**Chapter 3: the Main Contribution**

**3.1 Introduction:**

We notice that the evaluation of personalized user interface is often neglected. So, it is necessary to envisage new specific evaluation methods devoted to adaptive user interface evaluation. Indeed, adaptive user interface evaluation studies should benefit from the existing methods in every development phases to reach more rigorous level in terms of statistical analysis, correctness in procedures, experiment settings, etc.

In this chapter we are going to present the mains contributions of our studies that are established to generate an automatic evaluation method for adaptive user interface. In addition, we will describe in the next sections the proposed metrics for adaptive user interface and the overview of the evolutionary algorithm that is used to extract the evaluation rules.

**3.2 Proposed Metrics for AI evaluation:**

The ultimate goal of this research is to develop an effective metric-based tool to evaluate the adaptive user interface.

We aim to generate optimized rules to evaluate the quality of adaptive interface. In this context, we propose a set of metrics which can be adopted to evaluate the user adaptive interface: density, grouping, homogeneity, regularity, sequence, simplicity, symmetry, and unity. In fact, these proposed metrics denoted from existing studies (Ngo et al, 2002) and based on several ergonomic criteria proposed by (Bastien et al, 1993). So according to the definition of such criteria, we will group these metrics into two criteria:

* Guidance: grouping, regularity, sequence and simplicity.
* Coherence: unity, symmetry, density and homogeneity.

These metrics measure produce values between 0 and 1. Where, 0 present the lower value, 1 present the higher value and for the medium we use the average of this interval (0.5) that as the threshold.

* + 1. **Guidance:**

***Guidance*:** user guidance refers to the means available to advice, orient, inform, interact, and guide the users throughout their interaction with computer (message, alarm, label, etc.). This criterion is subdivided into two metrics: Regularity [Ngo et al, 2002] and grouping [Miyoshi et al, 2001].

* + - 1. **Regularity:**

Regularity refers to the consistency of the adaptive interface. This metric can be accomplished by providing a uniformity space between adaptive interface objects, and by minimizing the alignment point which is the number of row and columns of the interface. It aims to organize the structure of these components. The regularity of the adaptive interface is related to many user profile criteria such as age, motivation, education level, etc. In fact, the regularity of the interface provides guidance for novice users having a low education level. This well arrangement of components helps also these users to read in an easier way the content of adaptive interface (AI).So, adaptive interfaces must provide a high regularity level for this kind of users. In order, to measure this metric we will base on the formula proposed by [Ngo et al, 2002]:

RMalignment: is the extent to which the alignment points are minimized.

RMspacing: is the number of distinct distances between column and row starting points.

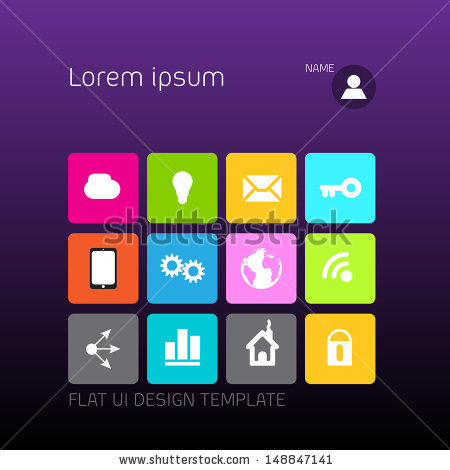
nv = the numbers of vertical alignment points ( number of row).

nh = the numbers of horizontal alignment points( number of column).

nspacing= the number of distinct distances between column and row starting points.

n: is the number of the component of the adaptive interface.

HigherRMalignment values are refers to how well the components are aligned in the adaptive interface. Higher RMspacing valuemeans that the space between components is normalized.



1. (b)

Fig.1: Two versions of adaptive interface having different regularity. (a) Low regularity. (b) High regularity.

* + - 1. **Grouping / composition**

Grouping is a guidance technique which aims to orient and guide user throughout their interaction with adaptive interface. In fact, the component of this adaptive interface that is semantically linked should be seems visually in some way. It calculates the number of objects that own a clear boundary by line, color, etc. This metric enhance the visual clarity which is achieved when the interactive objects are organized and presented in meaningful and understandable manner. It is related to experience and age of user.For example, grouping makes adaptive interface easier to understand by users having lower experience.

The formula of this metric is given by [Miyoshi et al, 2001]:

gi: is the number of groups with clear boundary by line, background, color, or space

: is the total number of groups.



1. (b)

Fig.2: Two versions of adaptive interface in different grouping. (a) The bad grouping. (b) The good grouping.

* + - 1. **Sequence / storing:**

Sequence is the metric that describe the way of how organize the objects in the adaptive system in relation to the eye movement that progress sequentially from dark area to light area, from big object to little object, etc. So it is providing to spruce up the component to lead the eye of user through the adaptive interface in a logical and sequential ordering that refer to the user’s needs. This metric is related to the age, education level and interests of user. For example, Sequence may facilitate the use of adaptive interfaces to user who have old age and lower level of education.

We can calculate it as follow [Ngo et al, 2002]:

With:

UL: upper-left

UR: upper-right

LL: lower-left

LR: lower-right

: is the total weight of quadrant j

: is the area of object i on quadrant j.

Each quadrant is given a weighting in q.So.



Fig.3: good adaptive interface with a high sequence.

* + - 1. **Simplicity:**

Simplicity refers to the optimization of the number of interactive object in the adaptive interface and the minimization of the alignment point. Simplicity helps user to appreciate easily the meaning of expected information. Moreover, only the information related to user’s tasks or needs is to be presented in the adaptive interface.

This metric is related not only to the age, motivation and education level of user but also to the type of platform used by them. For example, simplicity helps old people by giving them a simple interface’s layout, at variance to motivated person that having a higher education level that prefers an interface with low simplicity.

We can calculate it as follows [Ngo et al, 2002]:

nvap= number of vertical alignment points.

nhap= number of horizontal alignment points.

= number of object on the frame.

SMM is conversely proportional to the addition of the numbers of alignment points and the number of interactive objects of the interface so when these summations are decreased, SMM tend to increase. A high simplicity is achieved with the smaller number of component which they are aligned in the adaptive interface.



Fig.4: the same interface in different level of simplicity.

* + 1. **Coherence**

Coherence is the means that used to provide established interaction between users and interface, and secure the efficient use of the interface. This criterion is subdivided into four metrics: unity, density, size and symmetry [Ngo et al, 2002].

* + - 1. **Unity :**

Unity is grouping all interface elements to appear like a one piece. It concerns the adaptation to the platform capacities by which the users interact with the system ( PC, Smartphone, etc.) this property is related to the term plasticity [Thévenin and Coutaz, 1999], [Calvary et al, 1990]. Authors define this term as the capacity of the interface to adapt the context use by respecting its usability. This criterion is related only on to information support (container). In fact, the interface surface of the digital personal assistant is not the same as of a computer that is why it is necessary to adapt the information quantities and form of information, the navigation in the interface, graphic objects placement according to the visualized support. The measure of this metric is determined by the extent to which the components are related in size, and the relative measure of the space between groups (groups of component) and that of margins [Ngo, 2002]. It helps to secure centered element on the interface and avoid its fragmentation.

Well unity obtain by using the optimum number of size component (minimize the uses of different sizes in adaptive interface) and leaving less space between objects. So when the level of unity increase, the adaptive interface is not centered as well .

This metric is given by (Ngo, 2000)

UMform: is the extent to which the objects are related in size.

UMspace: is a relative measure of the space between groups and that of the margins.

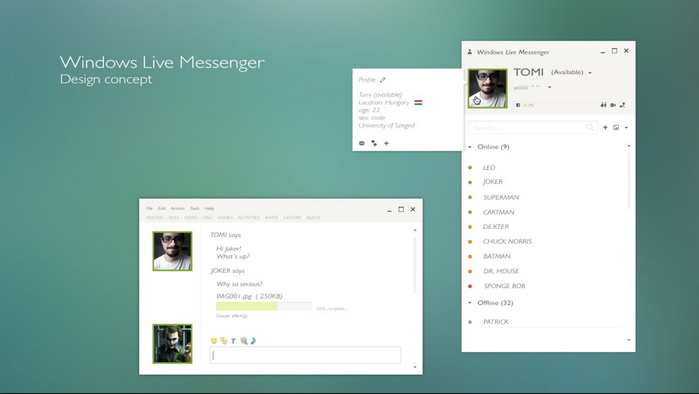
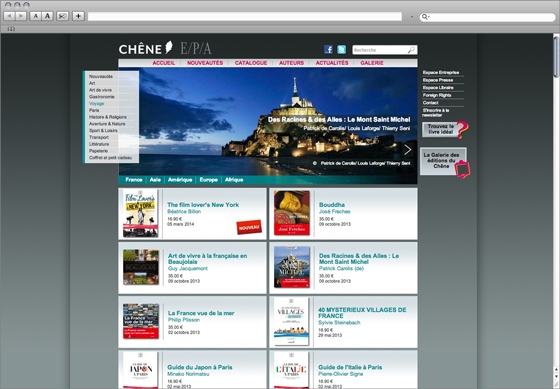
nsize:the number of size used.

n : the number of object.

aAI:the area of the adaptive interface.

aframe: the area of the screen.

ai: the area of the interactive object i.

****

1. (b)

Fig.5: two versions of screens in unity study. (a) Interface with high unity.

(b) Interface with low unity.

* + - 1. **Density:**

Density concerns the workload of adaptive interface. It is a set of information presented to the user. Density is the extent to which the adaptive interface is covered with objects. This metric is related to the motivation, experience and center of interest of users. In fact, user with a low motivation prefers an interface with a low density level. Density can also depend on the type of platform employed.

Density measure (DM) is given by [Ngo et al, 2002]:

ai= area of the interactive object i.

aframe= area of the frame ( screen of interactive platform).

n= number of interactive object.

DM present the percentage of interactiveobject area in the adaptiveinterface that should be equal to 50%, so DM= 1 when density level is 50, and a Higher values are related to how closely the level is 50.



1. (b)

Fig.6: Two versions of adaptive interface with different density levels. (a) Interface with higher density level.(b) Interface with medium density level.

* + - 1. **Symmetry:**

Symmetry is one of metrics who provide the coherence of interface. It gives by an equal distribution of the quantity of elements on the right and the left columns of adaptive interface.

Symmetry consists to duplicate component on the left, right, and radical of the adaptive interface centerline, and avoid the imbalance in the different part of interface.

Adaptive interface should be adapted to the motivation of the users, and it should also stimulate their interests. For example person who has less level of motivation should get an interface with higher coherence.

The Symmetry measure (HM) is given by [Ngo et al, 2002]

SYMvertical, SYMvertical, SYMradial are, respectively the vertical, horizontal and radial symmetries with

X’j, Y’j, H’j, B’j, Θ’j, and R’j are, respectively, the normalized values of

UL, UR, LL, and LR stand for upper-left, upper-right, lower-left, and lower-right, respectively.

Xj: is the total x-distance of quadrant j.

Yj: is the total y-distance.

Hj : is the total height.

Bj: is the total width.

Θj: is the total angle.

Rj: is the total distance

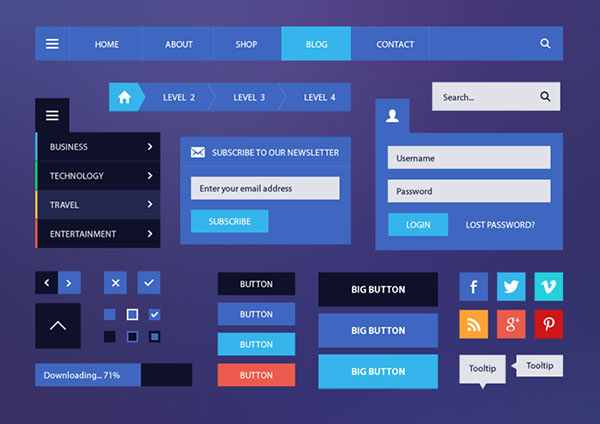
(*xij; yij*) : the coordinates of the centers of object i on quadrant j.

(*xc; yc*) : the coordinates of the frame.

bij: the width of the object.

bij: the height of the object.

nij: the total number of objects on the quadrant.



1. (b)

Fig.7: two versions of screens in symmetry study. (a) Interface with horizontal symmetry.(b) Interface with higher homogeneity.

* + - 1. **Homogeneity :**

Homogeneity is an overall seen of component distribution in adaptive interface, which gives users an equal arrangement of object among the four quadrants. This metric is the comparison between, the numbers of different ways that objects can be organized for the four quadrants, compared to an optimal distribution.

The optimal distribution obtain when the n object are evenly allocated with the quadrants of adaptive interface. However, for n component there are n! different ways to organize them. And in each quadrant (upper-left, upper-right, lower-left, lower-right), nj object can be organized with nj! different ways.

This metric is related to the level of user experience. In fact, adaptive interface should propose an optimal distribution for novice users in order to help them to navigate through it.

The homogeneity measure (HM) is given by [Ngo et al, 2002]

W: is the number of different ways of n object can be arranged for the four quadrants.

Wmax: W is maximum when the n objects are evenly allocated to the four quadrants of the adaptive interface.

n: is the number of object on the adaptive interface.

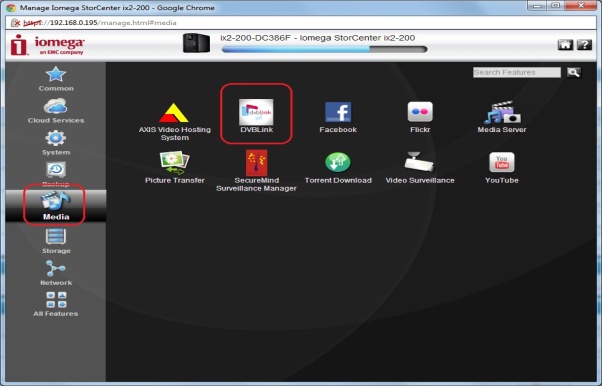
nj: is the total number of objects in quadrant j.

nUL: is the number of object on the upper-left.

nUR: is the number of object on the upper-right.

nLL: is the number of object on the lower-left.

nLR: is the number of object on the lower-right.



1. (b)

Fig.8: two versions of adaptive interface with different level of homogeneity. (a) Interface with low homogeneity. (b) Interface with high homogeneity.

* + 1. **Complexity of this metrics:**

The complexity of this metrics is calculated as follow:

COMG: is the complexity of guidance.

COMC: is the complexity of coherence.

* + 1. **Correlation between metrics and user profile**:

Each metrics is related with set of context, for example, regularity is related with age, motivation and education level. These context and criteria can have the following value: low (-), medium (±) and high (+).

When the values of age and education level are low, adaptive interface should have high regularity level. But when motivation is high regularity is not important and can be with low regularity level.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| User Pref  MIA | age | motivation | User experience | Interest | Education level |
| Regularity | **(+,-)** | **(-,+)** |  |  | **(+,-)** |
| Grouping | **(+,-)** |  | **(+,-)** |  |  |
| Sequence | **(+,-)** |  |  | **(+,+)** | **(+,-)** |
| Simplicity | **(+,-)** | **(-,+)** |  |  | **(+,+)** |
| Unity |  |  | **(+,-)** |  |  |
| Density |  | **(+,+)** | **(+,+)** | **(+,+)** |  |
| Symmetry |  | **(+,-)** |  | **(+,-)** |  |
| Homogeneity |  |  | **(+,-)** |  |  |

Table 1: mapping between adaptive user interface criteria and context.

* 1. **approach overview:**

We propose an approach that uses knowledge from previous empirical study of a several adaptive interface examples to generate a set of evaluation rules based on the combination of the results of this study. Fig9 shows the general structure of the proposed approach.

Fig9: approach overview

Our approach takes as inputs a set of information extracted from a several adaptive user interface used for the evaluation process:

-**The trace** is a base of examples that contains the profiles of the users which we invited to test the various adaptive user interfaces to estimate so that the problems experienced by them during their uses of these interfaces.

-**The context of use** contains the various preferences of users: age, motivation, education level, user experience, interest, and their taken values: low, medium, high.

-**Threshold of metrics** corresponds to the results of the calculation of the proposed metrics for each adaptive user interface generated by the parser.

-**Problems** are the different type of problems of the adaptive interface seen by the users and they are classified according to the values taken by the metrics of these interfaces.

in this context, we fitted the threshold of metric’s values to 0.5 thus every interface has a value of metrics superior to 0.5 is an interface which has problems of high quality, while an interface with a value of the metrics lower than 0.5 is considered as an interface which has problems of low quality.

In our study, we consider two types for problems that correspond to high quality or low quality of adaptive user interface basing on the values of the calculated metrics.

This table below contains the proposed metrics and the correspond problems that can be detected through the values of each metric.

|  |  |
| --- | --- |
| Metrics | Correspond problems |
| Regularity | -irregular interface : low quality  -regular interface : high quality |
| Grouping | -low guidance : low quality  -high guidance : high quality |
| Sequence | -not arranged interface: low quality  -well arranged interface : high quality |
| Simplicity | -Complex interface : low quality  -Simple interface: high quality |
| Unity | - not centered interface: low quality  -Centered interface: high quality |
| Density | - Not charged interface : low quality  -charged interface: high quality |
| Symmetry | -incoherent interface : low quality  -coherent interface : high quality |
| Homogeneity | -unequal arrangement interface : low quality  -equal arrangement interface: high |

Table 2: metrics and correspond problems

As outputs, our approach derives a set of evaluation rules from the genetic algorithm. Thus, the generation process can be viewed as a search-based that combines the inputs of this approach to generate these rules. So the best solution of our study is a set of rules that detect the maximum of problems and contains the minimum of rules.

In addition, the quality of this solution is calculated by the fitness function that compare the different generated rules with the examples from the trace.

As many inputs combinations are possible, we have also a huge number of possible solutions derived from these combinations, so we need to use a meta-heuristic search to explore this large number of these combinations.

***-The parser:***

This parser is a set of classes that used to parse different type of code source of adaptive interface to extract the dimension of each component and it apply the formula of each metrics to generate the threshold of metrics that are the input of the evolutionary algorithm used to generate evaluation rules.

We describe the conception of this parser in a uses cases diagram:

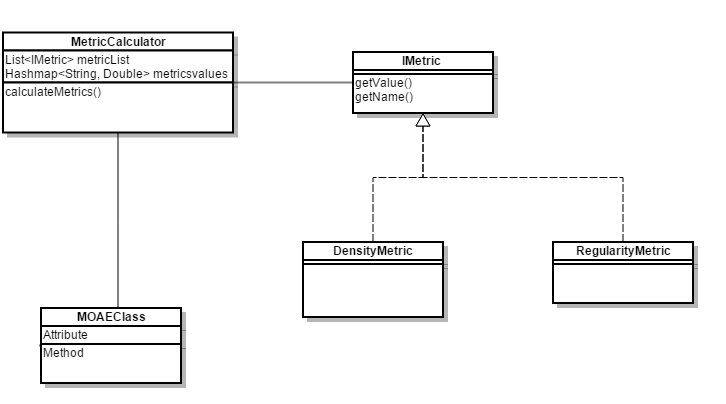


Fig10. Diagram of uses cases of the parser

Fig 11 . Global representation of the approach

* 1. **Evaluation rule extraction using a evolutionary algorithm:**

This section describes the evolutionary algorithm which we used as well as the adaptations brought to its general functioning.

* + 1. **Evolutionary algorithm overview:**

An evolutionary algorithm is a stochastic optimization method based on the Darwirian theory of evaluation. It aims to explore a large search space of a specific problem by producing a population of a set of solutions Submitted to a continuation of generations, develop toward the optimal solution that optimizes a fitness function of this problem.

The functioning of an evolutionary algorithm takes place around three successive stages which are;

* **Selection:** is an essential operator which aims at allowing the best individuals of a population to reproduce further to a comparison between them based on the values of their fitness functions.
* **Crossover:** is an binary operator acting on a set of individual, it allows the transmission of the characteristics of the best individuals parent to the new individuals children by replacing the randomly chosen dimensions of the individual parent with those of another individual parent to obtain two different children.
* **Mutation:** is the operator which allows the modification, with a certain probability, of one or several nodes of the selected individual, to introduce some variability into the population.

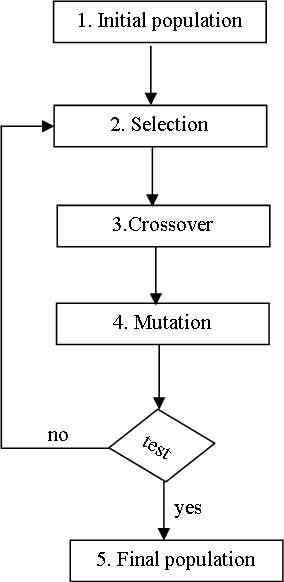


Fig 12. Evolutionary algorithm’s functioning

During the process of comparison of the individuals of the population, the evolutionary algorithm taken into account four fundamental concepts:

**-Pareto optimality:** the definition of Pareto optimality states that *x*\* is Pareto optimal if no feasible vector *x* exists which would improve some objectives without causing a simultaneous  
worsening in at least another one. ( Bechikh, 2013)

**-Pareto dominance:** a solution1 is said to dominate another solution2 if and only if the fitness functions of solution1 is partially less than the fitness function of solution2.

**-Pareto set**: is the set of solutions which called non-dominate solutions because they aren’t dominated by any others. The image of this set of solution in the objective space called **Pareto front**.

The evolutionary algorithm has a crucial advantage when it comes to solve a multi-objective problem since it treats a population of solutions instead of a single solution. Indeed, the use of evolutionary algorithm in our work results from the need to generate a set of evaluation rules and to reach this goal we use one of the multi-objective evolutionary algorithms which is MOEA/D.

MOEA/D a multi-objective evolutionary algorithm based on decomposition. It decomposes a multi-objective optimization problem into a number of scalar optimization sub problems and optimizes them simultaneously. (Zhang and Lui, 2007)

During every generation, the population is a collection of the best solution found for every sub problem. The optimal solutions of two neighboring sub problems should be similar. Every sub problem is optimized basing on information received from its neighboring sub problems.

According to (Zhang and Lui, 2007), MOEA/D has several features:

* MOEA/D introduces the decompositions approaches into multi-objective evolutionary computation.
* MOEA/D solve many issues of multi-objective evolutionary algorithm such as fitness assignment and diversity maintenance because it optimizes many sub problems rather than a whole problem.
* MOEA/D has many advantages then the others evolutionary algorithm in terms of complexity and solution quality.
* MOEA/D can incorporate the objective normalization technique.

**Algorithm of Multi-Objective Evolutionary Algorithm/Decomposition *MOEA/D***

**Inputs:** values of metrics

**Inputs:** set ofpreference of user.

**Inputs:** set of problems of AUI.

**Inputs:** the trace

**Outputs**: evaluation rules

1. Initialization of the population
2. Update population*: For i=1,…,N Do*

* **Reproduction**: Randomly select two indexes *k,l* from *B(i)* , and then generate a new solution y from*xk* and *xl* by using genetic operators.
* **Improvement**: Apply a problem-specific repair/improvement heuristic on *y* to produce y’.
* **Update of** *z*: For each *j= 1,…,m,* if *zj < fj(y’)*, then set *zj = fj(y’).*
* **Update of Neighboring Solutions**: For each index, the set.
* **Update of each population: Remove** from each population all the vectors dominated by *F (y’)* .Add *F (y’)* to each population if no vectors in each population dominate *F (y’).*

1. Stopping Criteria: If stopping criteria is satisfied, then stop and output each population. Otherwise, go to 2.
   * 1. **Adaptation of MOEA/D:**

The four following subsections describe our adaptation of MOEA/D to our problem that generates a set of evaluation rules for adaptive user interface.

* + - 1. **Individual representation:**

Our research consist on elaborate a set of evaluation rules for adaptive user interface so we need to have a formative form for this rules.

In fact, we can consider that our population is composed of a set of solutions that are also a collection of rules. Our rule presented as a set of IF-THEN.

Consequently, a detection rule has the following representation:

**The context of use**

**=**

**Value**

**Problem**

**Metric**

**(<=,>=)**

**Value**

Fig13: representation of the rule

The IF clause correspond to the combination of some preference of users (age, motivation, etc.) with their following possible values (low, medium, high), and a metric with its threshold value using the logic operator AND. Besides, the THEN clause highlights the type of problem detected on the adaptive user interface.

* + - 1. **Generation of an initial population:**

This step takes in input the preference of users (experience,age,motivation, etc.), the calculated metrics resulting from the parser (regularity, density, simplicity.etc.), the trace that contains the bases of examples and the list of possible problems . Note that each attribute of the preference can take several values: low, medium and high and the metrics can take a number between 0 and 1. This step allows extracting an initial population of rules in a random manner from a possible combination of inputs.

**-*Encoding:*** the EA needs a significant coding of individuals to facilitate the processing of the algorithm. Indeed, the coding can have a significant impact on the way in which examples are processed. The rules will undergo a process of improvement; they must then be presented as a ArrayList of a new type Rule.

|  |
| --- |
| **R1:** IF (Interest=Low) and ( Sequence<=0.31 ) THEN (not\_arranged\_Interface) |

Fig13: Encoding rule.

* + - 1. **Evaluation of population:**

In this step, we evaluate the solutions of the initial population in order to estimate and discover the interesting rules. This evaluation is based on the calculation of the fitness function which will determine the quality of each rule by referring to the trace. The quality of generated solutions is evaluated using the fitness function F (x) given in equation (1). The idea is to improve the quality of evaluation rules by reaching the two objectives of the problem:

-maximizing the number of problem in each solution

- minimizing the number of rules in each solutions.

Our fitness function is calculated using the following formula:

Where, x is the decision variables of the problem.

N: number of solution by population.

Nb: number of rules by solution.

Rate: is the rate of problem by solution an it is calculated as following :

* + - 1. **Genetic operators:**
         1. **Crossover:**

the genetic operator crossover used is the one-point crossover that is first applied to the two solutions and generates one child solution. The crossover operator swaps rules from one solution to another.

**Solution1**

|  |
| --- |
| **R1:** IF (Interest=Low) and ( Sequence<=0.31 ) THEN (not\_arranged\_Interface) |
| **R2:** IF (Age=High) and ( Regularity<=0.40 ) THEN (Irregular\_Interface ) |
| **R3:** IF (User Experience=Meduim) and ( Regularity<=0.11 ) THEN (Irregular\_Interface) |
| **R4:** IF (Motivation=Low) and ( Density>=0.82 ) THEN ( charged\_Interface ) |

**Solution2**

|  |
| --- |
| **R1:** IF (User Experience=Meduim) and ( Simplicity<=0.56 ) THEN (complex \_Interface) |
| **R2:** IF (Motivation=Meduim) and ( Grouping<=0.5 ) THEN (Low\_guidance ) |
| **R3:** IF (Age=High) and ( Simplicity<=0.3 ) THEN (complex \_Interface) |
| **R4:** IF (Age=Meduim) and ( Sequence<=0.06 ) THEN (not\_arranged\_Interface ) |

**Child 1**

|  |
| --- |
| **R1:** IF (Interest=Low) and ( Sequence<=0.31 ) THEN (not\_arranged\_Interface) |
| **R2:** IF (Age=High) and ( Regularity<=0.40 ) THEN (Irregular\_Interface ) |
| **R3:** IF (Age=High) and ( Simplicity<=0.3 ) THEN (complex \_Interface) |
| **R4:** IF (Age=Meduim) and ( Sequence<=0.06 ) THEN (not\_arranged\_Interface ) |

**Child2**

|  |
| --- |
| **R1:** IF (User Experience=Meduim) and ( Simplicity<=0.56 ) THEN (complex \_Interface) |
| **R2:** IF (Motivation=Meduim) and ( Grouping<=0.5 ) THEN (Low\_guidance ) |
| **R3:** IF (User Experience=Meduim) and ( Regularity<=0.11 ) THEN (Irregular\_Interface) |
| **R4:** IF (Motivation=Low) and ( Density>=0.82 ) THEN ( charged\_Interface ) |

Fig14: an example of the crossover process

**3.4.2.3.2 Mutation:**

The genetic operator mutation used is the standard mutation operator. It mutates one rule of solution to produce a new solution.

**Solution X**

|  |
| --- |
| **R1:** IF (Interest=Low) and ( Sequence<=0.31 ) THEN (not\_arranged\_Interface) |
| **R2:** IF (Age=High) and ( Regularity<=0.40 ) THEN (Irregular\_Interface ) |
| **R3:** IF (User Experience=Meduim) and ( Regularity<=0.11 ) THEN (Irregular\_Interface) |
| **R4:** IF (Motivation=Low) and ( Density>=0.82 ) THEN ( charged\_Interface ) |

**Child**

|  |
| --- |
| **R1:** IF (Interest=Low) and ( Sequence<=0.31 ) THEN (not\_arranged\_Interface) |
| **R2:** IF (Age=Low) and ( Regularity<=0.40 ) THEN (Irregular\_Interface ) |
| **R3:** IF (User Experience=Meduim) and ( Regularity<=0.11 ) THEN (Irregular\_Interface) |
| **R4:** IF (Motivation=Low) and ( Density>=0.82 ) THEN ( charged\_Interface ) |

Fig15: an example of the mutation process

Once reproduction and crossover have been applied according to given probabilities, parents and children are merged and the newly created generation of individuals is evaluated by the fitness function F (x). This process is repeated iteratively, usually for a fixed number of generations. The result of evolutionary algorithm (the best solution found) is the fittest individual produced along all generations.

* 1. **Conclusion**

In this chapter, we present our two contributions of our research which are the proposed metrics of adaptive user interface evaluation and the meta-heuristic used to generate a set of evaluation rules that evaluate a several adaptive interfaces.

In the next chapter, we will discuss the experimentation of our evolutionary algorithm . Moreover, we will present a comparison study of our used multi-objective evolutionary algorithm with another mono-objective evolutionary algorithm.