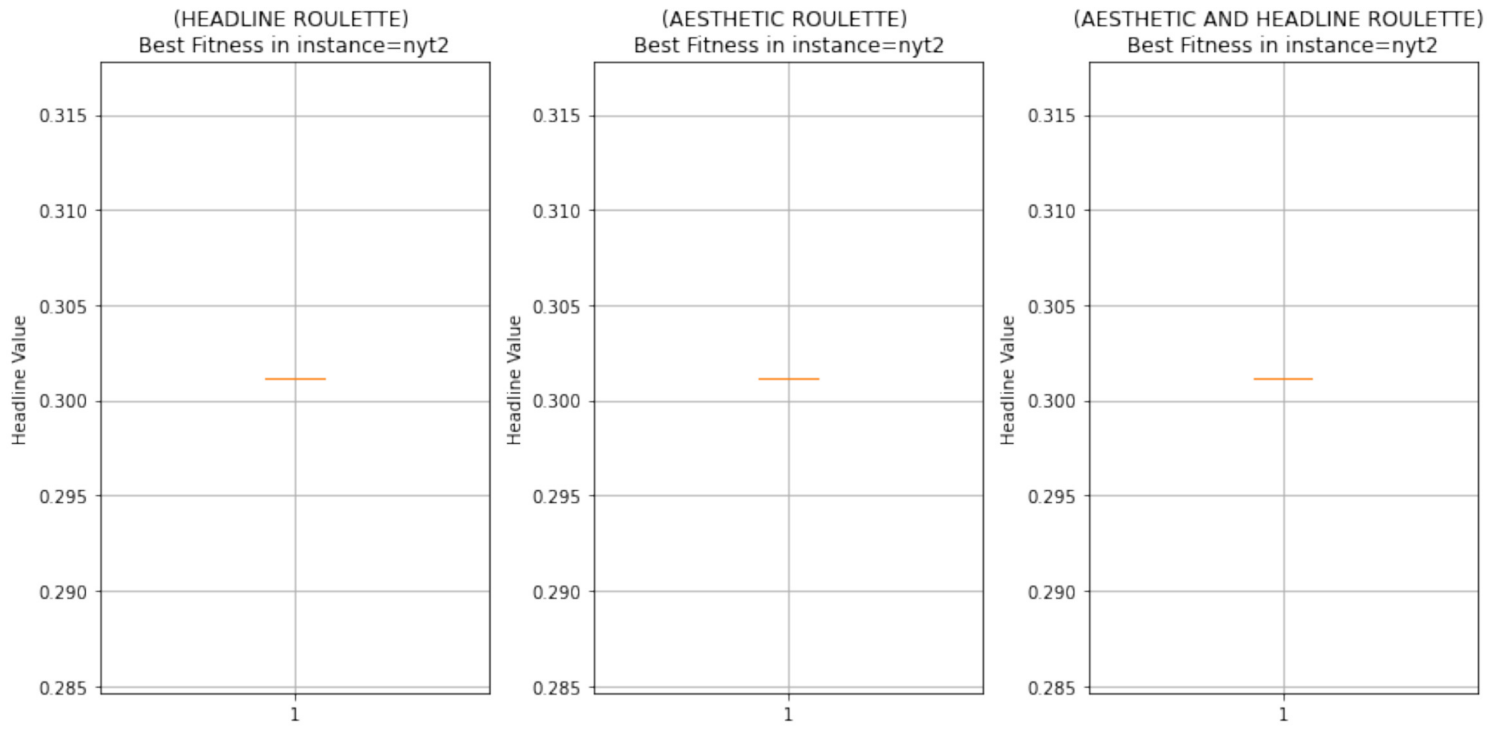


Figure 42: Exp 1: Comparison of Headline Value for instance nyt2

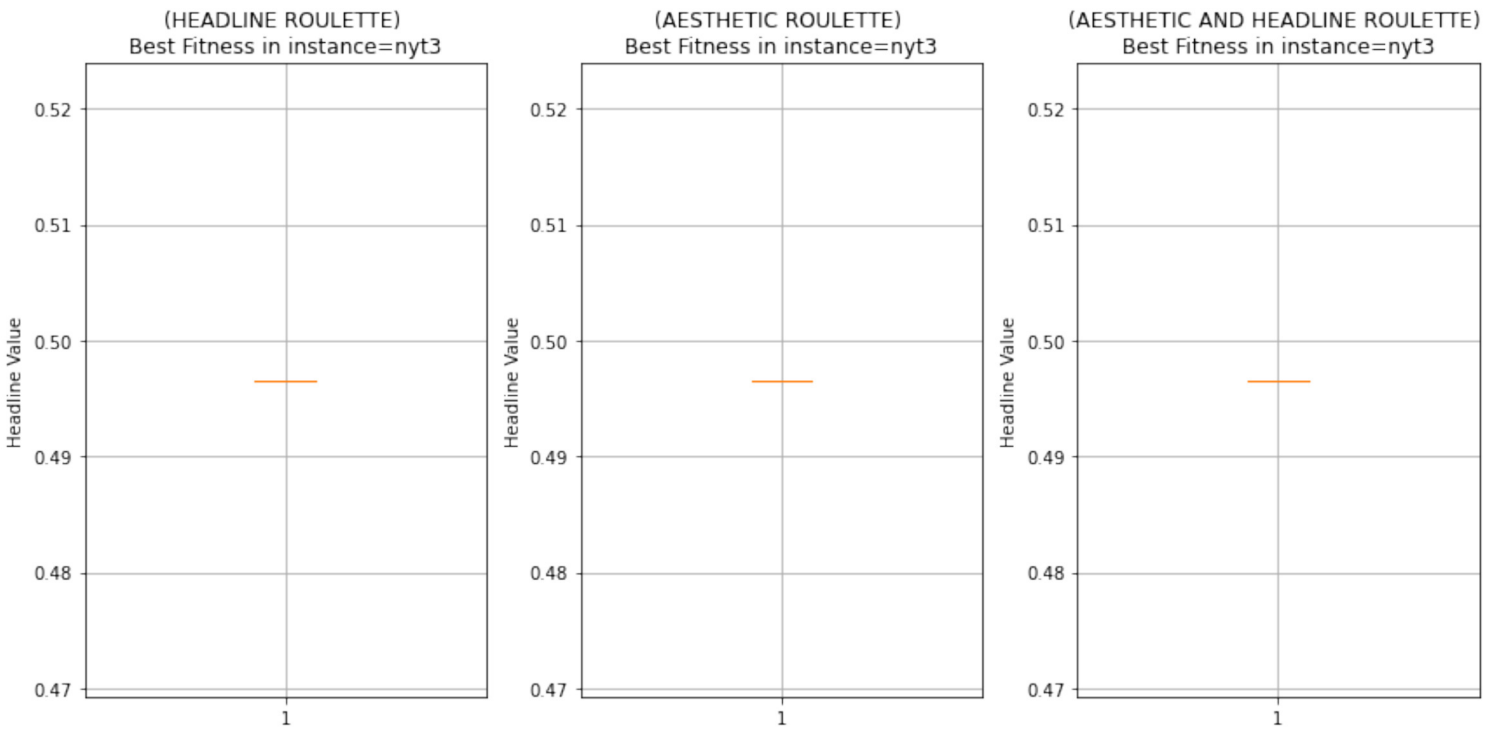


Figure 43: Exp 1: Comparison of Headline Value for instance nyt3

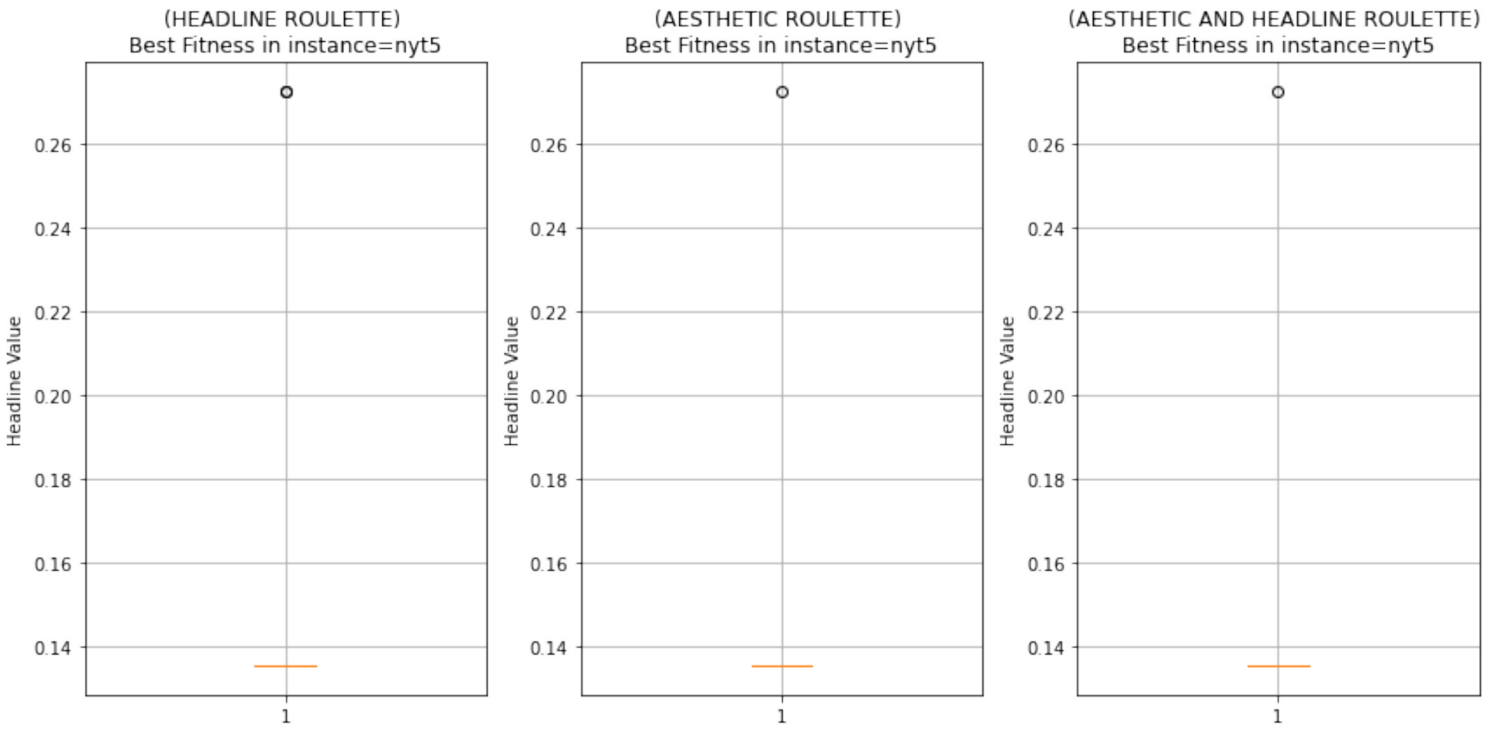


Figure 44: Exp 1: Comparison of Headline Value for instance nyt5

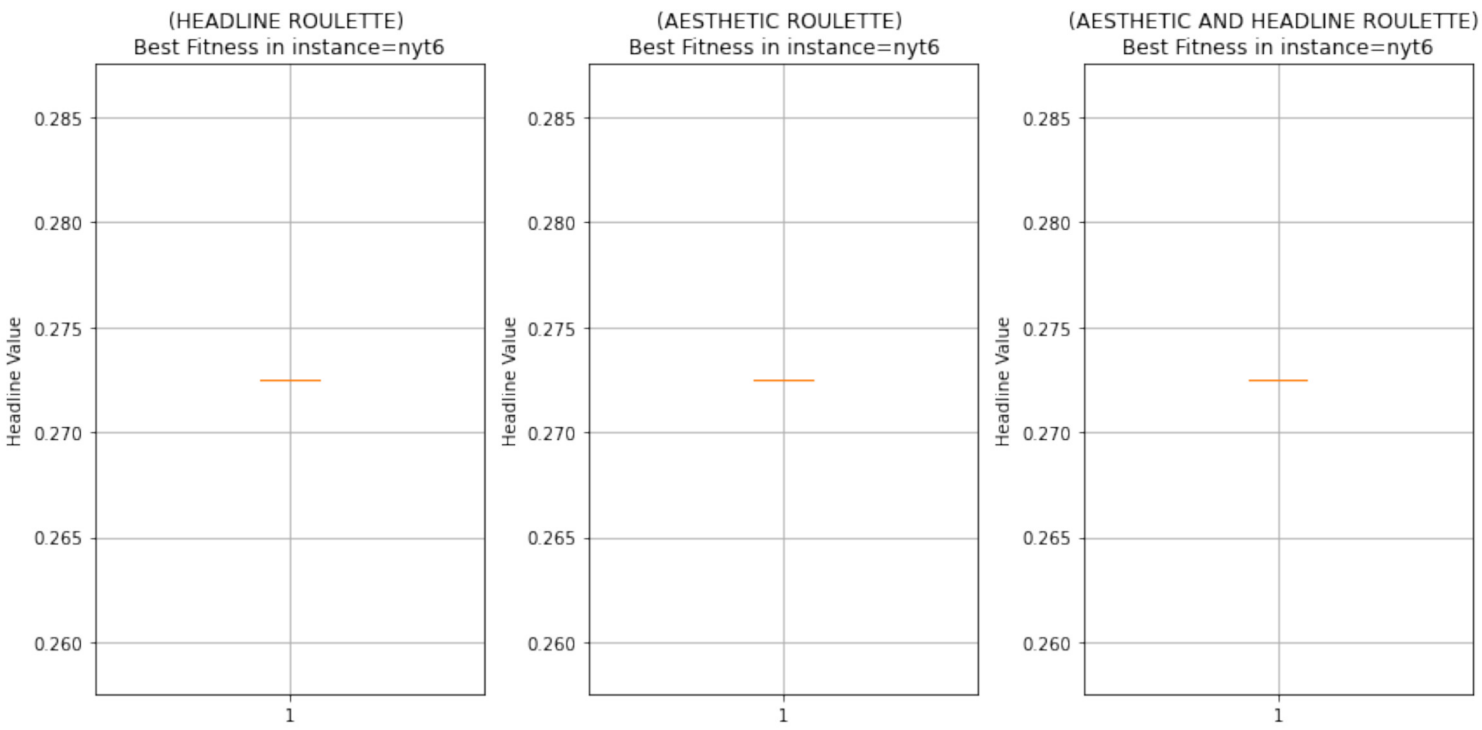


Figure 45: Exp 1: Comparison of Headline Value for instance nyt6

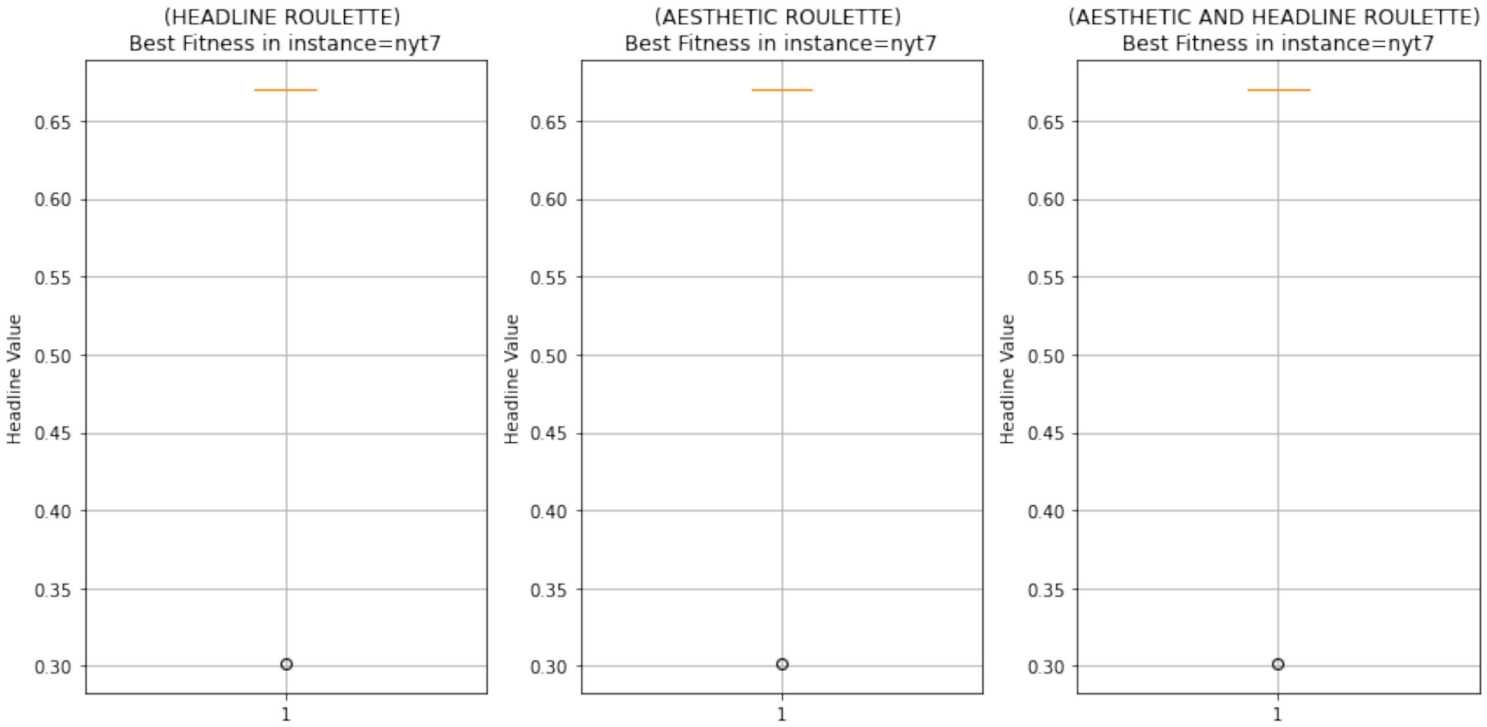


Figure 46: Exp 1: Comparison of Headline Value for instance nyt7

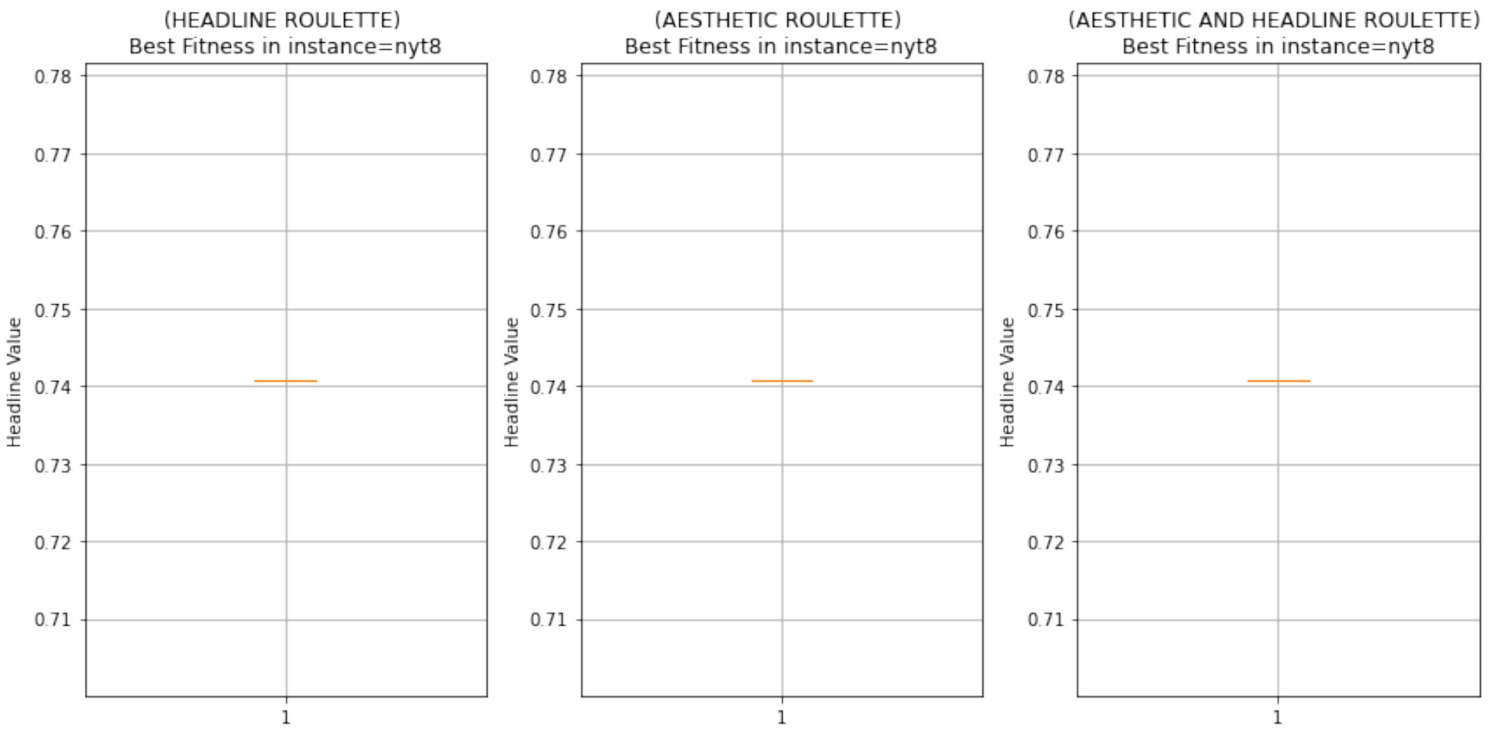


Figure 47: Exp 1: Comparison of Headline Value for instance nyt8

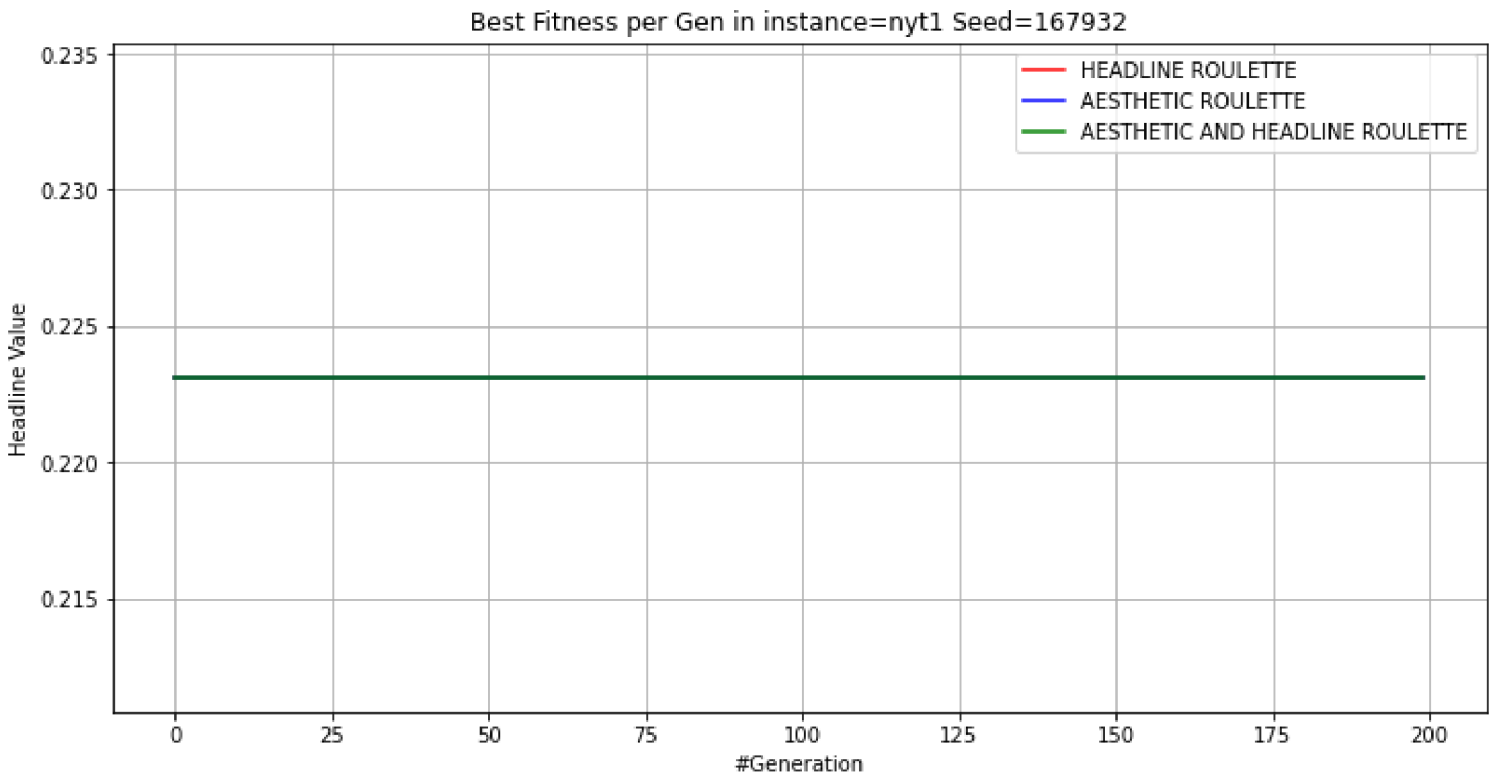


Figure 48: Exp 1: Best solution evolution across generations in instance nyt1. We can see that the algorithm found the best set of shapes at the initial generation of individuals.

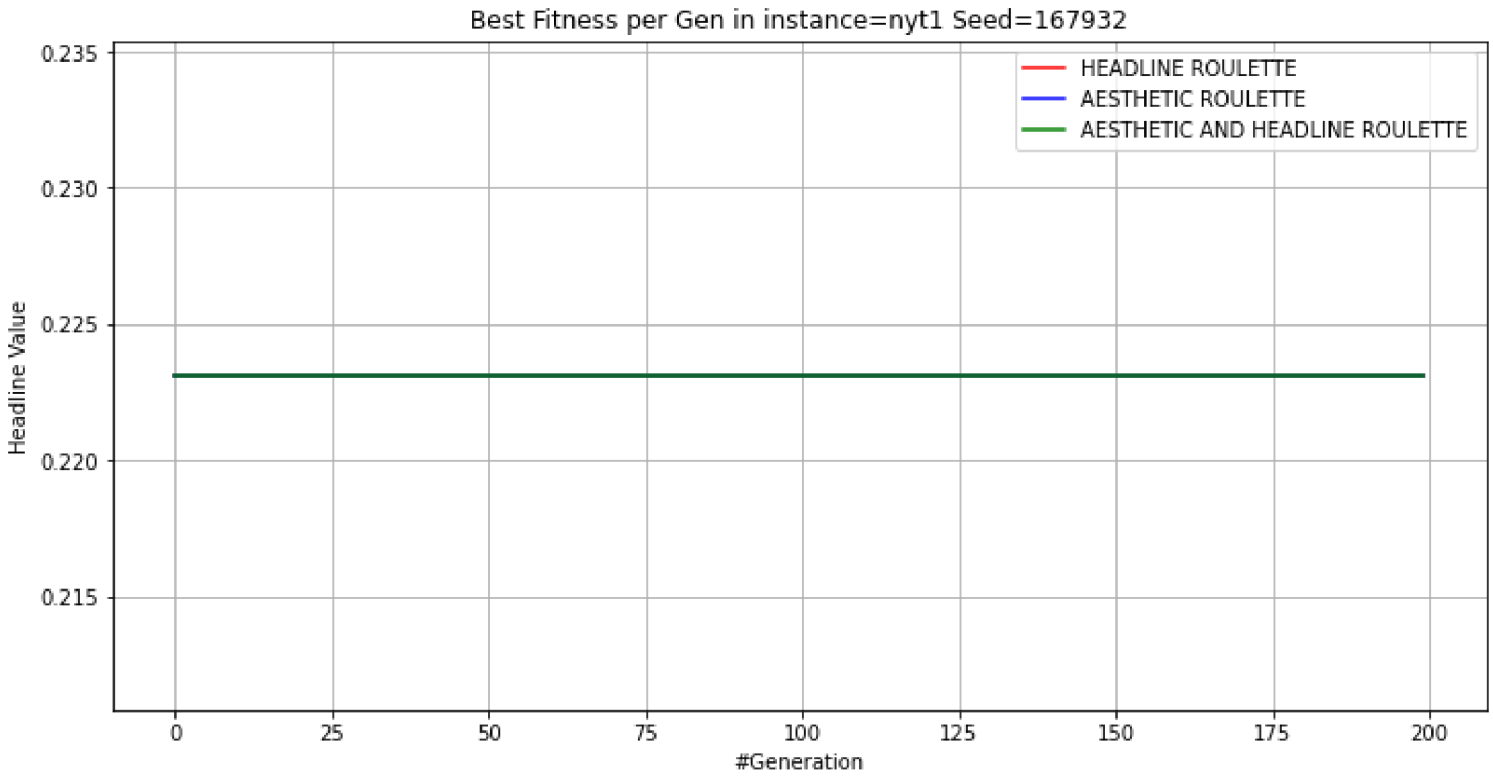


Figure 49: Worst solution evolution across generations in instance nyt1. We can see the worst solution converge quickly to the best one.

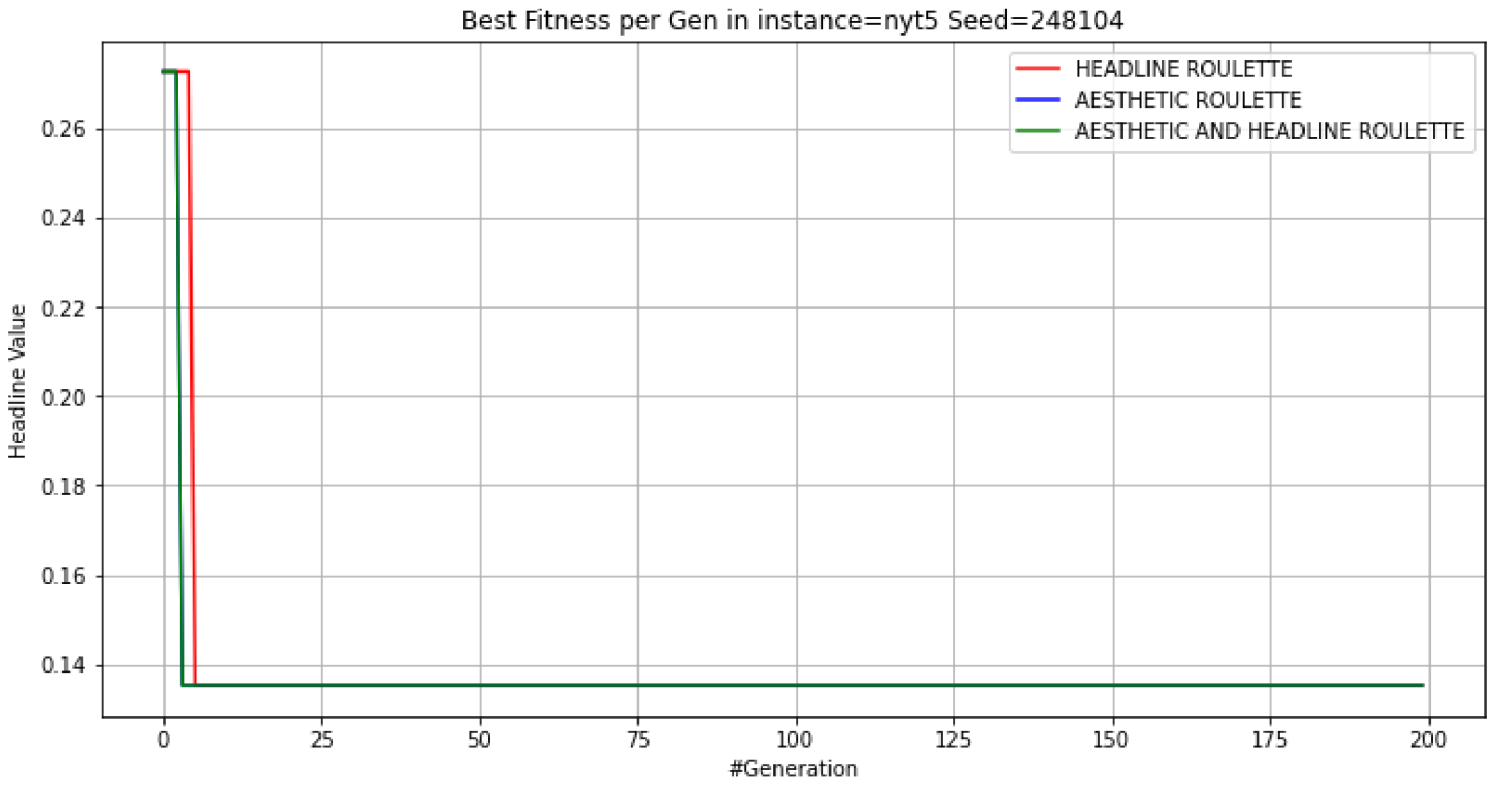


Figure 50: Exp 1: Best solution evolution across generations in instance nyt5. This shows the role of the decision tree; instead of choosing the best in terms of $H(.)$, chooses an example that is more homogeneous in terms of number of lines.

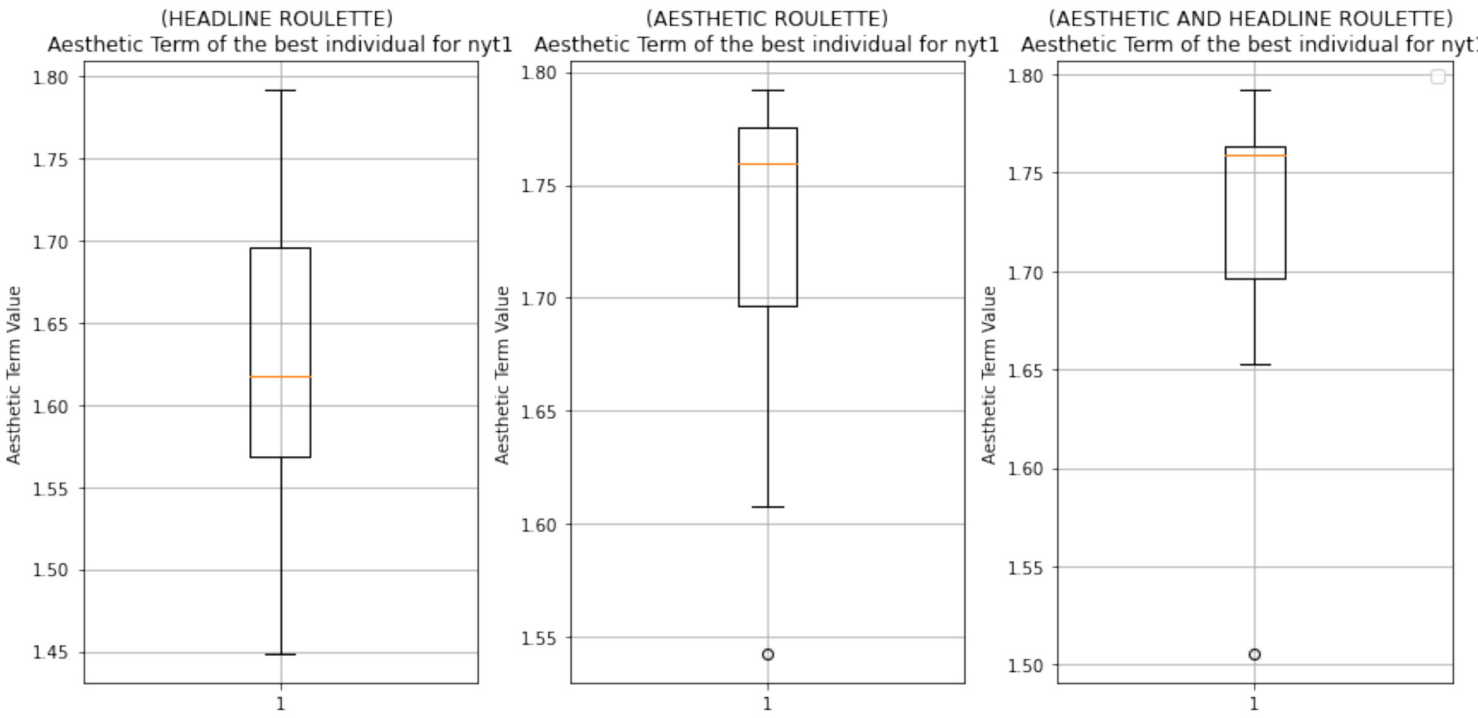


Figure 51: Exp 1: Comparison of Aesthetic Value for instance nyt1

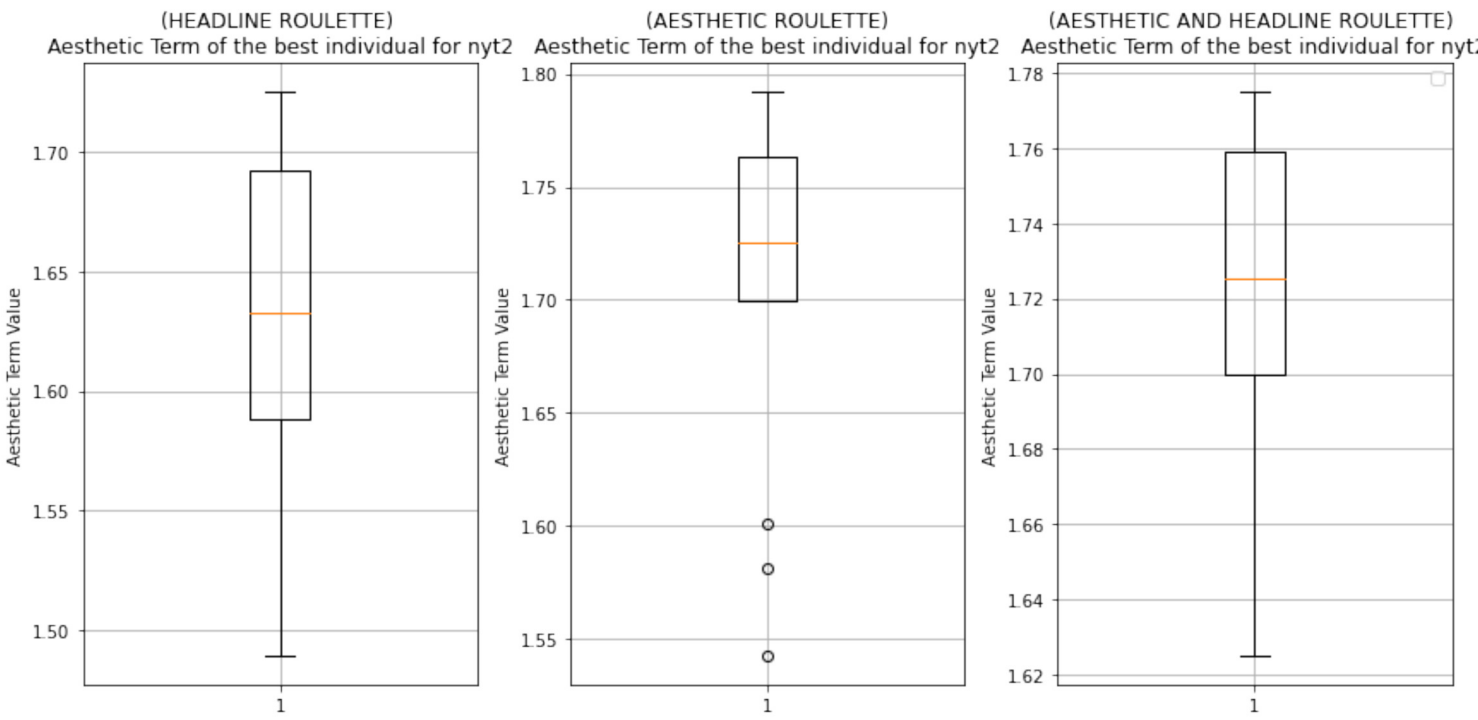


Figure 52: Exp 1: Comparison of Aesthetic Value for instance nyt2

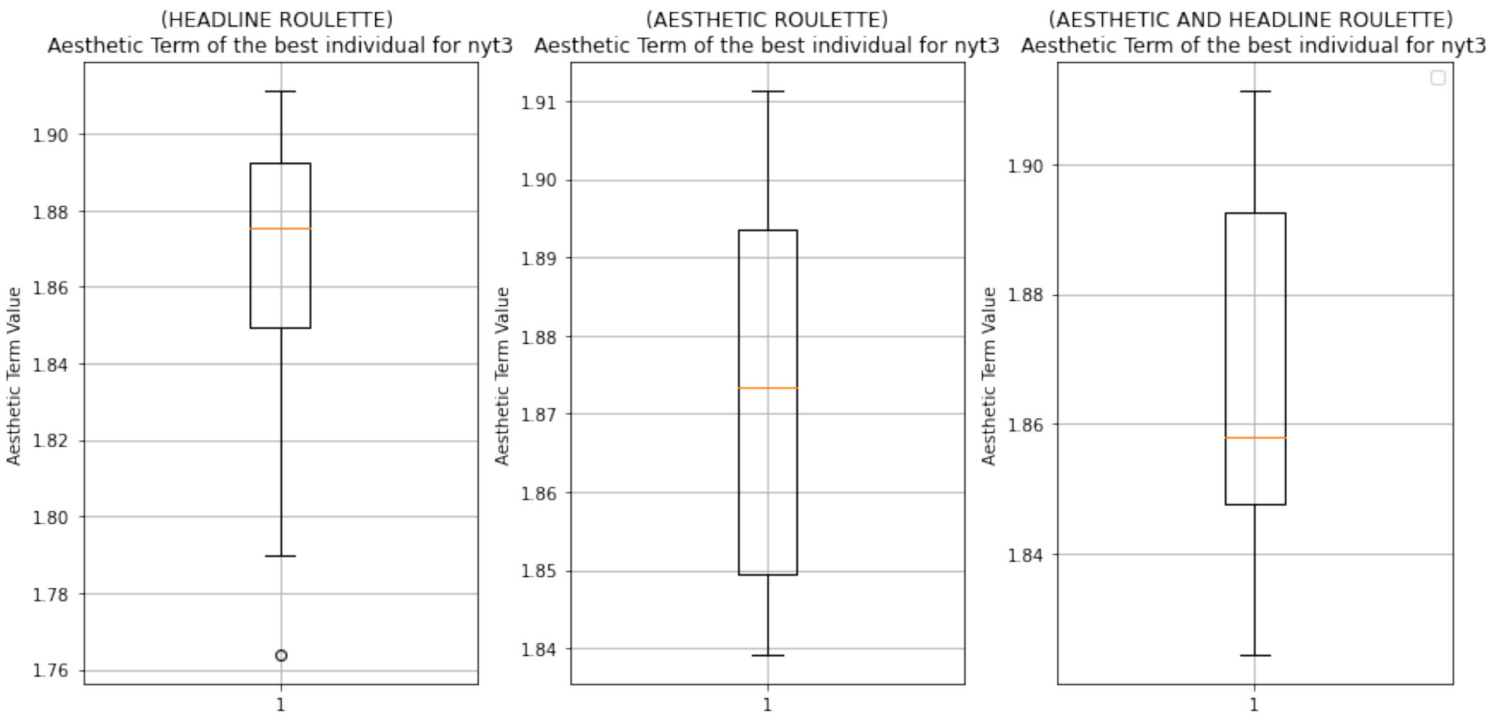


Figure 53: Exp 1: Comparison of Aesthetic Value for instance nyt3

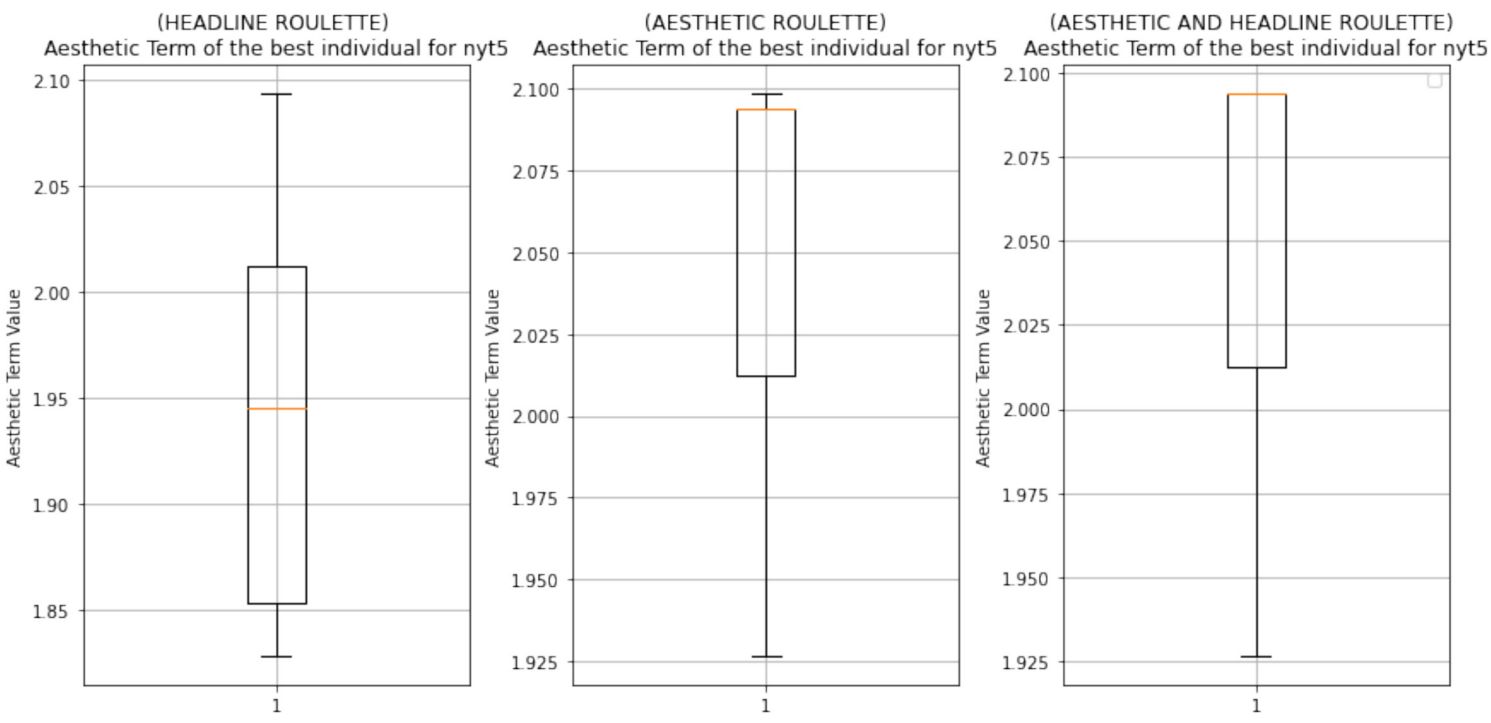


Figure 54: Exp 1: Comparison of Aesthetic Value for instance nyt5

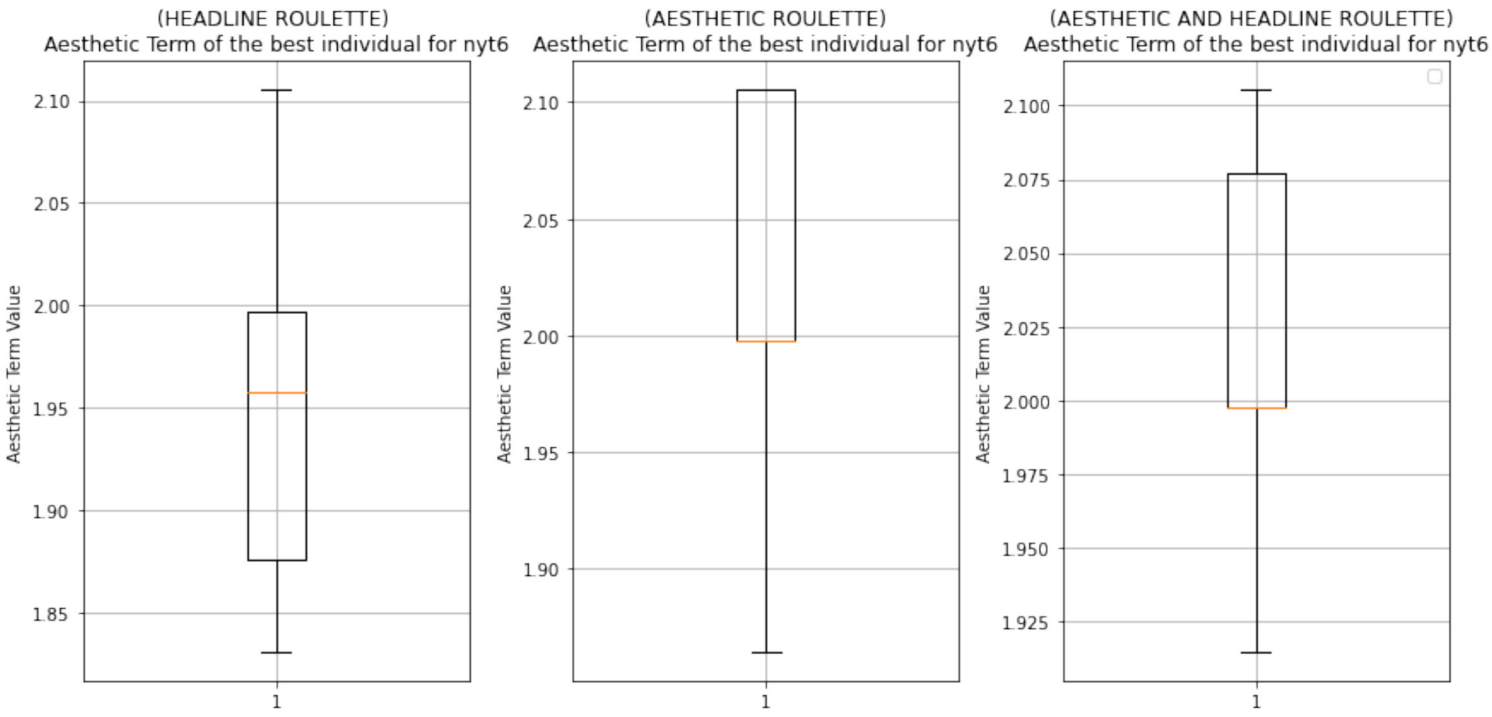


Figure 55: Exp 1: Comparison of Aesthetic Value for instance nyt6

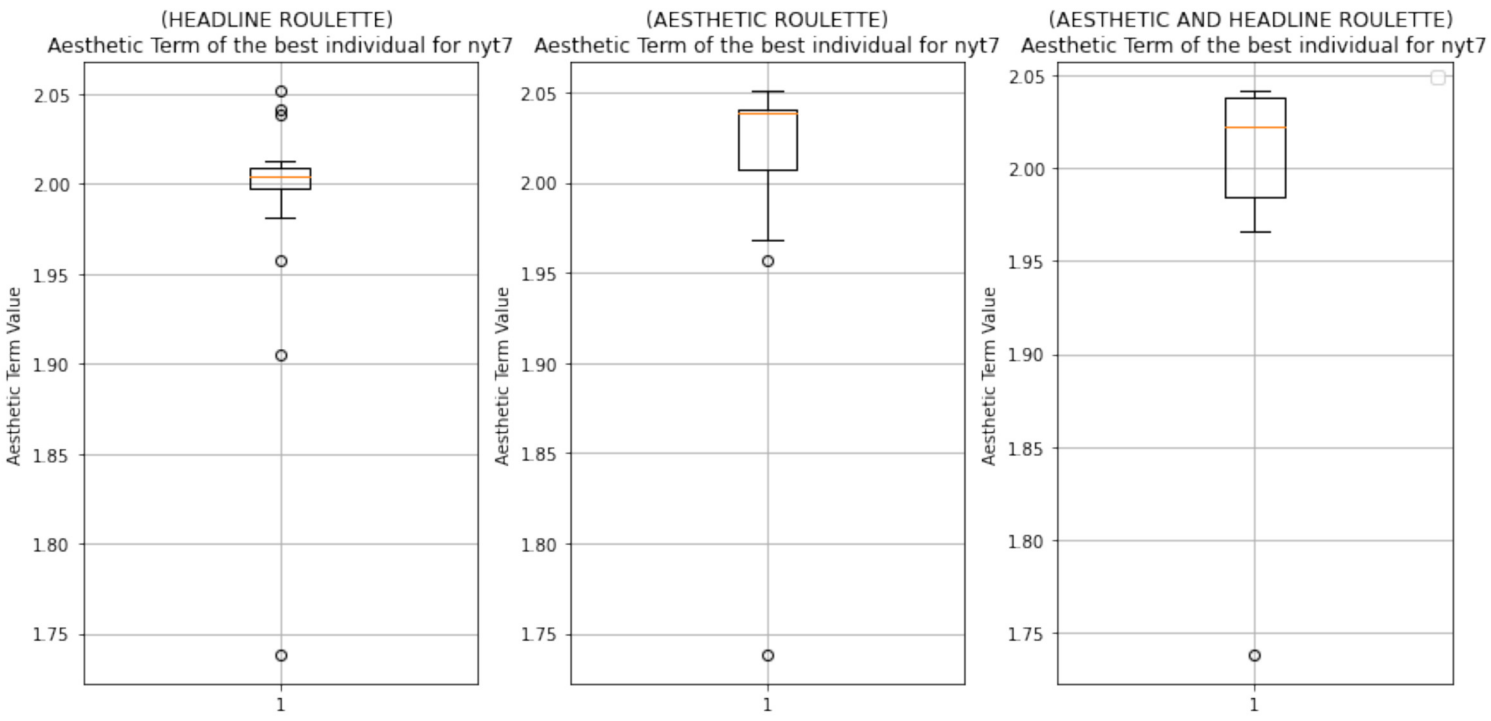


Figure 56: Exp 1: Comparison of Aesthetic Value for instance nyt7

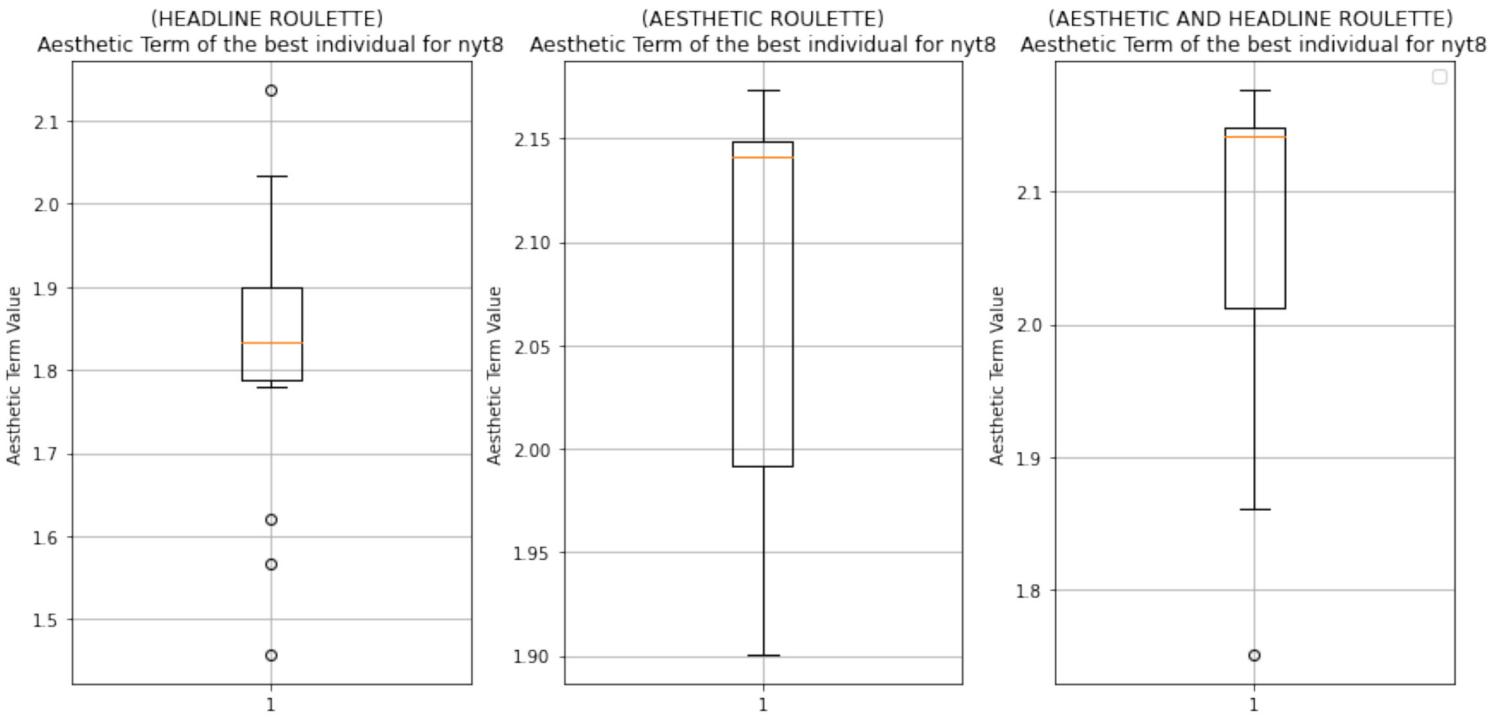


Figure 57: Exp 1: Comparison of Aesthetic Value for instance nyt8

4.6 Second Experiment: Comparing Genetic Algorithm Framework with different heuristics

Then, we test 3 different versions of the algorithm, fixing the heuristic in each version to iBL, BLF or BF. In all cases, we used **aesthetic-only roulette wheel**.

4.6.1 Headline Value Comparison

As we can see in Fig. 58 to Fig. 64, there are no differences between heuristics, in terms of $H(.)$. There are some outliers in *nyt2* (obtained because of the randomness of the roulette), and *nyt5* (obtained because the decision tree chose a different solution as the optimal, and that's why the median is located in an *apparent* worse value).

4.6.2 Aesthetic Value Comparison

In aesthetic terms, as we can see in Fig. 65 to Fig. 71, there are not considerable differences, explained by some differences in the placement using different heuristics.

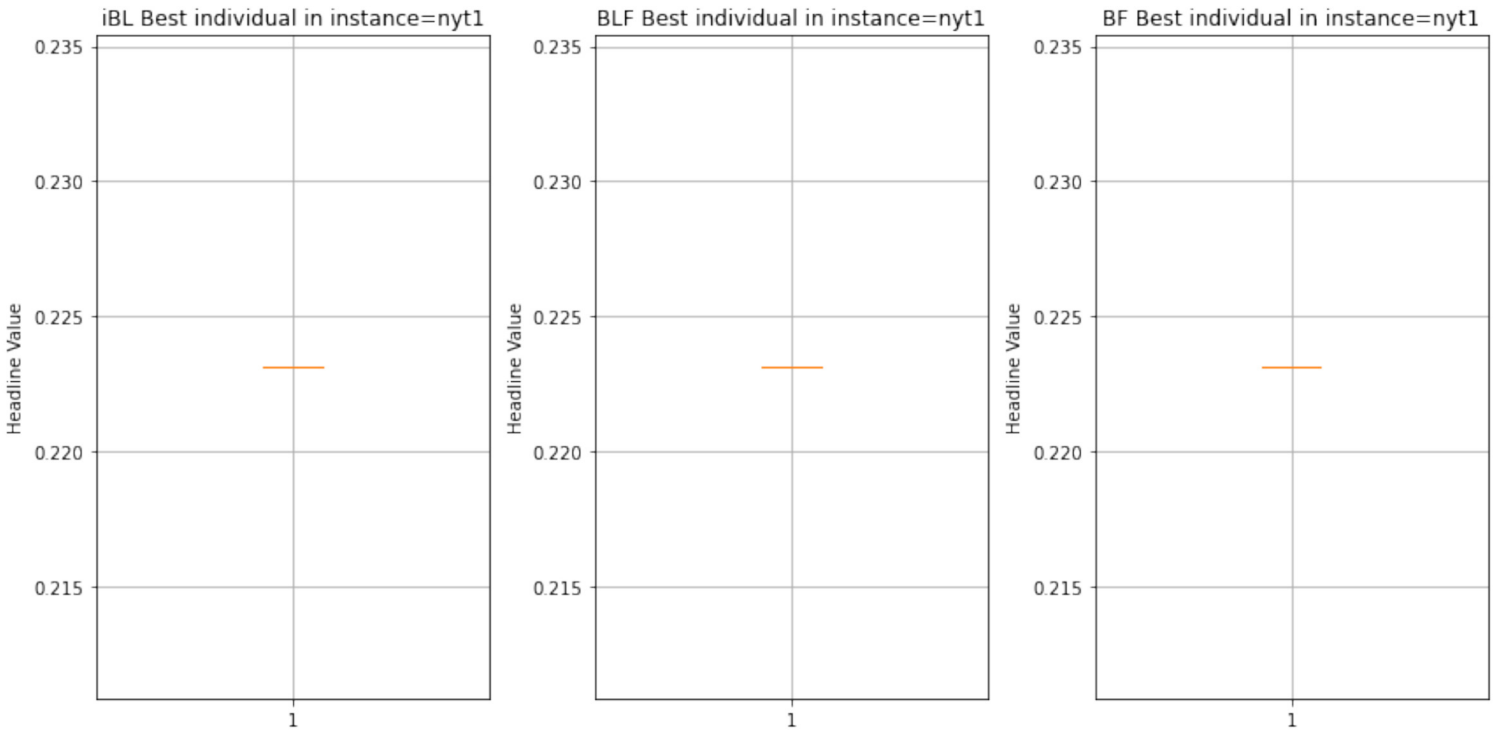


Figure 58: Exp 2: Comparison of Headline Value for instance nyt1

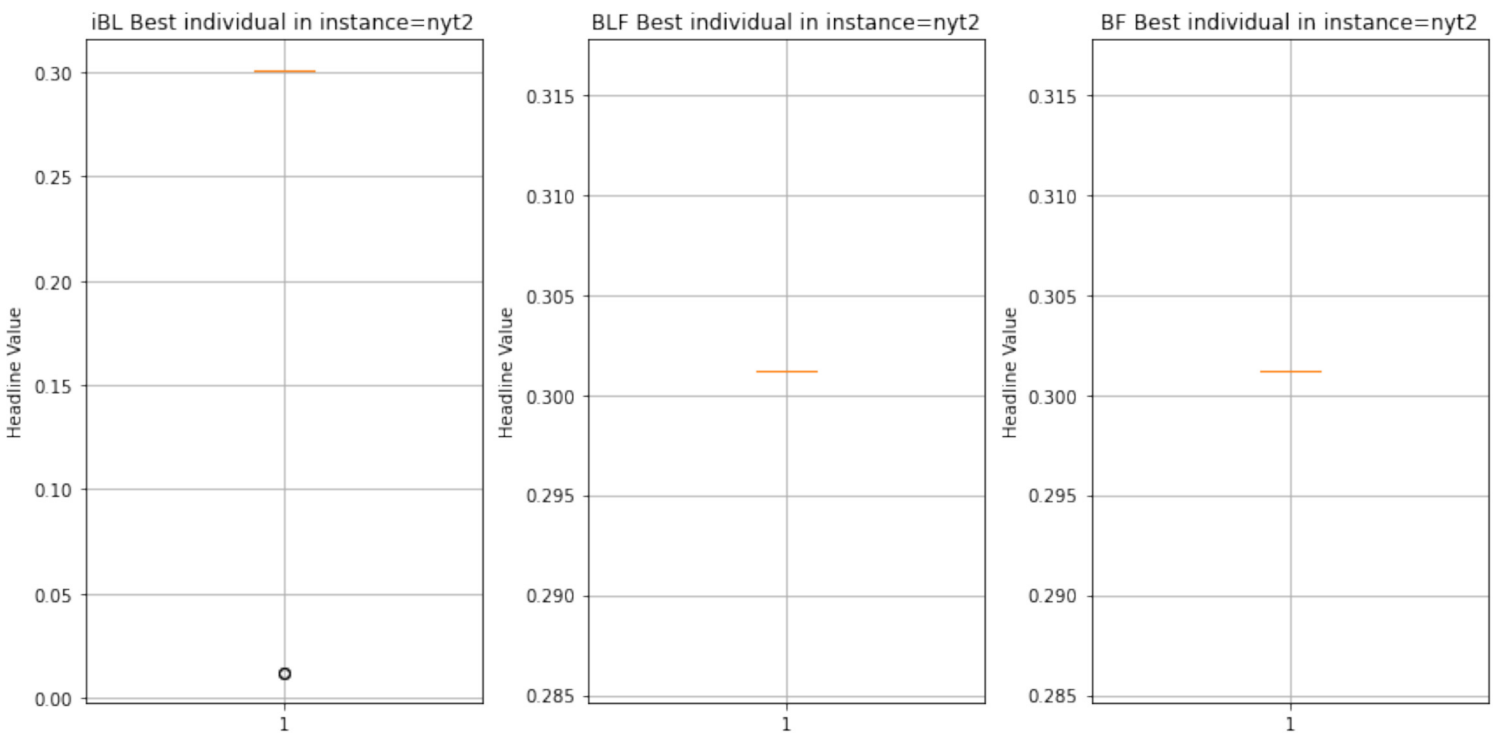


Figure 59: Exp 2: Comparison of Headline Value for instance nyt2

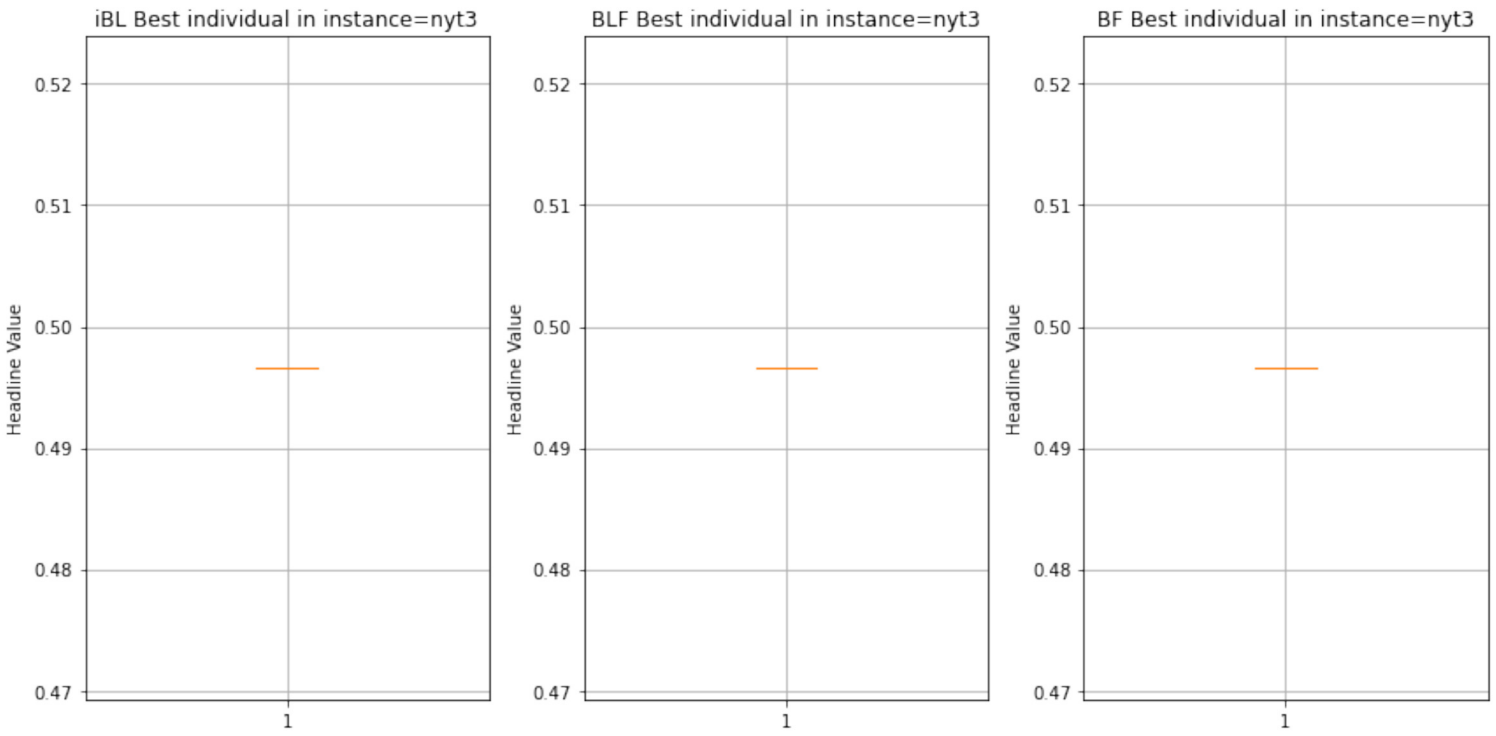


Figure 60: Exp 2: Comparison of Headline Value for instance nyt3

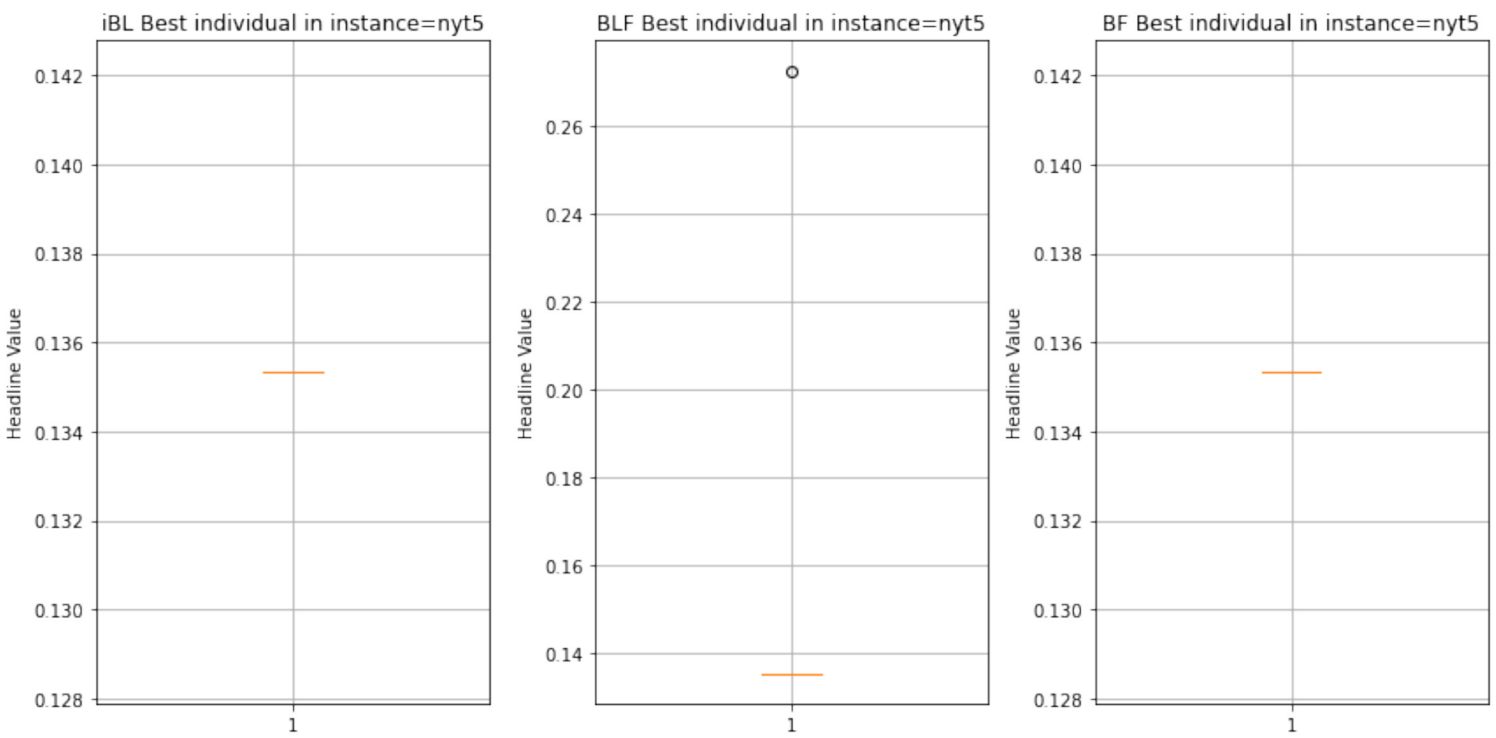


Figure 61: Exp 2: Comparison of Headline Value for instance nyt5

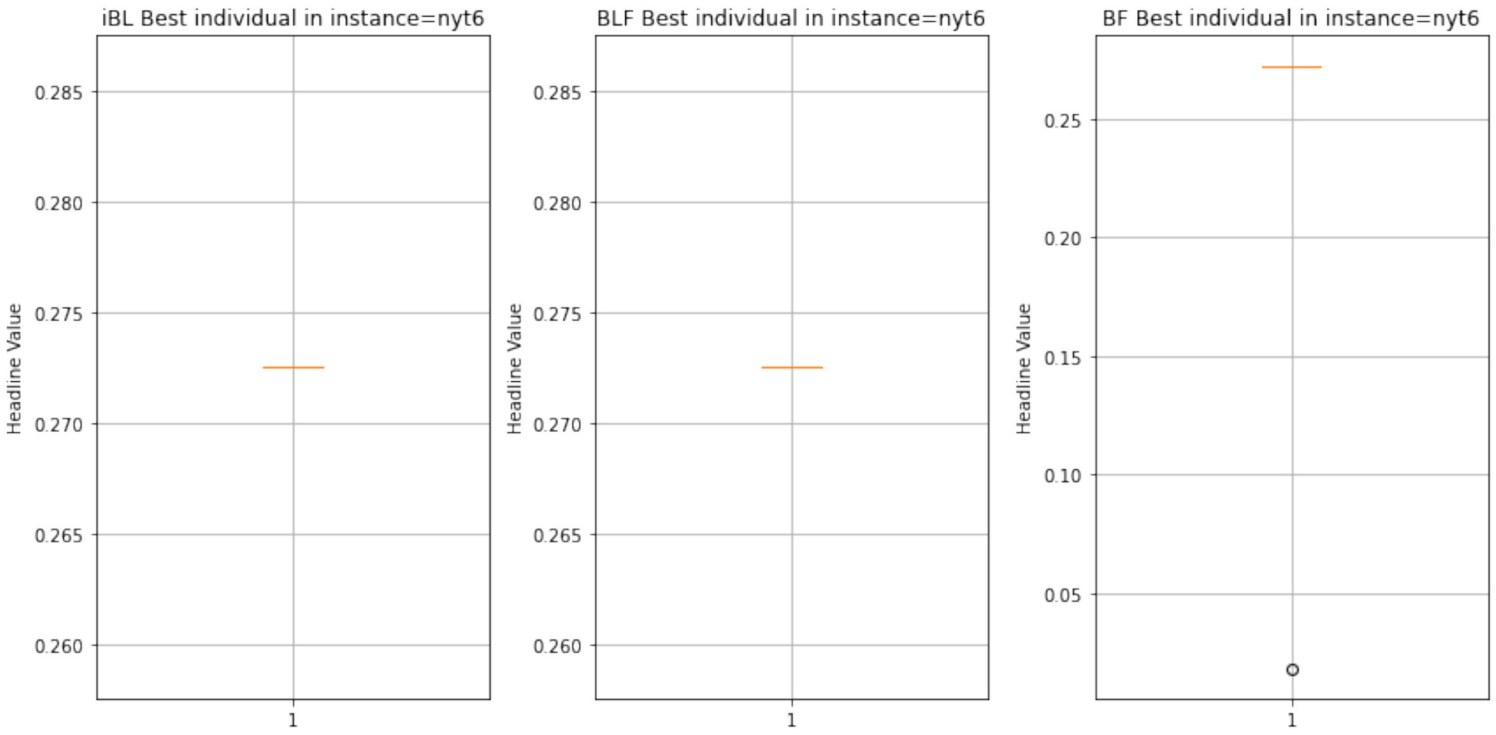


Figure 62: Exp 2: Comparison of Headline Value for instance nyt6

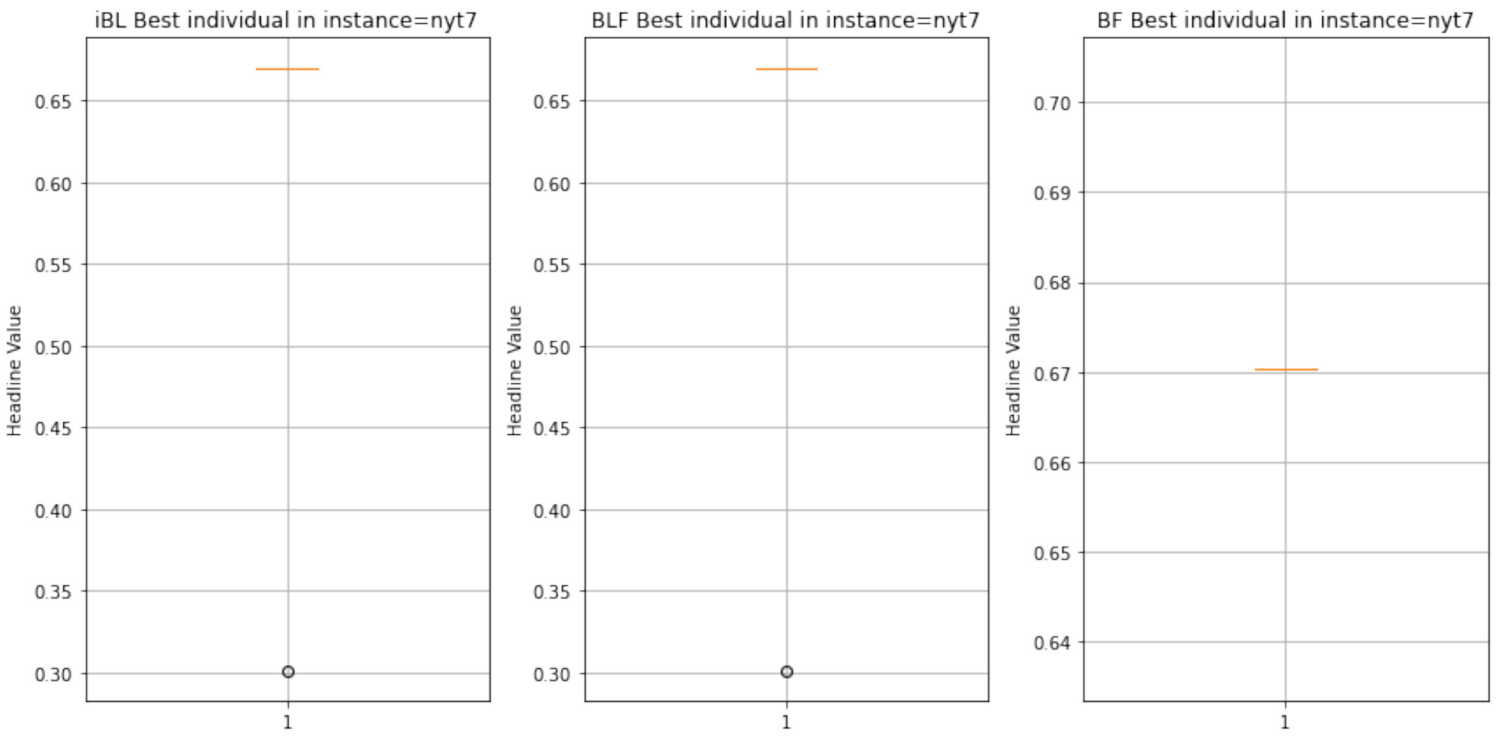


Figure 63: Exp 2: Comparison of Headline Value for instance nyt7

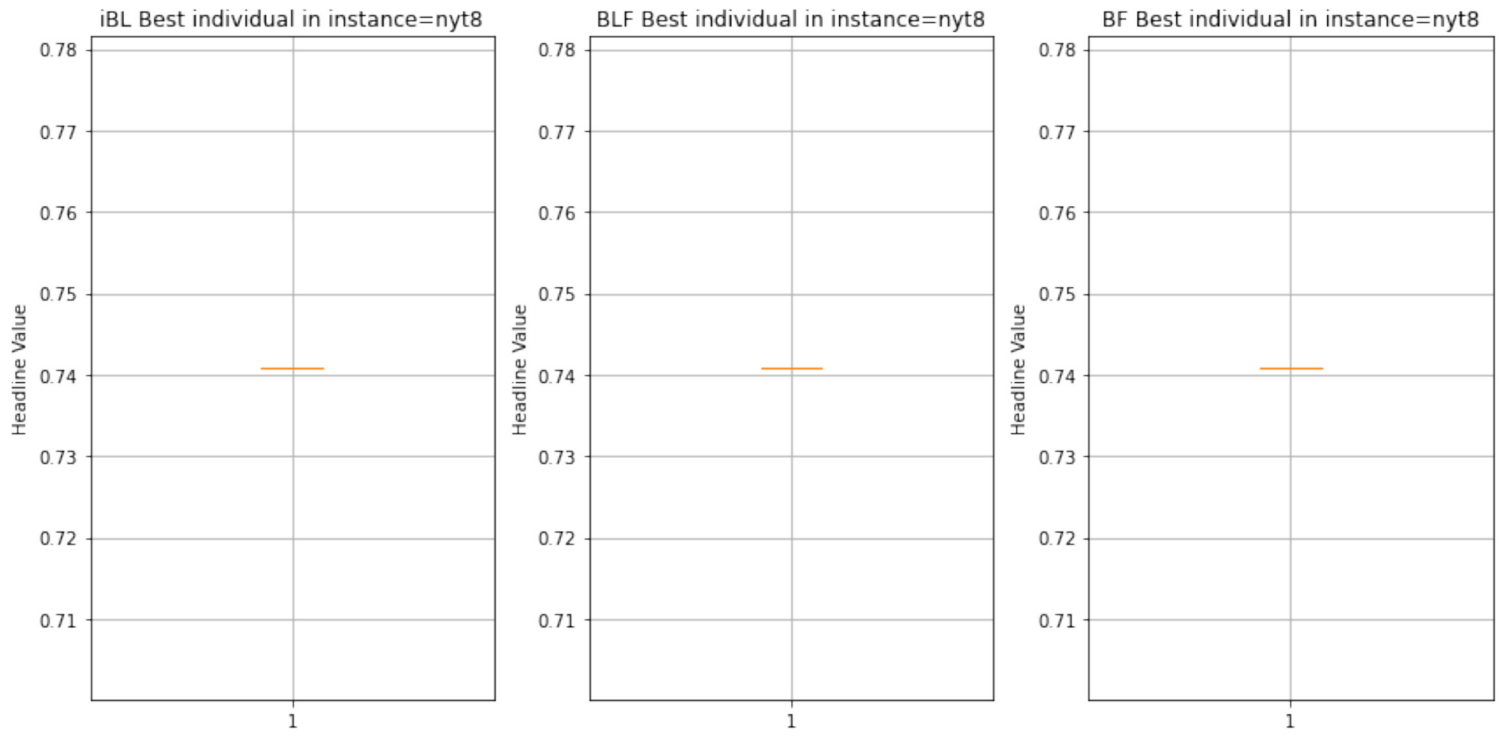


Figure 64: Exp 2: Comparison of Headline Value for instance nyt8

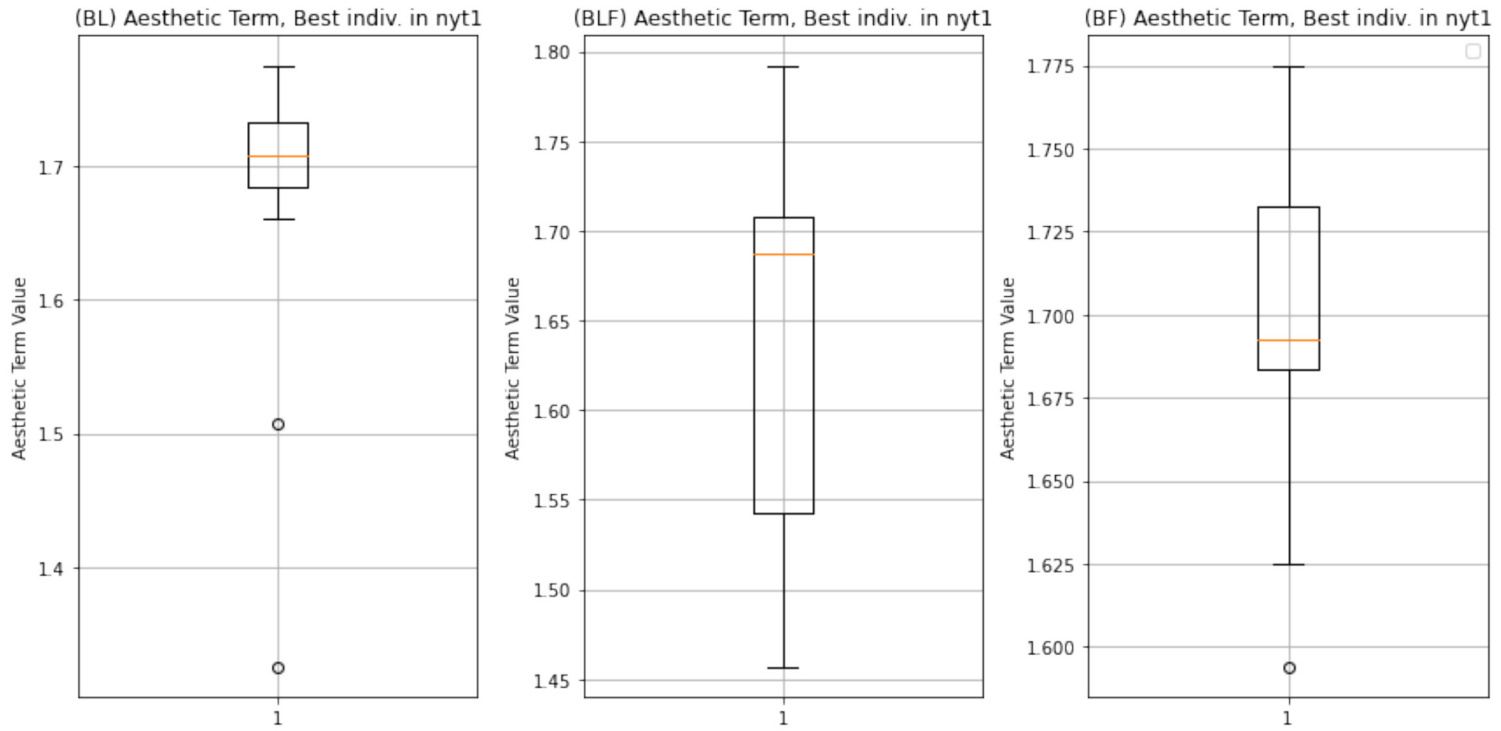


Figure 65: Exp 2:Comparison of Aesthetic Value for instance nyt1

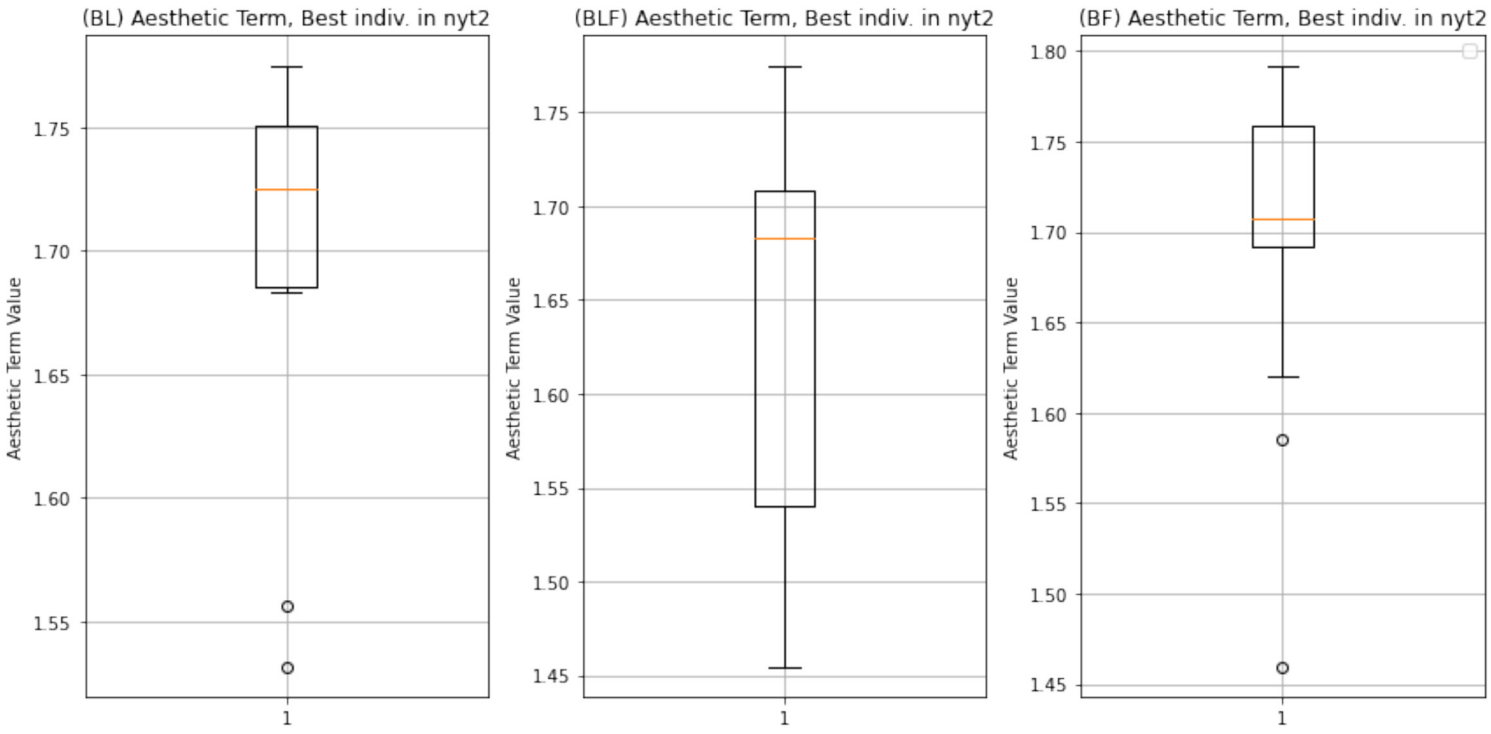


Figure 66: Exp 2: Comparison of Aesthetic Value for instance nyt2

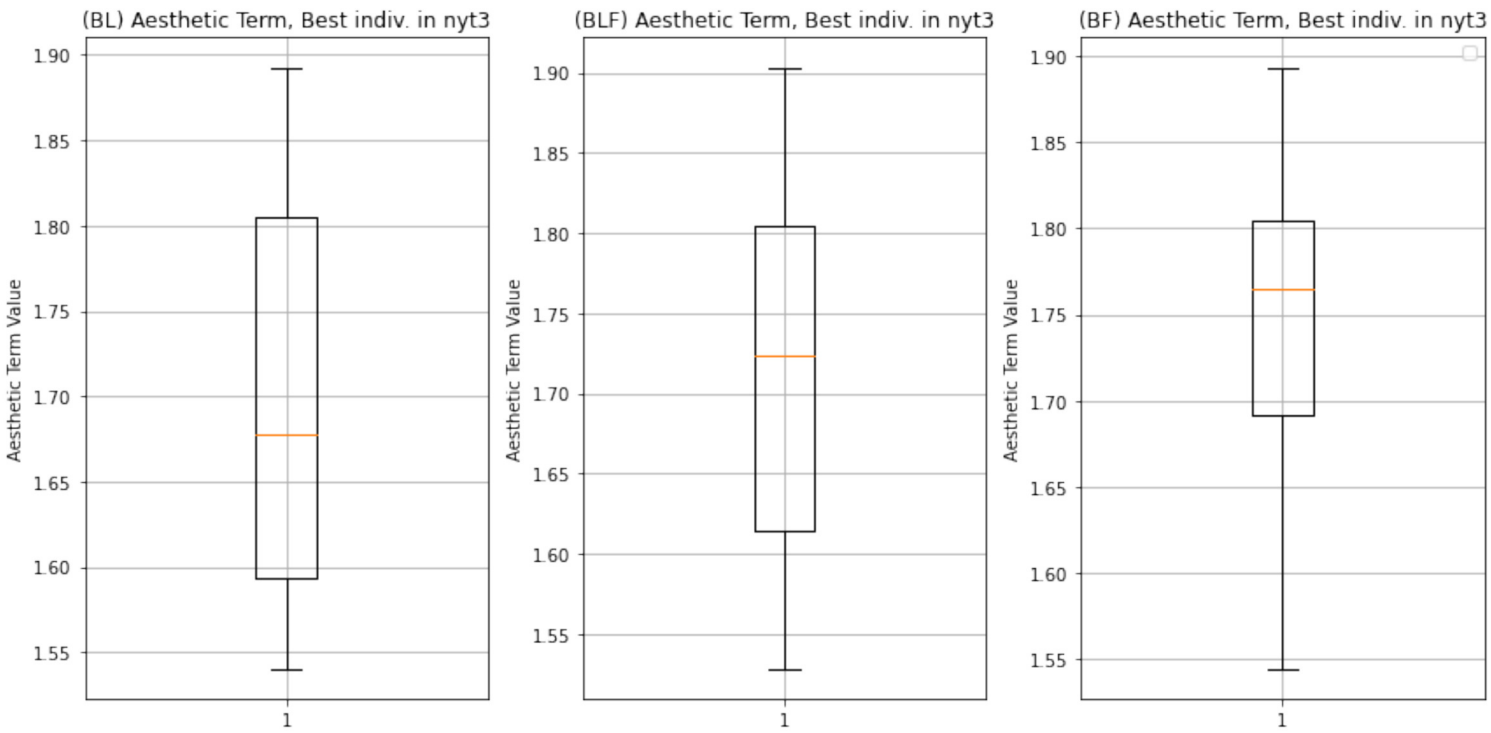


Figure 67: Exp 2: Comparison of Aesthetic Value for instance nyt3

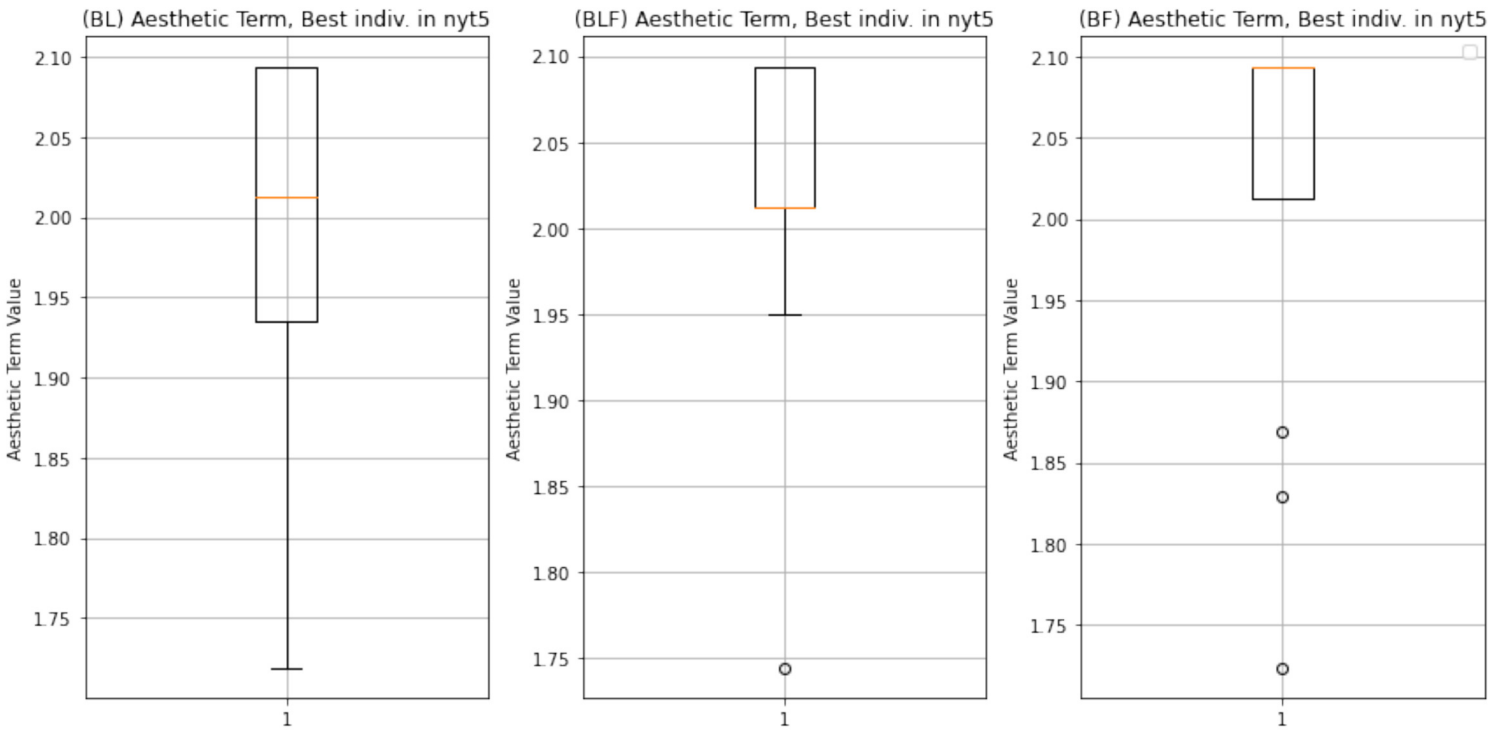


Figure 68: Exp 2: Comparison of Aesthetic Value for instance nyt5

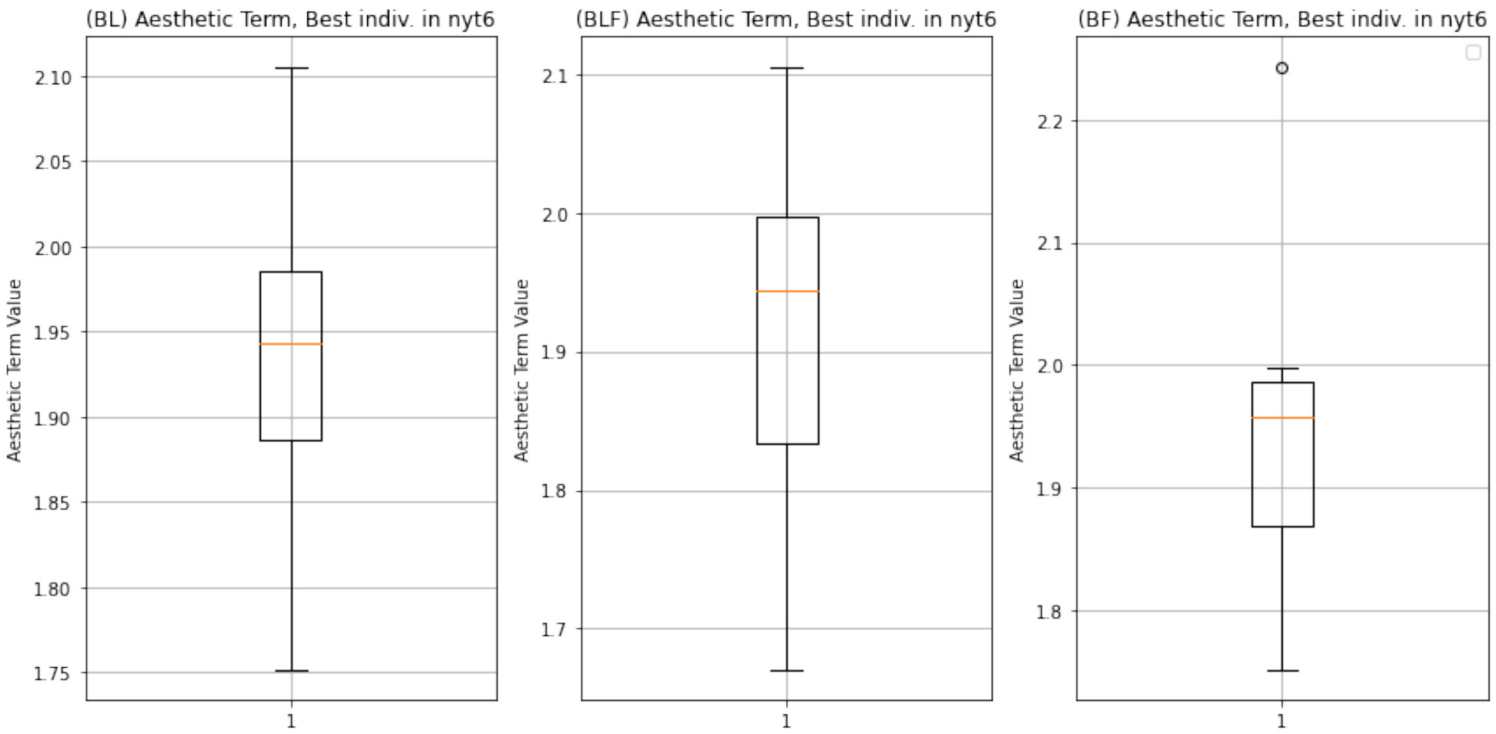


Figure 69: Exp 2: Comparison of Aesthetic Value for instance nyt6

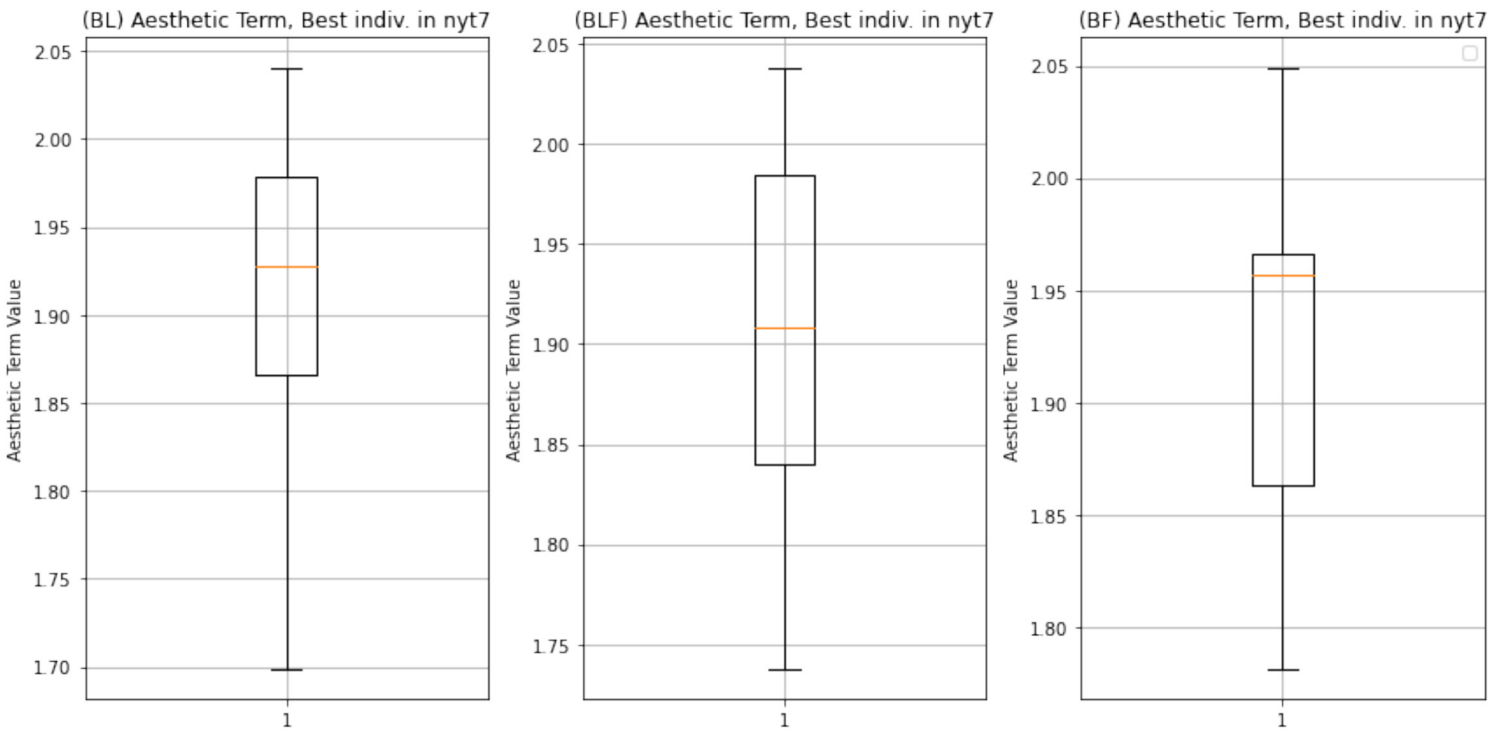


Figure 70: Exp 2: Comparison of Aesthetic Value for instance nyt7

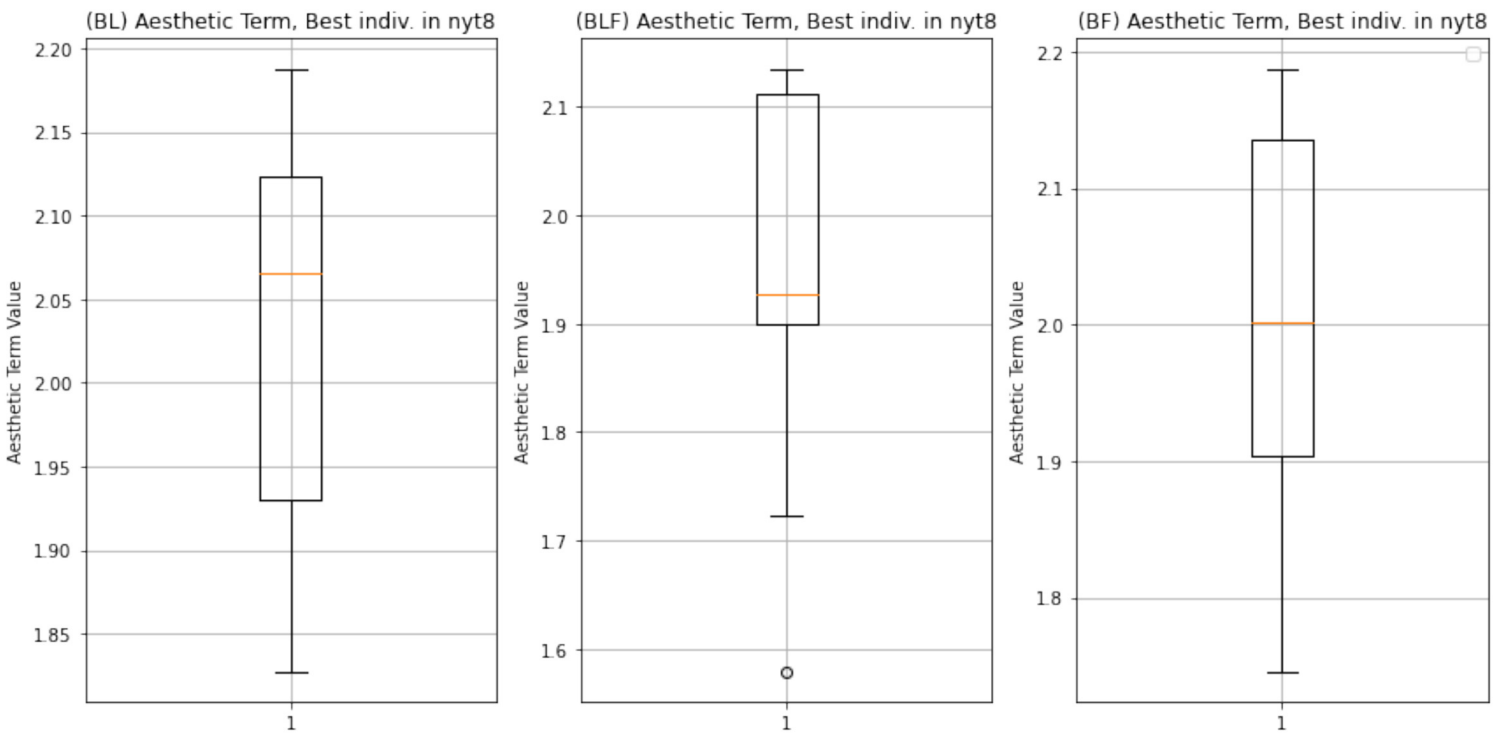


Figure 71: Exp 2: Comparison of Aesthetic Value for instance nyt8

4.6.3 Execution Time

In this topic we see a lot of differences: in Fig. 72 we can see a considerable difference in execution time. iBL is, in theory, slower and limited because it is not capable to fill *holes* generated in the packing process, something that BLF and BF can. Given that, the algorithm has to search for a longer time for a feasible permutation using iBL. In the graph, we can see the impact of this behavior, specially in *nyt2* and *nyt6*. BF is slower than BLF because it must search for the best space in each placement, something that BLF does not need.

In conclusion, BLF can be considered as the best one; however, something nice about the design of our algorithm is that it can use any heuristic without any impact in the quality of the solution.

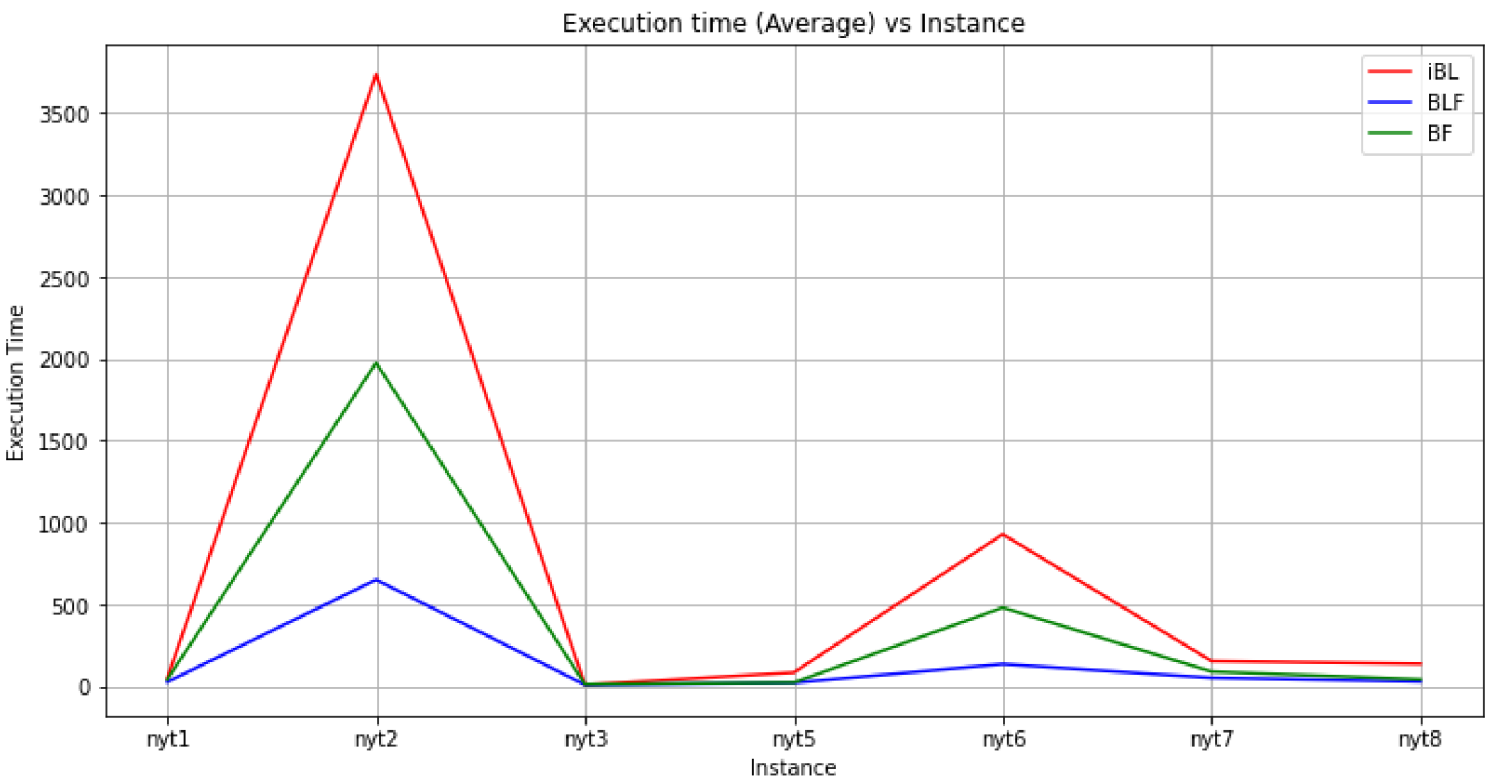


Figure 72: Exp 2: Time (average) vs instance, comparison between heuristics. We can see that iBL is considerably slower than the other 2, suggesting BLF as the best in general.

4.7 Third Experiment: Comparing Genetic Algorithm Framework with online choose of heuristics

Finally, we check the differences in quality between using BLF heuristic with the online version of the method; that means, including the heuristic inside each chromosome and allow mutation of this variable.

4.7.1 Headline Value Comparison

Results in Fig. 73 to Fig. 79 show that, in terms of $H(.)$, there are not considerable differences.

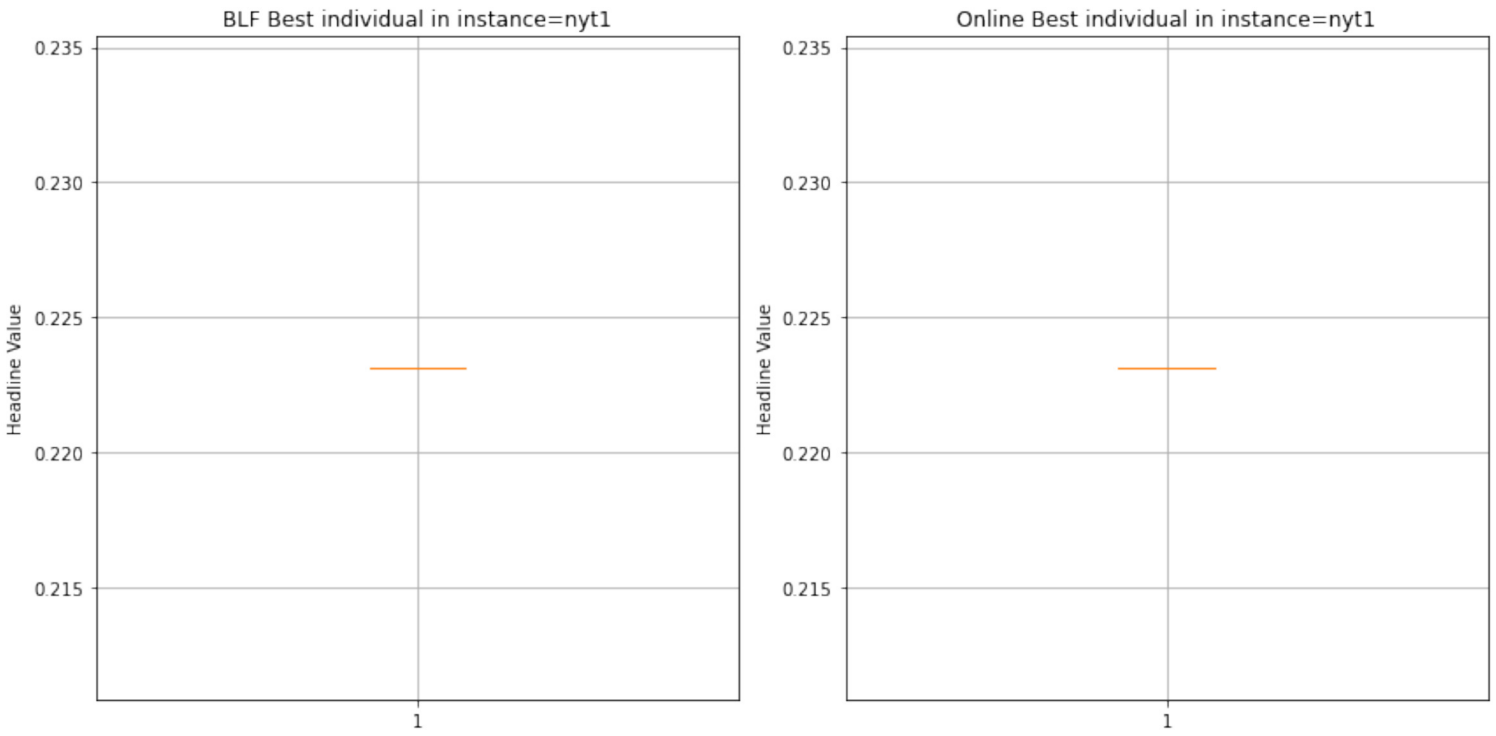


Figure 73: Exp 3: Comparison of Headline Value for instance nyt1

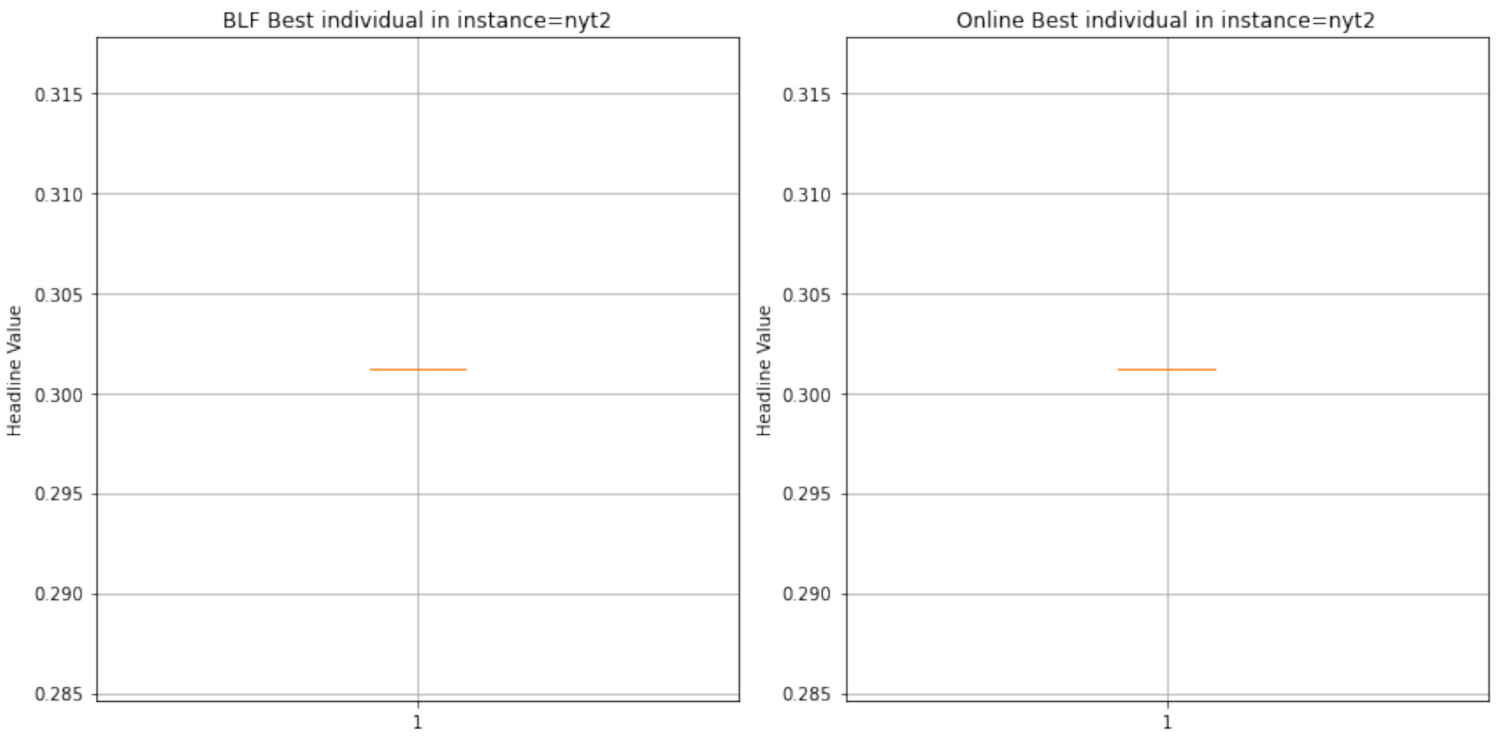


Figure 74: Exp 3: Comparison of Headline Value for instance nyt2

4.7.2 Aesthetic Value Comparison

Results in Fig. 80 to Fig. 86 show some differences; one of the variables that can explain this is that the pseudo-random number generator is called a lot more in the online case, because

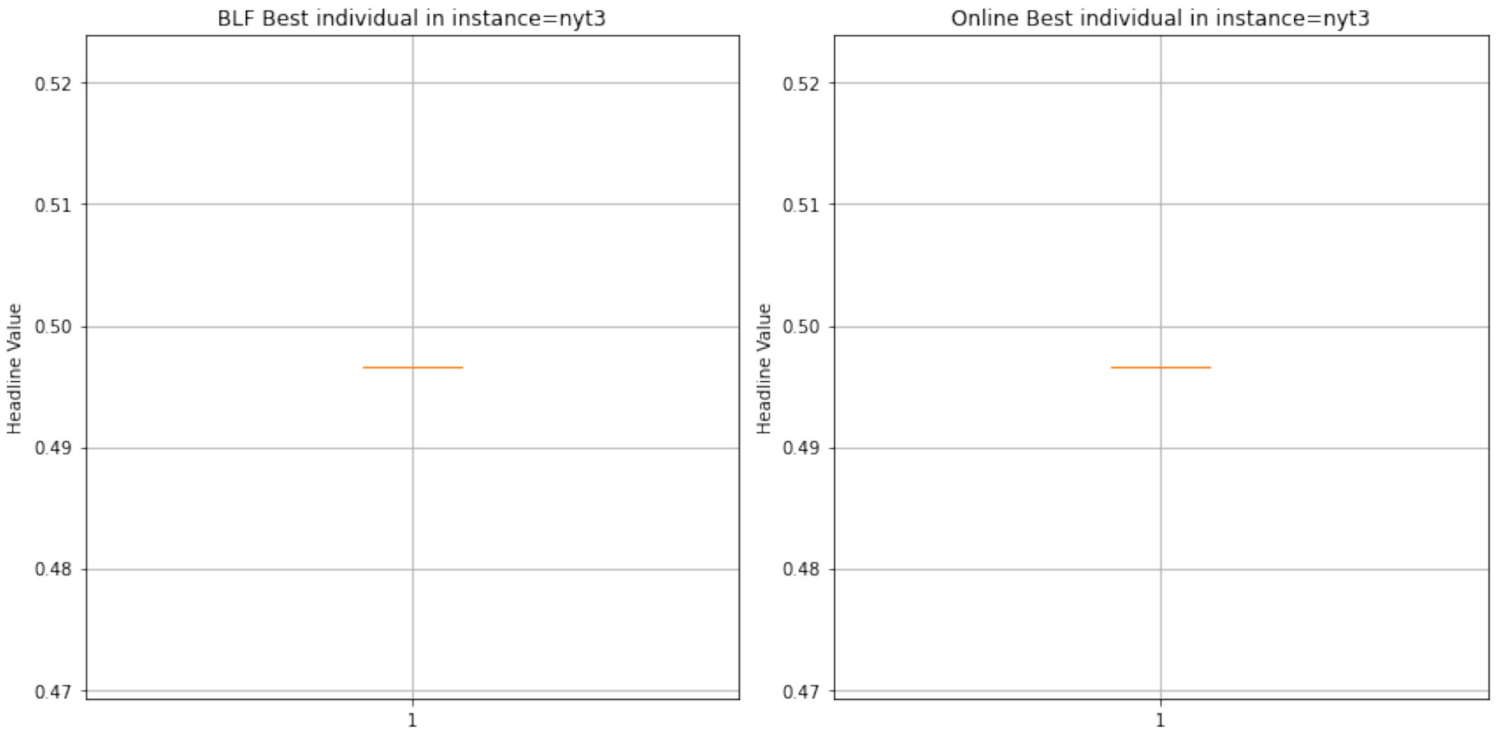


Figure 75: Exp 3: Comparison of Headline Value for instance nyt3

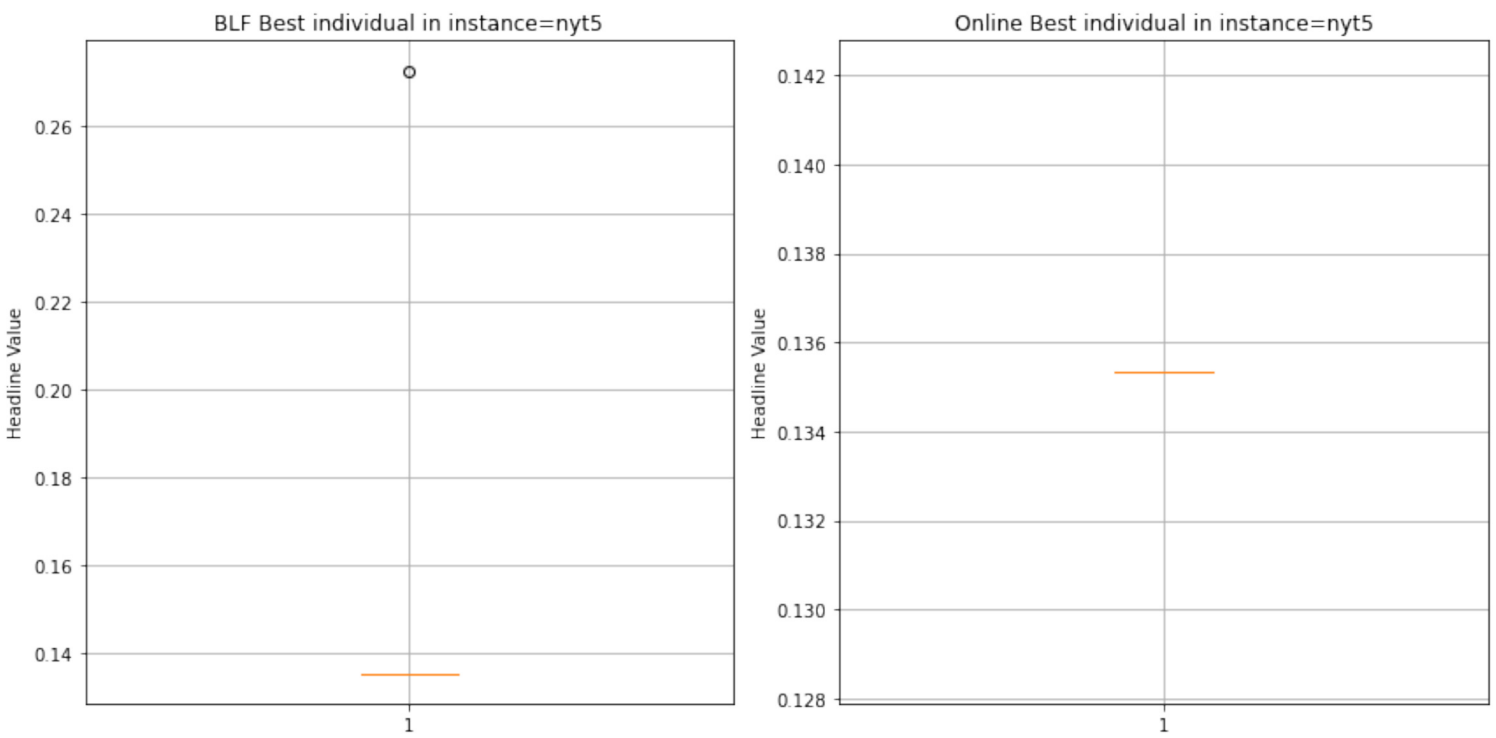


Figure 76: Exp 3: Comparison of Headline Value for instance nyt5

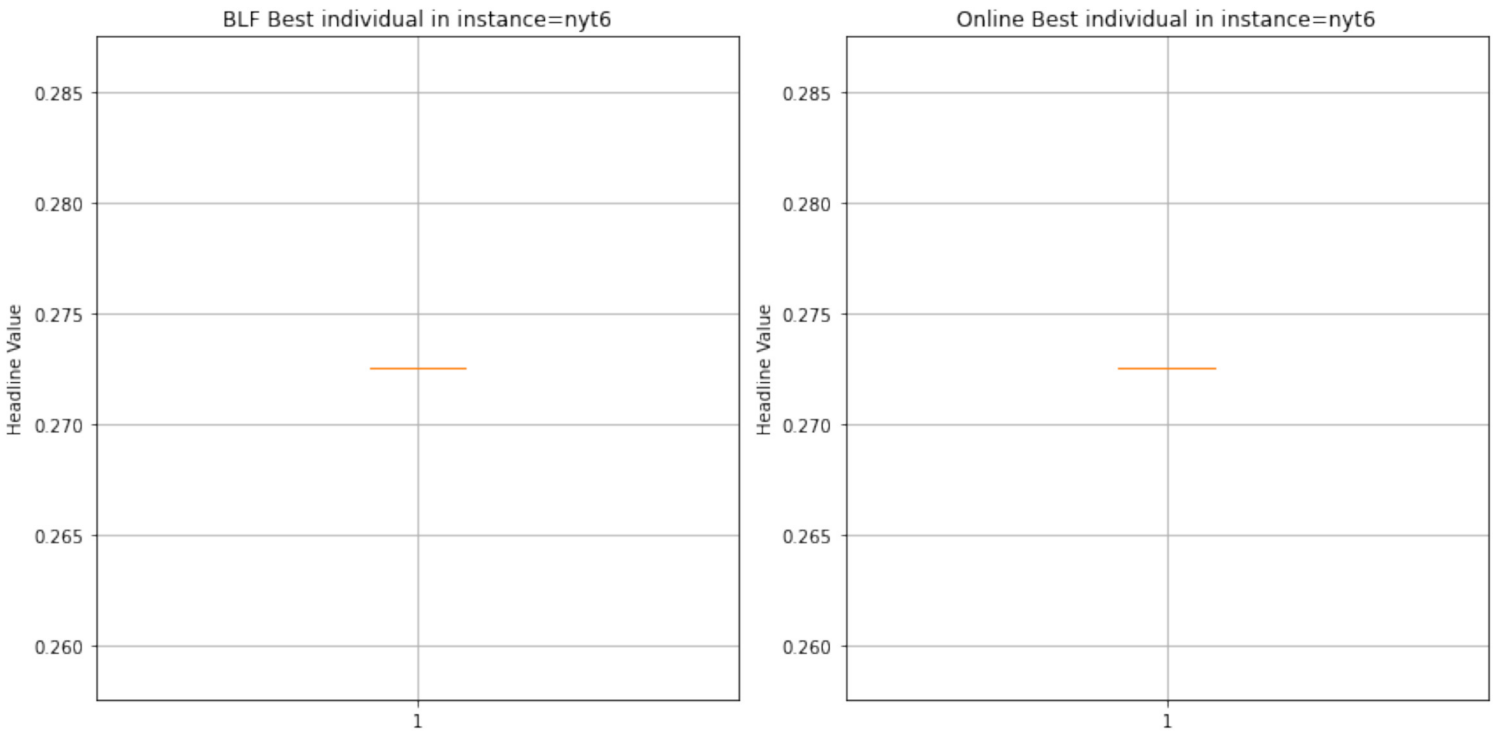


Figure 77: Exp 3: Comparison of Headline Value for instance nyt6

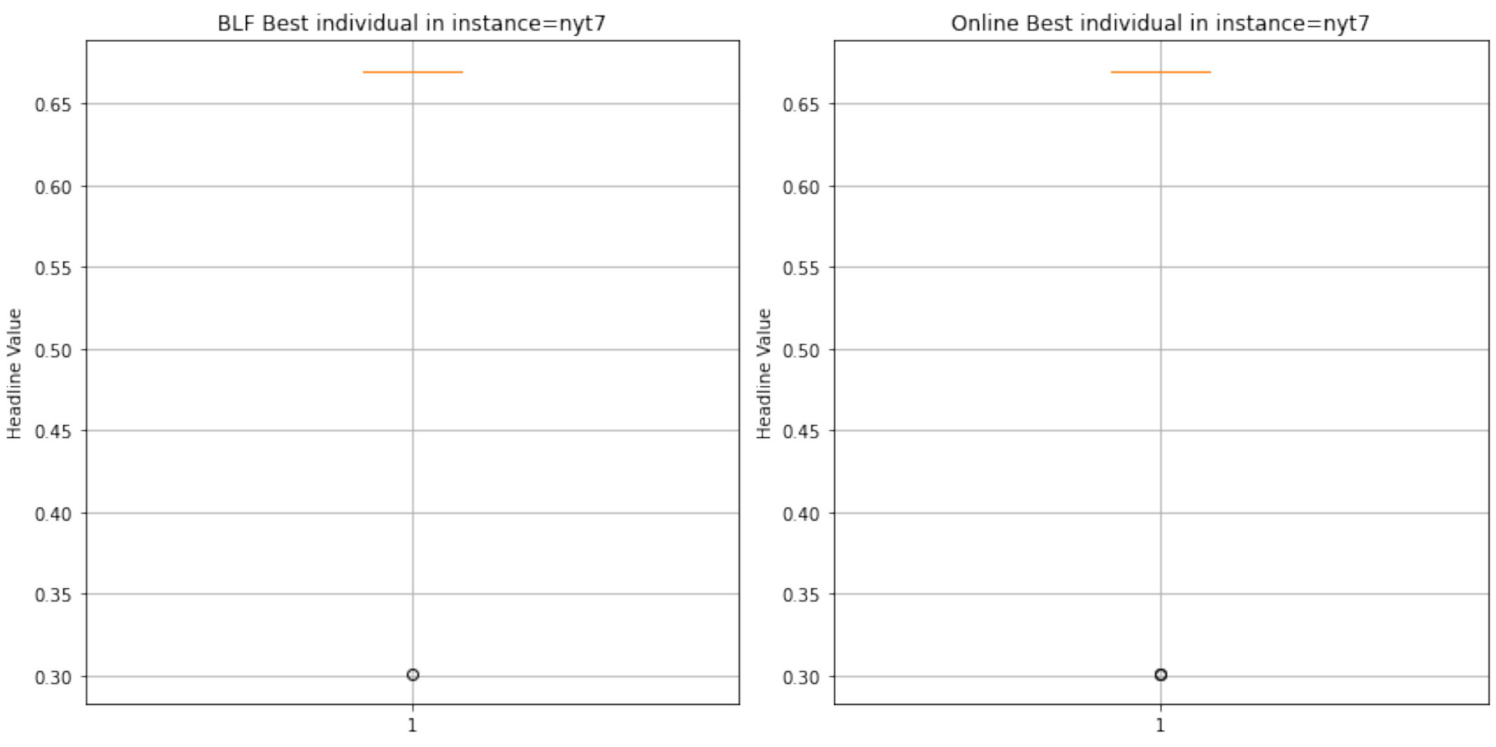


Figure 78: Exp 3: Comparison of Headline Value for instance nyt7

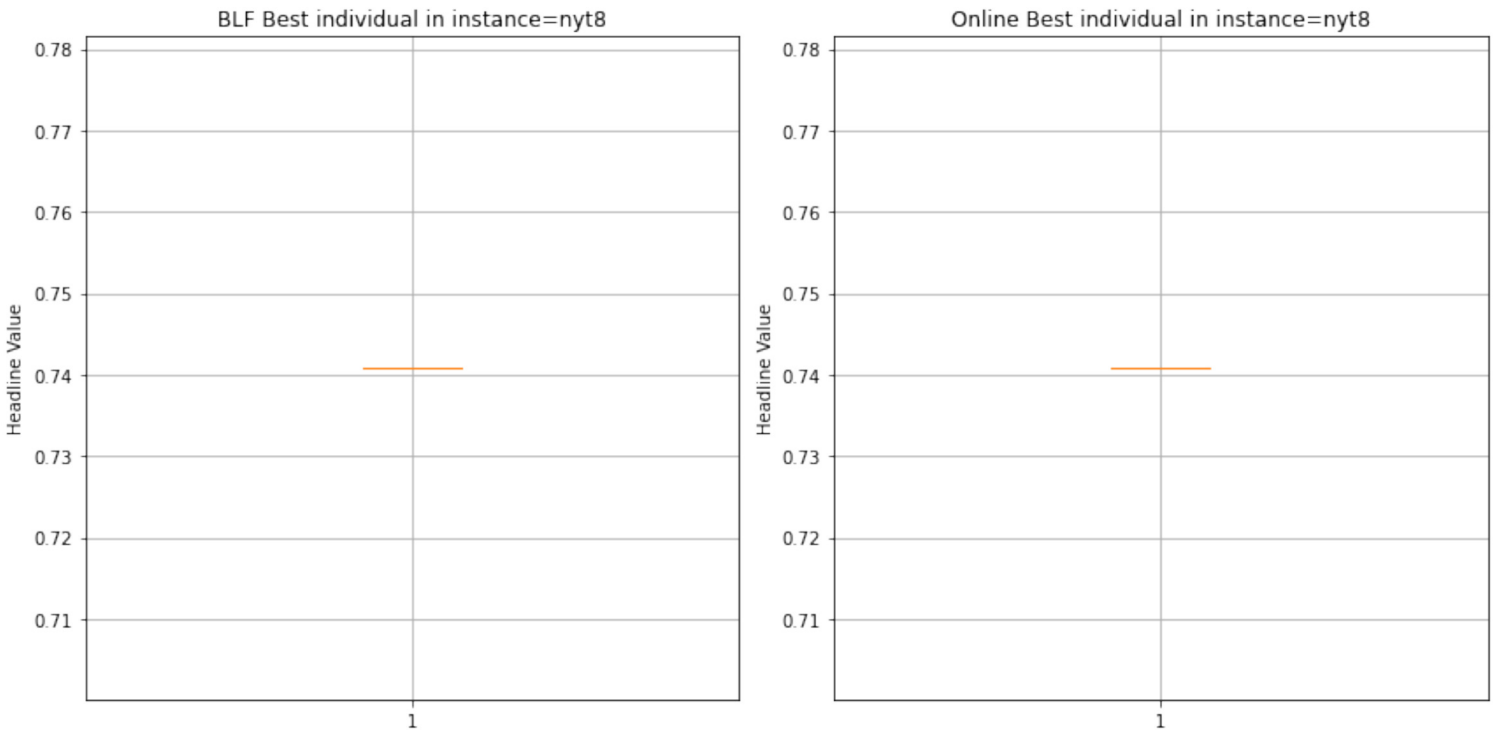


Figure 79: Exp 3: Comparison of Headline Value for instance nyt8

there is one extra mutation associated to heuristics. In any case, online version is preferred because it has greater space to explore, reaching slightly greater aesthetic values in some cases (For example, Fig. 84).

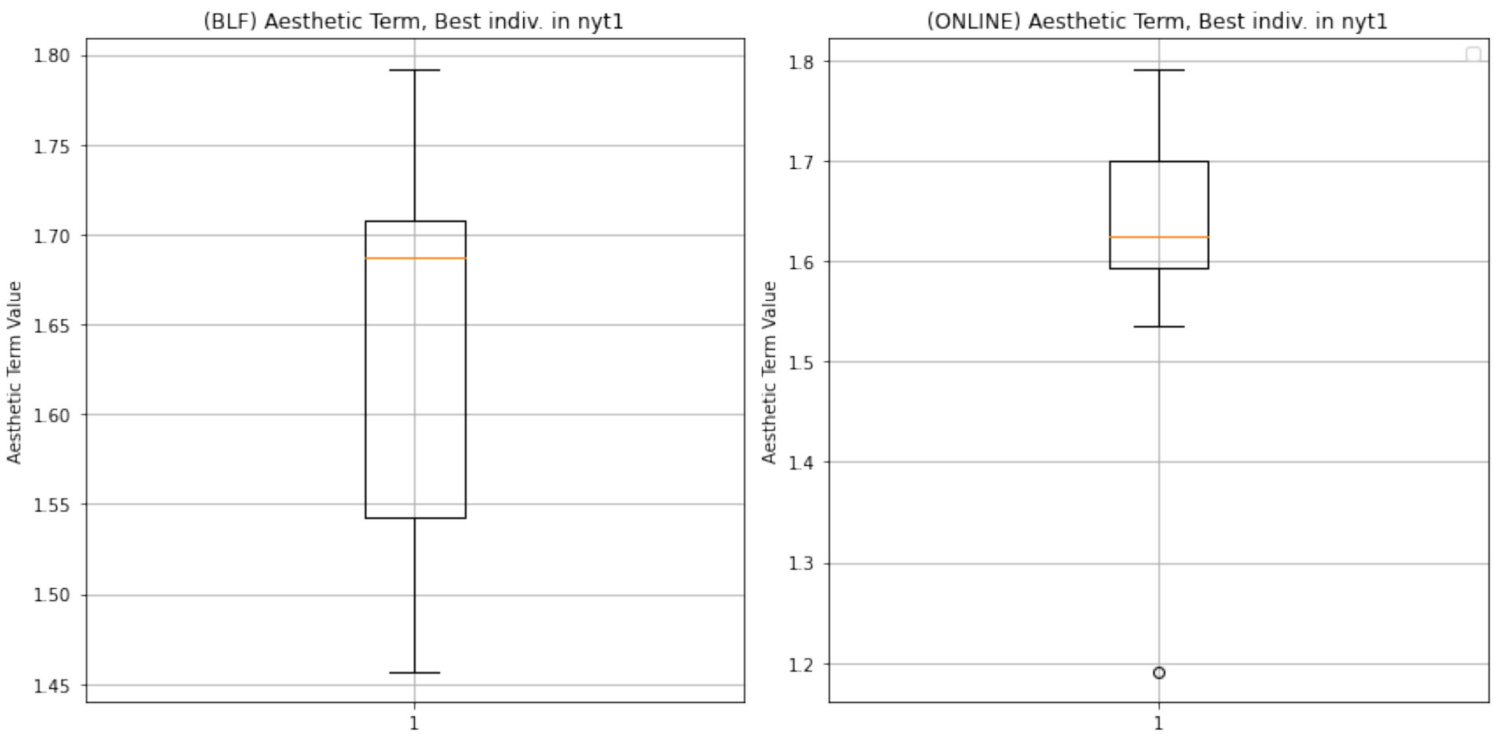


Figure 80: Exp 3: Comparison of Aesthetic Value for instance nyt1

4.8 Chapter Conclusions

Given the design of the algorithm and geometrical constraints of packing articles inside a closed container - page in this case - it is expected that small changes in shapes of any article

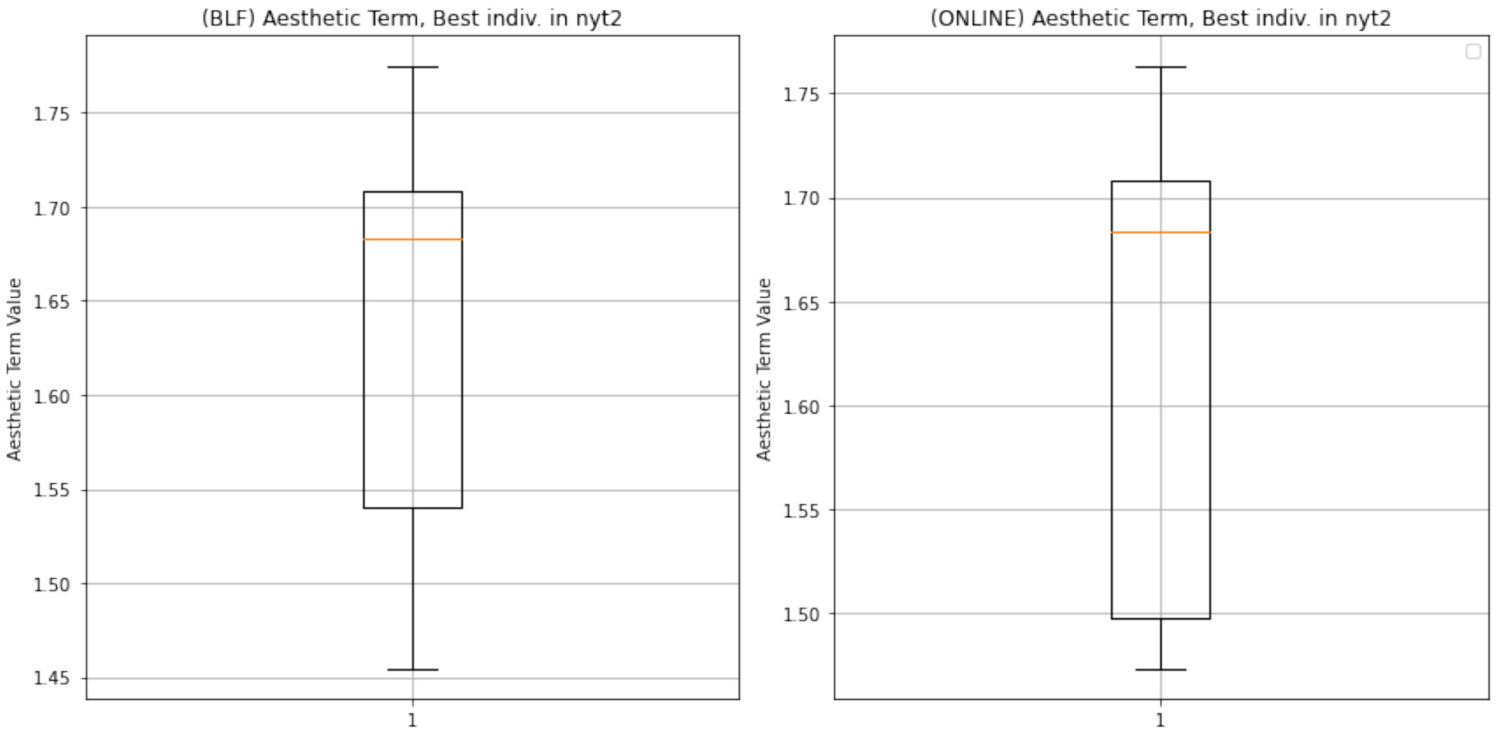


Figure 81: Exp 3: Comparison of Aesthetic Value for instance nyt2

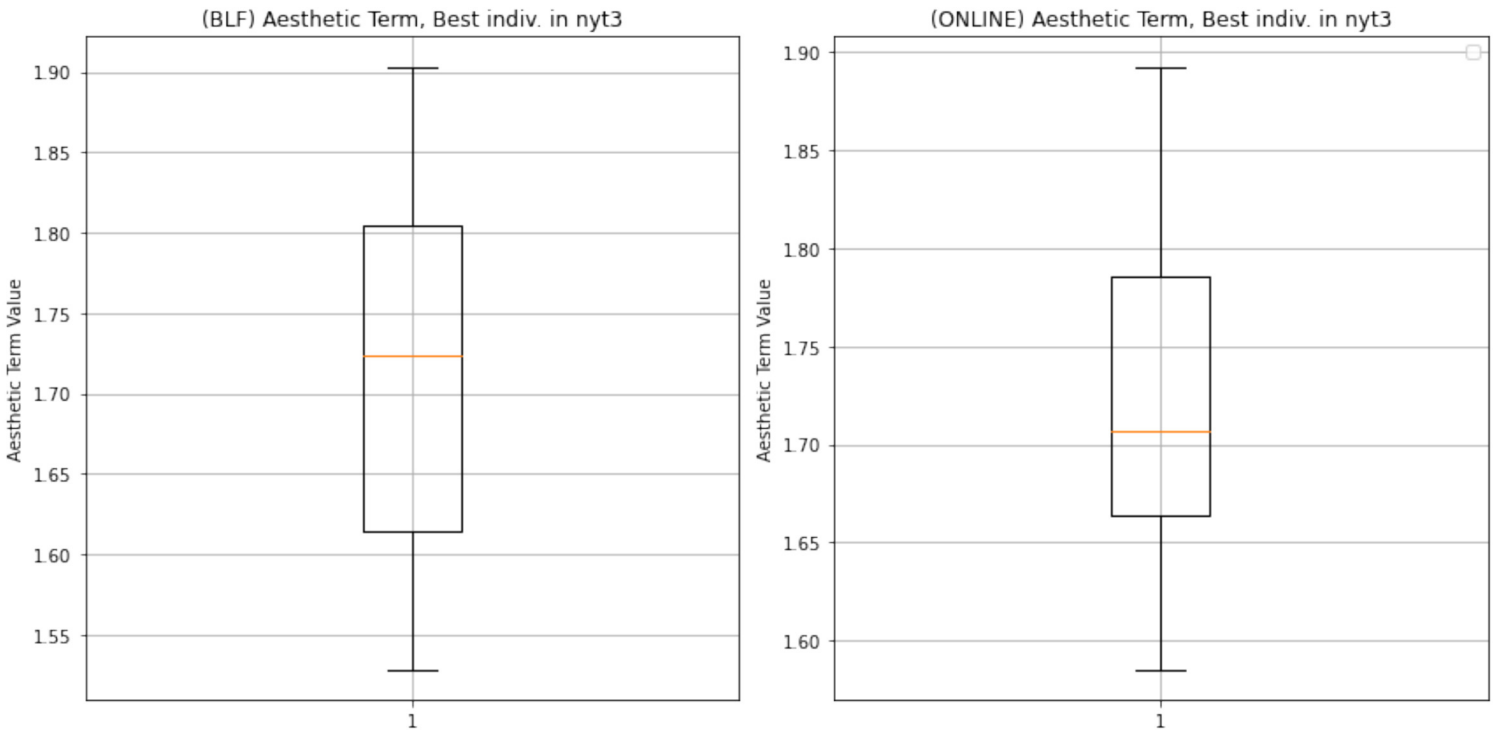


Figure 82: Exp 3: Comparison of Aesthetic Value for instance nyt3

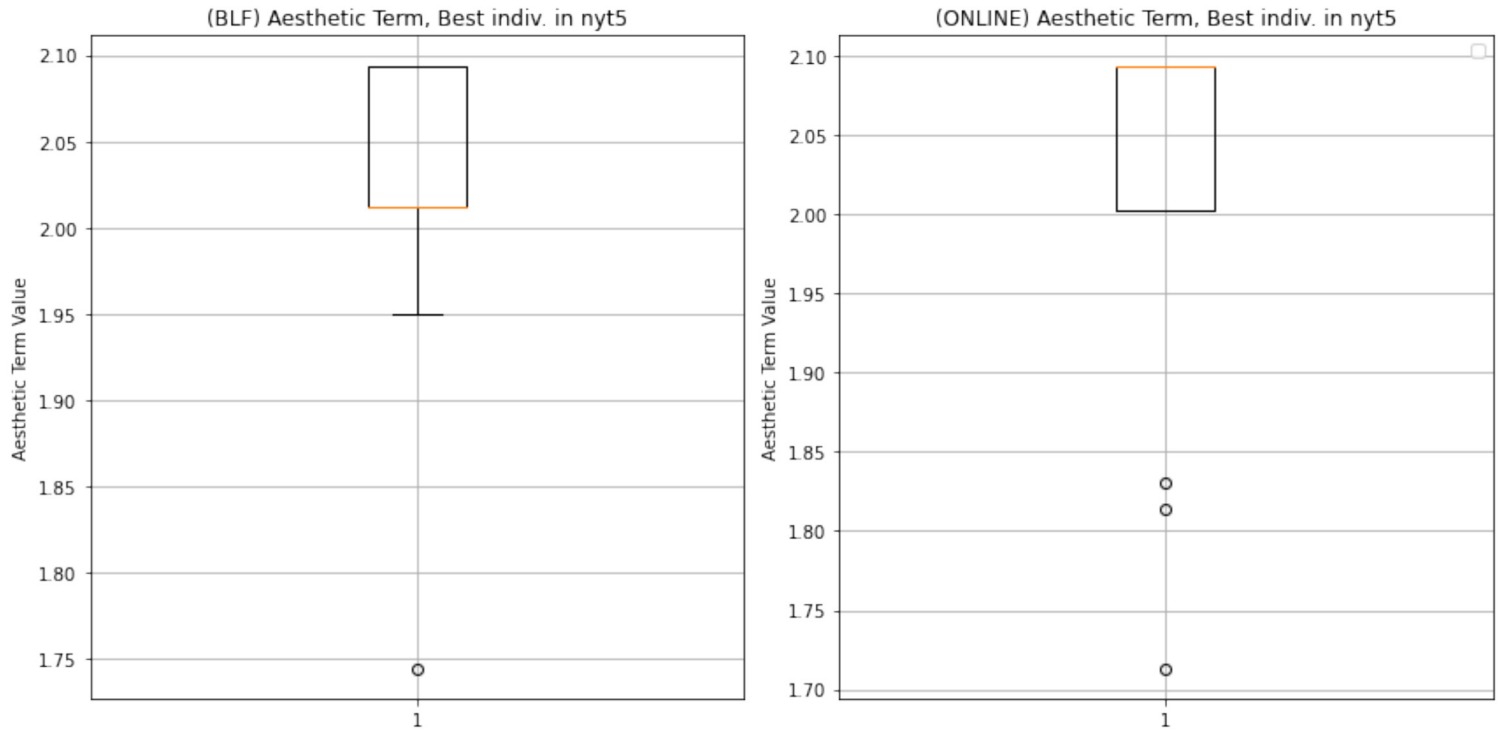


Figure 83: Exp 3: Comparison of Aesthetic Value for instance nyt5

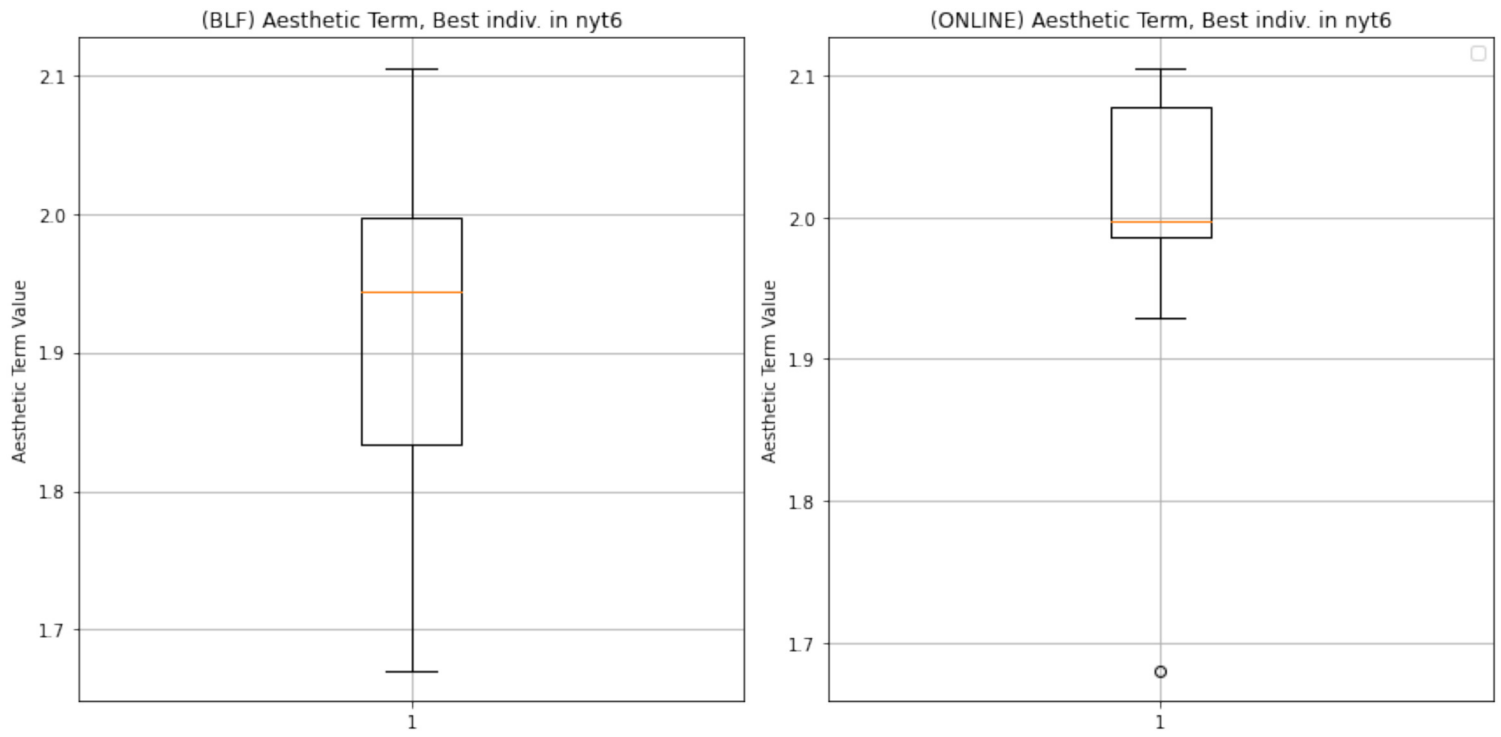


Figure 84: Exp 3: Comparison of Aesthetic Value for instance nyt6

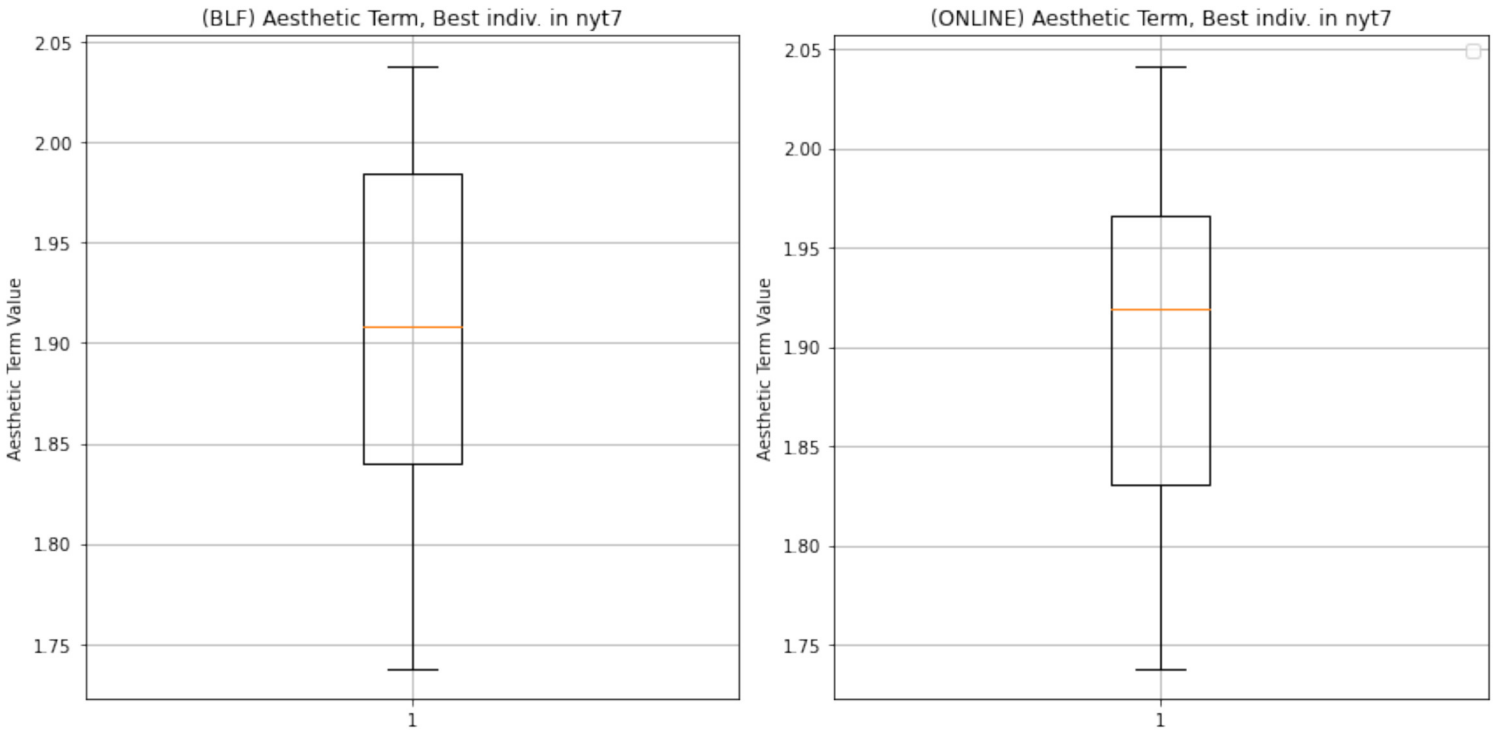


Figure 85: Exp 3: Comparison of Aesthetic Value for instance nyt7

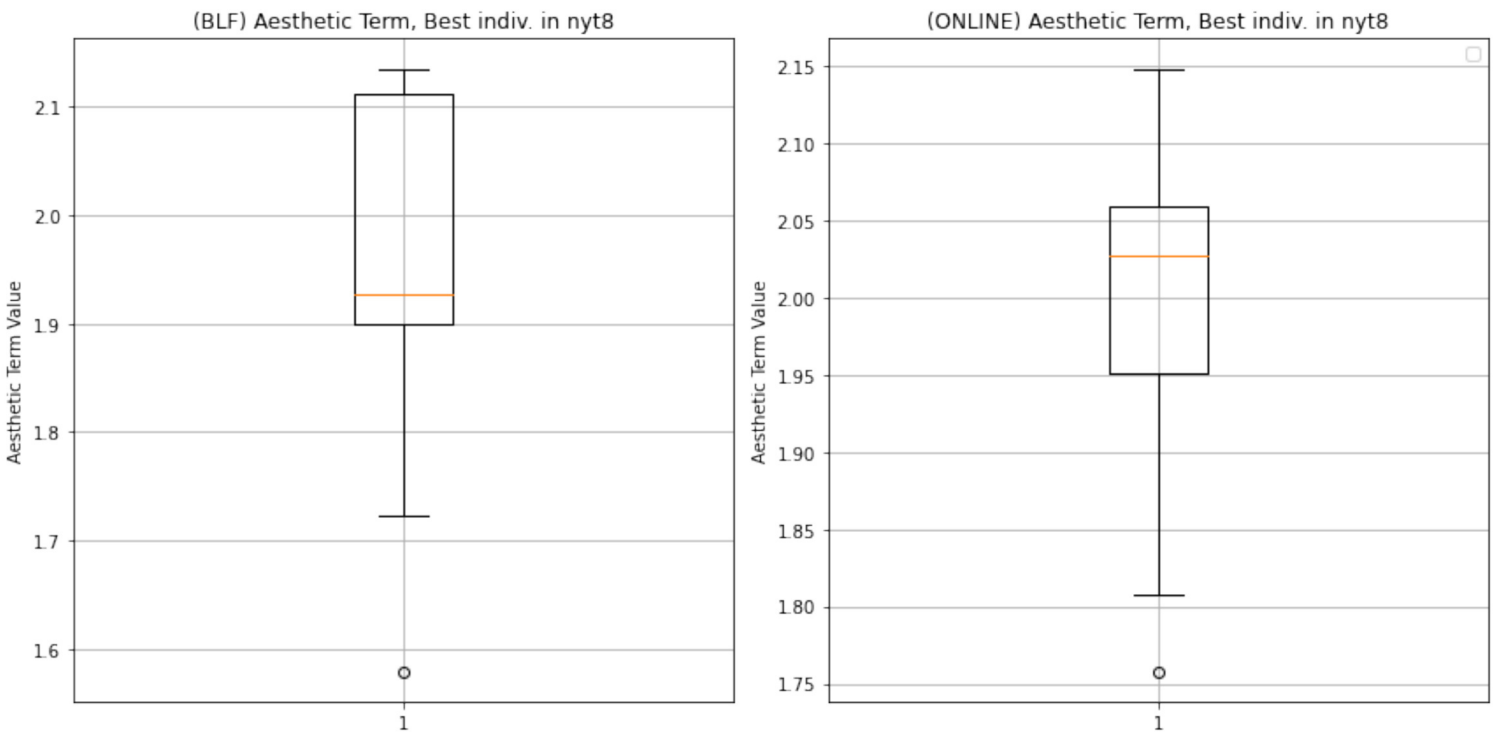


Figure 86: Exp 3: Comparison of Aesthetic Value for instance nyt8

might not lead to a feasible solution. that's why we design the genetic operators to exploit set of shapes found at the initial generation of solutions, so it is expected to converge to the same set of shapes in most of the cases. This leads to convergence in $H(.)$ in almost all cases. However, there are many differences between generated layouts in terms of placement inside the page, associated to the permutation list and quantified by the aesthetic measures and $F(.)$; in this case, as expected, using a roulette wheel with $F(.)$ gives better results.

Also, related to visual results, the genetic method is capable to reduce the excess of lines globally. Something common in some of the instances is that the original input, magnified by some factor, is better than other possible layouts, resulting in this layout as output of the algorithm; this is not a bad behavior, considering that in many other cases it effectively generate new set of shapes, and in others improves aesthetic measures of the input even if it keeps the original set of shapes. Also, it is interesting that the decision tree implemented in the selection procedure works as a polished way to measure quality of the shapes and the excess of lines in each article, as shown in *nyt5* instance where it prefers a worse layout in terms of $H(.)$ but more *homogeneous* in excess of lines.

Finally, second and third experiments show no differences between heuristics and online version of the method in terms of quality $H(.)$. Also, $F(.)$ quality show a small improvement on the online version. On the other hand, there are big differences in execution time between heuristics, with BLF having best performance, thanks to the possibility to fill *holes*. BF also can fill holes, but it needs to search for the best space, so it takes a little longer. iBL cannot fill holes so it needs considerably more time to execute the algorithm.

CHAPTER 5
CONCLUSIONS

We have highlighted that entry points are essential features for readers to scan pages, and thus, one main flaw of simply transposing print into digital media is that the reading experience cannot be as enjoyable and comfortable just because entry points do not play their role correctly anymore. A typical and easy-to-understand example of entry points that are impacted by magnification are headlines which have been our focus here. Interestingly, preserving headline quality (readability and usability) led us to a novel and nontrivial layout generation and packing problem where aesthetics were also taken into account.

There are some aspects of layouting, aesthetics and packing methods in the literature that inspired this work and gives some guidelines for the genetic design of the method.

Results are promising. A genetic method is capable to generate new layouts that optimizes headline accessibility and aesthetics. Some aspects of the design were tested, leading to similar results in terms of accessibility quality of generated layouts and some differences in aesthetic criteria. This is a nice aspect of the designed method, because it is capable to work as a framework that allows for different heuristics to pack articles, leading to similar results.

Of course, this is still a proof of concept since we considered simplifying assumptions here (like text-only articles), but main results lie in this entry point preservation principle. This work opens an entirely new avenue of research to tackle the complexity of newspapers, and make them accessible online to everyone. The designed method is capable of the desired task, with modular aspects that allows for improvements in the future.

5.1 Future Work

Additional criteria could be considered as well, but at the risk of diluting the effect of each term and with the difficulty to weight them with respect to each other.

Something interesting is to consider other newspapers examples of the same brand, and trying to generate layouts according the implicit guidelines imposed by these examples. This was considered in the method presented in this thesis, but was discarded because moves solutions in an opposing direction than $H(.)$, that is the main goal. Future work might add this aspect to the algorithm in a proper way. Analogously, one can choose a set of newspaper for another newspaper, and try to generate layouts that are different. The difficulty here is to measure how much a layout is different than other, in a global way.

Finally, another important aspect is to consider images and ads inside layouts; images should be considered in a different way, because it doesn't have text so it cannot be magnified in the same sense as text. Also, ads should have placement constraints, because different positions inside the page have different impact for the reader.

BIBLIOGRAPHY

[A. *et al.*, 2002] A., L., S., M., and D., V. (2002). Recent advances on two-dimensional bin packing problems. *Discrete Applied Mathematics*, 123:379–396.

[Alvarez-Valdes *et al.*, 2008] Alvarez-Valdes, R., Parreño, F., and Tamarit, J. (2008). Reactive grasp for the strip packing problem. *Computers Operations Research*, 35(4):1065–1083.

[B. and Özcan E., 2009] B., A. and Özcan E. (2009). Bidirectional best-fit heuristic for orthogonal rectangular strip packing. *Annals of Operations Research*, 172(4).

[B., 1983] B., C. (1983). The bottom-left bin-packing heuristic: An efficient implementation. *Computers, IEEE Transactions*, 100(8).

[Baldi *et al.*, 2012] Baldi, M. M., Perboli, G., and Tadei, R. (2012). The three-dimensional knapsack problem with balancing constraints. *Applied Mathematics and Computation*, 218(19):9802 – 9818.

[Berkey and Wang, 1987] Berkey, J. O. and Wang, P. Y. (1987). Two-dimensional finite bin-packing algorithms. *Journal of the operational research society*, 38(5):423–429.

[Brabrand, 2008] Brabrand, G. B. (2008). Dynamic Layout Optimization for Newspaper Web Sites using a Controlled Annealed Genetic Algorithm.

[B.S. *et al.*, 1980] B.S., B., E.G, C., and R.L., R. (1980). Orthogonal packings in two dimensions. *SIAM Journal of Computing*, 9:846–855.

[Buchwald and Scheithauer, 2016] Buchwald, T. and Scheithauer, G. (2016). Upper bounds for heuristic approaches to the strip packing problem. pp. 65–70. Springer.

[Burke *et al.*, 2010] Burke, E. K., Hyde, M., Kendall, G., and Woodward, J. (2010). A genetic programming hyper-heuristic approach for evolving 2-d strip packing heuristics. *IEEE Transactions on Evolutionary Computation*, 14(6):942–958.

[Burke *et al.*, 2011] Burke, E. K., Hyde, M. R., and Kendall, G. (2011). A squeaky wheel optimisation methodology for two-dimensional strip packing. *Computers Operations Research*, 38(7):1035–1044.

[Chen *et al.*, 2015] Chen, B., Wang, Y., and Yang, S. (2015). A hybrid demon algorithm for the two-dimensional orthogonal strip packing problem. *Mathematical Problems in Engineering*, 2015:1–14.

[D. and H., 1999] D., L. and H., T. (1999). An improved bl-algorithm for genetic algorithm of the orthogonal packing of rectangles. *European Journal of Operational Research*, 112(2).

[D. *et al.*, 2006] D., Z., kang Y., and A., D. (2006). A new heuristic recursive algorithm for the strip rectangular packing problem. *Computers Operations Research*, 33:2209–2217.

[David Chek Ling Ngo, 2000] David Chek Ling Ngo, Lian Seng Teo, J. G. B. (2000). A mathematical theory of interface aesthetics. *Visual Mathematics*, 10(8).

[E. *et al.*, 1980] E., C. J., D., G., and R., T. (1980). Performance bounds for level oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 9(1).

[E.K. *et al.*, 2004] E.K., B., G., K., and G., W. (2004). A new placement heuristic for the orthogonal stock-cutting problem. *Operations Research*, 52(4).

[Errea and Gestalten, 2018] Errea, J. and Gestalten, editores (2018). *Newspaper Design: Editorial Design from the World's Best Newsrooms*. gestalten.

[Fortunati *et al.*, 2015] Fortunati, L., Taipale, S., and Farinosi, M. (2015). Print and online newspapers as material artefacts. *Journalism*, 16(6):830–846.

[Garcia and Stark, 1991] Garcia, M. R. and Stark, P. (1991). *Eyes on the News*. Poynter Inst Media Studies.

[Gautier and Gautier, 2017] Gautier, D. and Gautier, C. (2017). *Design, Typography etc.: A Handbook*. niggli.

[Geigel and Loui, 2000] Geigel, J. and Loui, A. C. P. (2000). <title>Automatic page layout using genetic algorithms for electronic albuming</title>. *Internet Imaging II*, 4311(August 2002):79–90.

[Hadjiconstantinou and Iori, 2007] Hadjiconstantinou, E. and Iori, M. (2007). A hybrid genetic algorithm for the two-dimensional single large object placement problem. *European Journal of Operational Research*, 183(3):1150–1166.

[Harrington *et al.*, 2004] Harrington, S. J., Naveda, J. F., Jones, R. P., Roetling, P., and Thakkar, N. (2004). Aesthetic measures for automated document layout. En *Proceedings of the 2004 ACM Symposium on Document Engineering*, DocEng '04, p. 109–111, New York, NY, USA. Association for Computing Machinery.

[Hays, 2018] Hays, S. (2018). An Analysis of Design Components of Award-winning Newspaper Pages. 9(2):44–63.

[Hollander *et al.*, 2011] Hollander, B. A., Krugman, D. M., Reichert, T., and Avant, J. A. (2011). The e-reader as replacement for the print newspaper. *Pub Res Q*, 27:126–134.

[Ihlström *et al.*, 2004] Ihlström, C., Åkesson, M., and Nordqvist, S. (2004). From print to web to e-paper-the challenge of designing the e-newspaper. *Proceedings of ICCC 8th International Conference on electronic Publishing*, p. 249–260.

[Imahori and Yagiura, 2010] Imahori, S. and Yagiura, M. (2010). The best-fit heuristic for the rectangular strip packing problem: An efficient implementation and the worst-case approximation ratio. *Computers Operations Research*, 37:325–333.

[Jacobs *et al.*, 2003] Jacobs, C., Li, W., Schrier, E., Bargeron, D., and Salesin, D. (2003). Adaptive grid-based document layout. *ACM Transactions on Graphics*, pp. 838–847. Computer graphics.

[Jakobs, 1996] Jakobs, S. (1996). On genetic algorithms for the packing of polygons. *European Journal of Operational Research*, 88(1):165–181.

[Jylänki, 2010] Jylänki, J. (2010). A thousand ways to pack the bin-a practical approach to two-dimensional rectangle bin packing. *retrived from http://clb. demon. fi/files/RectangleBinPack. pdf.*

[Jylänki., 2019] Jylänki., J. (2019). A thousand ways to pack the bin-a practical approach to two-dimensional rectangle bin packing. `https://github.com/solomon-b/greedypacker`.

[Leung and Zhang, 2011] Leung, S. C. and Zhang, D. (2011). A fast layer-based heuristic for non-guillotine strip packing. *Expert Systems with Applications*, 38(10):13032–13042.

[Lodi *et al.*, 2002] Lodi, A., Martello, S., and Monaci, M. (2002). Two-dimensional packing problems: A survey. *European Journal of Operational Research*, 141(2):241 – 252.

[Lok and Feiner, 2001] Lok, S. and Feiner, S. (2001). A survey of automated layout techniques for information presentations. *Proc. Smart Graphics 2001 (First Int. Symp. on Smart Graphics)*, pp. 61–68.

[Morell, 2016] Morell, R. (2016). Design as a driving force for audience engagement. Nieman Reports.

[Ntene and van Vuuren, 2009] Ntene, N. and van Vuuren, J. (2009). A survey and comparison of guillotine heuristics for the 2d oriented offline strip packing problem. *Discrete Optimization*, 6(2):174–188.

[Oliveira *et al.*, 2016a] Oliveira, J., Neuenfeldt Júnior, A., Silva, E., and Carravilla, M. (2016a). A surveyonheuristics for the two-dimensional rectangular strip packing problem. *Pesquisa Operacional*, 36:197–226.

[Oliveira *et al.*, 2016b] Oliveira, J. F., Júnior, A. N., Silva, E., and Carravilla, M. A. (2016b). A survey on heuristics for the two-dimensional rectangular strip packing problem. *Pesquisa Operacional*, 36(2):191–226.

[Purvis, 2002] Purvis, L. (2002). A genetic algorithm approach to automated custom document assembly. pp. 131–136.

[Riff *et al.*, 2009] Riff, M., Bonnaire, X., and Neveu, B. (2009). A revision of recent approaches for two-dimensional strip-packing problems. *Engineering Applications of Artificial Intelligence*, pp. 833–837.

[Strecker and Albayrak, 2010] Strecker, T. and Albayrak, S. (2010). Adaptive user-centric layout of news with aesthetic constraints. *Information Management SPIM 2010*, p. 8.

[Thomas and Chaudhari, 2014] Thomas, J. and Chaudhari, N. (2014). A new metaheuristic genetic-based placement algorithm for 2d strip packing. *Journal of Industrial Engineering International*, 10.

[Wauters *et al.*, 2013] Wauters, T., Verstichel, J., and Vanden Berghe, G. (2013). An effective shaking procedure for 2d and 3d strip packing problems. *Computers Operations Research*, 40:2662–2669.

[Wäscher *et al.*, 2007] Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(1):1109–1130.

[Yang *et al.*, 2013] Yang, S., Han, S., and Ye, W. (2013). A simple randomized algorithm for two-dimensional strip packing. *Computers Operations Research*, 40.

[Zhang *et al.*, 2016] Zhang, D., Shi, L., Leung, S. C., and Wu, T. (2016). A priority heuristic for the guillotine rectangular packing problem. *Information Processing Letters*, 116(1):15–21.