RESEARCH ARTICLE

# ECG signals filtering and analysis using the Compact Genetic Algorithm Based on Abstract Data Types in GPU/CUDA

**Maciel A.**[1*], **Vieira R. V.**[2]

[1] Federal Rural University of Pernambuco - IKAS/CODAI, Av. Dr. Francisco Correia, 643, São Lourenço da Mata, Pernambuco, 54740-000 Brazil; andrilenef@yahoo.com.br

[2] Computer Institute, Federal University of Alagoas - UFAL, Campus A/C Simões, Av. Lourival Melo Mota, S/N - Tabuleiro do Martins, Maceió, Alagoas, 57072-900 Brazil; rvvl@ic.ufal.br

*Corresponding author.

***Abstract:*** This paper presents the process of adaptive filtering of cardiovascular disease signals from the processing and cleaning of ECG signals developed by the Compact Genetic Algorithm Based on Abstract Data Types (CGAADT), implemented in MATLAB using GPU/CUDA architecture from the examples of the base of MIT-BIH data. The results show that CGAADT can improve the filtering, cleaning, detection and diagnosis of arrhythmias using a single algorithm (CGAADT) in the adoption of a representation for the population with fixed size of chromosomes, pre-established by fragmentation of the GPU base when implemented in high performance systems, aiming to improve the health systems offered to patients with cardiovascular problems.

***Keywords:*** Electrocardiogram; genetic algorithm; adaptive filter; graphics processing unit; compute unified device architecture

**Received:** October 12, 2019 **Accepted:** December 26, 2019 **Published:** March 18, 2020

# 1. Introduction

Cardiovascular diseases (CVDs), or diseases of the circulatory system, are the largest of all endemic[1] in developed countries, and this behavior has been occurring in recent decades in emerging countries, for which health statistics indicate that cardiovascular diseases occupy the first or second place as cause of death[1]. Diseases associated with patients with heart disease may aggravate their cardiovascular conditions and vice versa, as well as interfere in the diagnosis and therapy of these diseases[3].

Cardiopathy or cardiovascular disease can be described as: heart failure, arrhythmias, hypertension, coronary artery disease, valvular diseases, endocardial, myocardial and pericardia diseases, aortic diseases, among others[1,2]. The identification of these diseases is usually performed by electrocardiogram exam, which record the functionality of the heart over time[4].

According to Moffa[5], the wave detection process of an electrocardiogram presents difficulties related to its analysis due to the oscillations in the signal, absence of uniformity in the wave morphology that compose it and the appearance of noises during its extraction. These noises in turn can be visualized during the collection of these signs and hide important characteristics typical of cardiovascular diseases described in the morphology of the heart signal, among


doi: 10.18063/phci.v3i1.1076


1 The endemic differs from the epidemic because it is of a more continuous and restricted character[5].

which the arrhythmias[2] stand out.

Cardiac arrhythmias represent any change in the regularity, frequency, site of origin of the impulse, or abnormality in the conduction of this impulse, in order to modify the normal sequence of depolarization of the atria and ventricle[1].

The efficacy in the diagnosis of a cardiac arrhythmia depends on the capture of the signal[6], its filtering[7], detect segments and complexes[8,9] for a possible diagnosis[10], which are divided in three steps: 1. Signal capture is considered if the patient is at rest or moving. In the filtering step, the captured signal must be softened, but care must be taken to ensure that the information responsible for the characteristics of the cardiac arrhythmias is not altered. 2. Detection of the components of the heart signals is necessary to identify the periods of the ECG to later identify the waves, segments and complexes present in each period. 3. Diagnostic of the arrhythmias is necessary to have a model of each arrhythmia and to verify if the signal analyzed may or may not be considered an instance of this model. All these steps should be performed at a time considered by the doctor to be able to help save lives. The diagnosis of such arrhythmias can be difficult to identify, even with the help of electrocardiogram and other methods of clinical investigation[11]. The main points for its identification are related to the constant morphological changes in the waves that form the electrocardiogram signal, such as: P wave, PR follow, QRS complex, ST follow and U wave[5,11].

The adoption of the genetic algorithm based on abstract data types for the system is based on the feedback property of this algorithm during its execution. This feature allows the analyzed signal considered abnormal for the patient under observation to be filtered, identified and diagnosed by the system with a single approach. However, the resolution of each of these tasks can take weeks or months to be performed, when they are processed in computational terms that do not take into account the intrinsic parallelism of the algorithm. Still, simultaneous parallel processing of these functions can dramatically accelerate the filtering resolution of identification and diagnosis of cardiac arrhythmias from the ECG signals.

The function of data dimensionality treatment of ECG signal processing almost always requires the combination of various algorithms for its processing, making the process slow and infeasible for rapid diagnosis of anomalies. In general, more than one algorithm is used to represent and process these tasks sequentially in isolation. Therefore, this article presents the implementation of an instantiation of genetic algorithms based on abstract data types in a parallel accelerator architecture, based on a hardware component, called CGAADT (Compact Genetic Algorithm Based on Abstract Data Types)[12], as a solution to quickly and accurately filter, detect and diagnose sinus cardiac arrhythmias, atrial flutter, and atrial fibrillation from the electrocardiogram examination. This approach drastically reduces the convergence time of these algorithms, making it useful in solving complex problems such as the vital sign monitoring system.

# 2. The ECG signal filtering system

The detection and diagnostic filtering system presented in this article represents a first step towards generating an intelligent system to monitor the simple ECG signal by working with a single computational approach, capable of encompassing more than one of the steps of a monitoring system of medical signs.

One of the advantages of this implementation is the simplicity of the architecture, which makes it possible to increase the representation of the number of arrhythmias to be diagnosed and the amount of ECG analyzed by the system, using only one accelerator card, at no additional cost.

The computational system is high performance, being able to be expanded internally in the existing co-processing unit, or externally with the addition of more units of co-processing (GPUs), depending on the complexity and performance requirement.

Electrocardiogram (ECG) signals have specific characteristics according to signal morphology and ECG wave types. Arrhythmias may hide many heart diseases associated with patients with heart disease and aggravate their cardiovascular conditions, as well as interfere with the diagnosis and therapies of such diseases[5].

Most of the algorithms that deal with the filtering, identification and diagnosis functions for ECG signal treatment treat each of these features in isolation, because there is no algorithm in the literature that performs all the functions at

the same time.

In addition to these aspects, it is still necessary to deal with the adequate accomplishment of the cleaning of the signal that has a high dimensionality, without loss of information with the processing of data of fixed or floating point with the modeling of the algorithm, so that it can be applied to any kind of similar problem of ECG signal processing.

# 3. The genetic algorithm of Holland

The genetic algorithm proposed by Holland[13] works on a population of possible results (chromosomes) for a problem, which are improved through several iterations in order to generate an optimal chromosome for the problem. The mechanism of transformation of one population into another follows the principle of natural selection, described by Darwin[14], according to which, in nature, the chromosomes of a population compete with each other for resources such as food, water and shelter[15]. The chromosomes of a species that present the most environment friendly characteristics, called adapted chromosomes, are more likely to survive than chromosomes that do not have these characteristic. Adapted chromosomes are preferred by the chromosomes of their species to procreate, thus generating a greater number of offspring potentially adapted to the environment since the characteristics presented by an individual represent the results of the combination of characteristics inherited from their parents[15]. **Figure 1** presents a pseudocode for the genetic algorithm of Holland[13].

```
t ←0
    generates p
    calculates the value of the adaptation of chromosomes of p
    while (stop condition isn't satisfied) do
    t ← t +1
    p ← selects h chromosomes of population p
    ^ p ← applies the crossing and mutation operation on the
    chromosomes
    of the population p
    p ← replaces l chromosomes of the population p by l chromosomes of
    the population ^ p
    calculates the value of the adaptation of chromosomes of
end of the while
```

**Figure 1.** Pseudocode of a genetic algorithm[15]

According to this pseudocode, a genetic algorithm begins with the generation of the initial population *p* consisting of *n* chromosomes, which can be supplied by the user, randomly generated or constructed by an algorithm based on the existing knowledge about the problem to be solved.

Then, the adaptation of the chromosomes of the population is calculated, to verify the quality of these chromosomes for the problem investigated. If the stop criterion defined for the problem is not satisfied, then a new cycle is executed.

In this new cycle, the iteration counter *t* is selected, chromosomes are selected for the generation of new offspring, that is, they generate offspring, the birth of some of the chromosomes and the death of the existing chromosomes in the poor quality population, for the problem analyzed. The stop criteria used by genetic algorithms are: find a chromosome with adaptation *x* or reach the *j-th* iteration[15].

The population of the chromosomes habited of generating *p* descending is constructed by the selection of *h* chromosomes from the *p* population based on the adaptation of the chromosomes or a measure relative to the merits of the chromosomes within the population. The chromosomes of population *p* are submitted to the action of the genetic operators of crossing and mutation to generate offspring[15]. The frequency with which the chromosomes of population *p* generate offspring is established by a measure of probability associated with each operation[15].

The new population *p* is formed by 1 chromosomes of the population *p* and by *n-1* chromosomes of the population *p*. The criterion adopted for the birth of the chromosomes of the population *p* is the same as that used for the selection of the progenitor chromosomes of the population *p*, which is applied in reverse order to determine the chromosomes of the population *p* that should die. Finally, the adaptation value of the chromosomes of the newly constructed *p* population is calculated[15].

The chromosome is a binary vector of fixed size *m*, which stores the result of the problem in a discretized fashion. And the population is represented by a vector of fixed size *n* of chromosomes. **Figure 2** shows the definition of the data structures manipulated by the genetic algorithm of Holland[13].

```
types
binary = {0,1}
chromosome: array of dimension n de binary
population: array of dimension m de chromosome
```

**Figure 2.** Definition of the data structures used in Holland's genetic algorithm[15]

Adaptation of a chromosome is defined by the objective function and intention of the problem. The objective function of the problem, represented by $\hat{o}: D \rightarrow \mathbb{R}$, receives an element of the domain of problem *D*, and returns the measure of the degree of satisfaction of this element to the problem. The intention of the problem informs if the result is the element with greater or lesser degree of satisfaction. Thus, if the intention of the problem is to find the element with the highest degree of satisfaction, then the adaptation function *f* of the genetic algorithm will be equal to $\hat{o}$, otherwise the adaptation function *f* of the genetic algorithm will be equal to $-\hat{o}$. This convention is made necessary because the genetic algorithm by definition is always looking for the most adapted chromosome[13,16]. The condition of stopping the genetic algorithm of Holland may be the least satisfactory adaptation value, a number of iterations determined, or any combination of these two conditions, further details of this algorithm can be seen in Vieira[15].

# 4. Specification of Compact Genetic Algorithm Based on Abstract Data Types (CGAADT)

The GAADT model in Vieira[15] presents the basic definitions of the genetic algorithm based on abstract data types, with the representation of the basic types: base, gene, and chromosomes.

The modeling of the GAADT in Vieira[15], presents the basic definitions of genetic algorithm based on abstract data types, with the representation of the chromosome types, gene, and base. The GAADT architecture, the process of formation of alphabets (gene, base, and chromosome), representing 70% of the algorithm processing time when run on conventional CPUs. In the new compact version of CGAADT the chromosome representation is divided into two levels:

- The abstract type basis (*B*) is represented by $B_{Host}$, run directly in CPU;
- The chromosome types and gene run in Compute Unified Device Architecture (CUDA).

The CGAADT algorithm starts the process of forming the elementary units to generate the alphabets. This process allows the algorithm to perform the genetic material formation in different platforms, from the creation of the bases in the CPU and the genes and chromosomes in the GPU, so that the genetic material is fragmented and processed in parallel. In **Figure 3**, it presents the block diagram referring to the data flow of the CGAADT algorithm. The input data are represented by ECG signals containing the twelve leads.

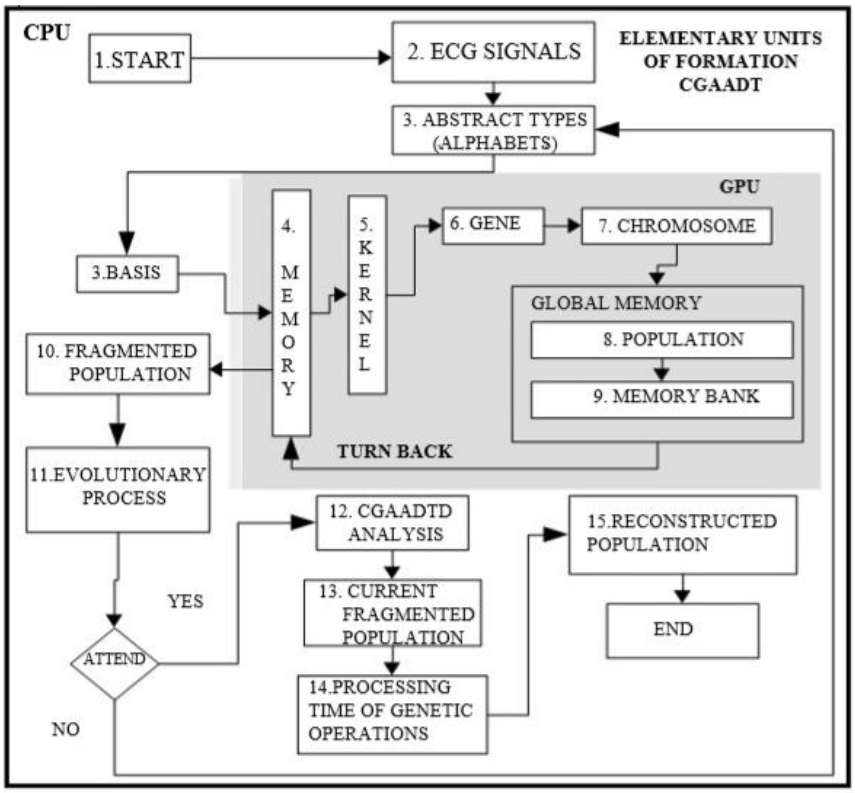


**Figure 3.** CGAADT DataFlow[12]

The CGAADT algorithm initializes the process according to its steps:

1. Input data is represented by the ECG signal containing the twelve leads.

2. These signs are read.

3. Begin the process of forming elementary units to generate the alphabet. This process allows the algorithm to perform genetic material formation in different architectures. It starts the process of creating the base abstract term, which runs on the CPU.

4. The abstract base term is routed to GPU memory.

5. Subsequently being automatically forwarded to the GPU kernel.

6. After the formation of the abstract base term, the genetic material is fragmented in the GPU so that it can form the complementary genetic material (gene and chromosome), run in parallel.

7. Fragments of the abstract gene and chromosome terms are stored in GPU global memory.

8. The process of creating a new population begins.

9. After the creation of this new fragmented population, it returns the memory of GPU (4), to be further fragmented to the CPU.

10. With this new fragmented population, the evolutionary process begins.

11. The evolutionary process performs the functions of genetic operators (selection, crossing, reproduction, mutation, and insertion of descendants in the population).

12. If the evolutionary process is satisfied, this population is directed to the new module called CGAADT ANALYSIS.

13. In this module, the analysis of population quality at the noise level is verified.

14. The processing time of genetic operations is evaluated.

15. The new population is rebuilt without loss of information.

The application of CGAADT requires a definition of specific elements in an environment featuring the problem in focus.

## 4.1 Basic types

### 4.1.1 Definition (Basis)

The basis type for the construction of the adaptive filter instantiated by CGAADT for processing cardiac signals is formed by $B_{PointsWavesHost}$ the set times of the ECG for each lead executed on the CPU. The ensemble $B_{NamesWavesHost}$ e the set $B_{Host\lambda}$ containing innocuous leads.

$$B_{Host} = B_{PointsWavesHost} \cup B_{NamesWavesHost} \cup B_{host\lambda} \quad (1)$$

The term $B_{host\lambda}$ is formed by the electrical phenomenon recorded by the electrocardiogram (ECG), the deflections that form a particular junction and the periods representing the end of ventricular depolarization and the start of repolarization.

The term $B_{NamesWavesHost}$ is the set *{waveP, waveQ, waveR, waveS, waveT, waveU}* which contains all the waves segments and complexes, which may be detected in the ECG examination, contained in the database MIT-Databases[10]. The identification and classification of waves that make up the ECG signals, obey the standard model of Willen Einthoven[17].

The elements are the set $B_{PointsWavesHost}$ are ordered pairs $X = (x, y)$, where $X \in N \times R$ that contains values that we can extract the morphological properties of waves in the ECG host (amplitude and time intervals).

The ensemble $B_{Host\lambda}$ formed by the $\lambda$ element, which represents the wave whose morphology is within the normal pattern.

The characteristics (genes) in the relevant GPU to the problem treated in this paper are $g_D = \{wave_P, wave_Q, wave_R, wave_S, wave_T, wave_U\}$, where $D$ is the device, that are part of the same derivation recorded by the ECG. The set representing these waves is $G_{DElements}$, which is formed by the junction of the basis of sets $B_{Host}$. The elements $G_{DElements}$ contain values from which we can extract the morphological properties of the ECG elements (amplitude, duration, and interval) used in the detection process. The structure adopted for the whole $G_{DElements}$ elements is $\text{element}_i =$ where i $\in$ # (don't care symbol can be replaced by an alphabet symbol adopted for chromosome), where:

$$\text{name} \in B_{\text{NameswavesHost}} \quad (2)$$

$$(x^-, y), (x^p, y^p), (x^+, y^+) \in B_{PointsWavesHost} \quad (3)$$

For example, if a gene GPU represents the QRS complex shown in **Figure 4** (where point *S* is the starting point of the wave, *M* is the maximum and the endpoint *F*). The resulting gene would be:

$$g_D = \left(\text{Complex}_{QRS}, (x_s, y_s), (x_m, y_m), (x_f, y_f)\right) \quad (4)$$

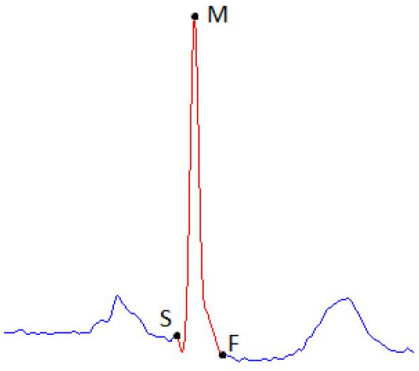


**Figure 4.** Location in a wave sample from a QRS complex with points *S*, *M* and *F*, which are responsible for determining the information wave[9]

The interpretation adopted for the elements $\text{element}_i$, where $i$ the element name; $x^-$ is the lower value of $x$ coordinate of the element to the wave; $x_p$ is the value of $x$ coordinate to the peak of the element; $x^+$ is the highest value

of the $x$ coordinate for the element; It is the $y^-$ coordinate value y for the element during $x^-$; $y_p$ is the value of $y^-$ coordinate to the peak of the element; and $y^+$ is the $y$ coordinate value for the "*Element*" element during $x^+$. When in an ECG is not entered a given wave during a period that the parameter name wave receiving λ value.

### 4.1.2 Definition (Gene)

The abstract type gene $G_D$ is a set of all elements $G_D = <b_{h1},b_{h2},b_{h3},b_{h4}> \in G_D$ in GPU, represented by $h$ (host), formed by the elements of abstract base type $B_{Host}$, where $D$ is the device (GPU).

The characteristics law is represented by the set of Axioms of Genes Formation in Device (AGFD), which shall be defined in each case, according to the semantics attributed to the gene in the CUDA device, as described in item 4.1.1.

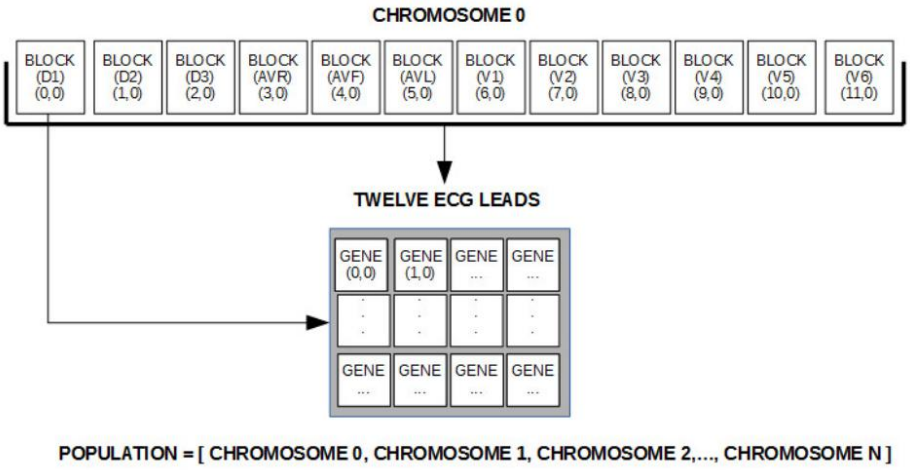


**Figure 5.** The formation of genes $g_D$ on the block grid on the GPU[12]

The axioms of set AGFD establish that:

• The basis $b_{h1} \in B_{NamesWavesHost}$;

• Basis $b_{h2},b_{h3},b_{h4} \in B_{PointsWavesHost}$;

• For every gene $g_D = <b_{h1},b_{h2},b_{h3},b_{h4}> \in G_d = [\mathrm{bk_{ij}}]_{mxn}$;

• For every gene $g_D = <b_{h1},b_{h2},b_{h3},b_{h4}> \in G_d = [\mathrm{bk_{ij}}]_{mxn}$, of size (16x12), which is the best formation of genes and chromosomes in the grid, where $i$ and $j$ represent respectively the row and column that the element occupies in $B_{NamesWavesHost}$;

• The ordered pair $b_{h2}$, should be a point whose occurrence is a period less than or equal to the ordered pair $b_{h3}$ on the ECG, at where $agfd1 = \forall g_D = <b_{h1},b_{h2},b_{h3},b_{h4}> \in G_{D,x^-} \leq x_p$;

• For every gene $g_D = <b_{h1},b_{h2},b_{h3},b_{h4}>$, the ordered $h_3$ pair must be a point whose occurrence is a period less than or equal to the ordered pair $b_{h4}$ of the ECG, at where $agfd2 = \forall g_D = <b_{h1},b_{h2},b_{h3},b_{h4}> \in G_D, x_p \leq x^+$;

• The $G_{D\lambda}$ is set innocuous gene formed by the basis $g_{D\lambda} = <b_{h1},b_{h2},b_{h3},b_{h4}>$ such that: The $b_{h1}=\lambda$. The elements of this set are represented by $g_{D\lambda} = [gene_{D0},...,gene_{Dn}]$ where $n \geq 0 \in \{0, ..., 191\}$ number of threads per block.

### 4.1.3. Definition (Chromosome)

Abstract chromosome type $C_D$ is the set of all genes constructed by the definitions established by the Axiom of Chromosome Formation in Device (ACFD);

Thus, the ACFD set is specified as:

- The elements must occur in sequences of waves *P*, *QRS*, *T*, and *U*, represented by $C_{D_{Period}}$;

- The non-occurrence of an element in the block is characterized by the replacement of the gene for the missing element in the block, by innocuous gene $g_{D\lambda}$;

- Occurrence ranges of the waves do not intercept, that is, $x^+$ *element* $p \leq x^-$ *element*$^{QRS}$ and $x^+$ *element*$^{QRS} \leq x^-$

$element^T \leq x^+ element^U$;

• For every gene $g_D = <b_{h1},b_{h2},b_{h3},b_{h4}> \in G_D = [bkij]_{mxn}$, of size (16x12), which is the best formation of genes and chromosomes in the grid, where $i$ and $j$ represent respectively the row and column that the element occupies in $B_{\text{NamesWavesHost}}$;

• For a given period may not be elements in the block ($bk$) of the same type, where:

$$acfd_1 = \forall C_{D1} \in C_D(\forall(g_{D1i},\text{g}_{D2i},\text{name}^{\text{gD1}i} \neq \text{name}^{\text{gD2}i})) \quad (5)$$

Where the name is a function which returns the gene base which stores the value of the element *name* $b_{h1}$;

• Each $bk$ is composed of a lead ECG, represented by D1, D2, D3, aVR, aVL, aVF, V1, V2, v3, v4, v5, v6 (twelve lead), the grid of blocks in GPU;

• The set $C_D$ forms a one-dimensional plane (**Figure 4**) $\forall C_D \in G_d$;

• The set streaming multiprocessor (*SM*), represented by $SM=\{sm_0,...,sm_n\}$where $n \geq 0 \in \{0,..,3\}$, form $C_D$ a size $2^{11} \times 16$ so that there is the exponential explosion of population in finding the most appropriate outcome during the process convergence.

• Each $SM$ will have a maximum of 32 threads that form 6 wraps ($W$), representing scaling units needed to improve the processing of genetic operations. The calculation for the quantity wraps results from the following expression:

$$\text{W}=\frac{\text{Th}_{\text{Max}_{\text{bk}}}}{\text{Th}_{\text{Max}_{\text{SM}}}} \quad (6)$$

Where $W$ represents the number of wraps $\text{Th}_{\text{Max}_{\text{bk}}}$ (maximum number of threads in the block), and $\text{Th}_{\text{Max}_{\text{SM}}}$(maximum number of threads per *Sms*).

• Chromosome set innocuous denoted by $C_{D\lambda}$ is formed by all sets of innocuous genes that satisfy the constraints established by Axiom Chromosome Formation in the Device (ACFD), according to equation 7.

$$C_D = \{g_{D1},g_{D2},...,g_{Dn}\}, \text{ at where } n \geq 0 \in N \quad (7)$$

The population is defined by the compact genetic based on abstract data types, the model in CUDA performs partitioning of the population in the stream processing (*SM*).

The system was developed representing the following configuration: an accelerator card from NVIDIA, GeForce GT 740M, Resolution 1366 x 768, 60 Hz, and a host machine, adopted an Intel (R) Core TM i7 4500 U CPU 1.80 GHz 8GB RAM, MATLAB R2014a[18-20]. The characterization of the problem to clean the ECG signal and the CGAADT algorithm acceleration in high-performance platform is described below.

### 4.1.4 Definition (Population)

The abstract type $P_{DFRAG}$ is the set of all the chromosomes built according to Definition 4.1.3, which is $P_{DFRAG} \geq 2^{15}$ the size of the most adapted population.

## 4.2 Genetic operators

The specification of abstract data types: *basis*, *gene*, *chromosome*, and *population*, preserved the requirements contained in the definition of CGAADT the specification of the functions and relationships necessary to calculate the function CGAADT must meet all preconditions to its original setting. The following are presented the definitions of functions and relations whose specifications for problem is more concrete than the origin setting, getting subtended the roles and relationships that are not redefined in this section preserve their original definition by GAADT[15].

Given an element "$element_o$" of the patient's ECG, and limits standards of height and width for this element in the device, the function *CompareElement* returns true if the element is within the range for the height and width provided for atrial flutter arrhythmias, atrial fibrillation and other irregularities found in ECG (arrhythmias), otherwise it returns FALSE.

### 4.2.1 Definition (Compare Element)

The function compares element is formally defined as:

$$CompareElement: B_{DElement}\ R\ x\ R \rightarrow \mathbb{B}$$

$$CompareElement(Element_0,X,Y) = \begin{cases} True\ if\ ((x_0^+ - x_0^-) \leq x) \wedge \\ (((y_0^+ \geq y_0^-) \wedge (y_0^p - y_0^-) \leq \\ (y_0^+ - y_0^-) \wedge (y_0^p - y_0^+) \leq y\ \ False \\ otherwise \end{cases} \tag{8}$$

Where $o \in \{P,QRS,T,U\}$, $\mathbb{B}$ is the set of Boolean values in the device.

The degree of adaptation of the chromosome for the detection of elements must consider the height and width of the waves (P, T, Q, R, S, U), are represented in the given gene. For each element of the chromosome that meet this verification must be added another to their degree of adaptation. So to calculate the degree of chromosome adaptation we must first define a function to return the wave patterns.

Given a chromosome $C_D = \{g_{D1},g_{D2},g_{Dn}\}$ and a standard element format, the default function returns the value 1 if one of the properties to $g_D$ meet the metric properties registered for the wave, and zero otherwise.

### 4.2.2 Definition (Standard)

The occurrence of an element in a given period of ECG is provided by $standard_D$ function of the following type:

$$standard_D\ (C_D,O_D) = \begin{cases} 1\ if\ (P \in wave_P\ (O_D,II) \Rightarrow compareWave\ (wave_P,\ Y,X)) \wedge \\ (QRS \in wave_{QRS}\ (O_D,\ II) \Rightarrow comparewave\ (wave_{QRS},\ Y,X)) \wedge \\ (T \in wave_T\ (OD,II) \Rightarrow comparewave)\ wav_eR,\ Y,\ X)) \wedge (U \\ \in wave_T\ (_{OD},\ II) \Rightarrow comparewave\ (wave_P,\ Y,X) \\ 0 \qquad otherwise \end{cases} \tag{9}$$

Where: $C_D = \{g_{D1},g_{D2},g_{Dn}\}$ and *II* contains the names of all the waves.

### 4.2.3 Definition (Degree)

The degree of adaptation of a gene is a function degree of the following type: $G \rightarrow K$ such that for each gene $g_D$, $g_D \in G_D$ is associated with a unique number $k$, $k \in K$ ($K$ is an ordered body[2]), called $degree_D$ ($g_D$) and reflects, according to the interpretation adopted for the problem, a comparative stratification between gene adaptation[12].

The dominant gene is identified by the *domi* function that receives a pair of genes, one from each of the parent chromosomes, and returns the most adaptive gene if the given genes express the same trait. If the given genes do not express the same trait, then the *domi* function will return $g_\lambda$.

$$\text{degree}_D{:}G_D \rightarrow R \quad adaptation_D(g_D) = \sum_{a \in I} \text{standard}_D\ (g_D,o_D) \tag{10}$$

*I* contain the names of all the waves registered in the system.

The weight given to a gene $g_{Di}$ of a chromosome is equal to $j + 1$, where $j$ is the number of waves whose characteristics meet the standards. For example, if a chromosome has been formed by genes that meet the specifications of the waves *P*, *T*, *QRS* complex and *U*, then the function $\text{degree}_D$ will return the value 1 for each gene of this chromosome, and the total adaptation of chromosome is equal to 10.

The specification for the detection of CGAADT waves working with genetic crossover and mutation operators between adjacent chromosomes, or chromosomes that are neighbors in the temporal space. The crossover

2 It is an algebraic structure, with two operations, without its own zero dividers and provided with an order[12].

occurs when a chromosome is found that has at least one harmless gene. In this case, it generates a new chromosome with a harmless gene being replaced by a gene on chromosome neighbor. This operation allows errors in the detection of waves and/or grouping of genes for the formation of chromosomes is corrected, as shown in **Figure 6**.

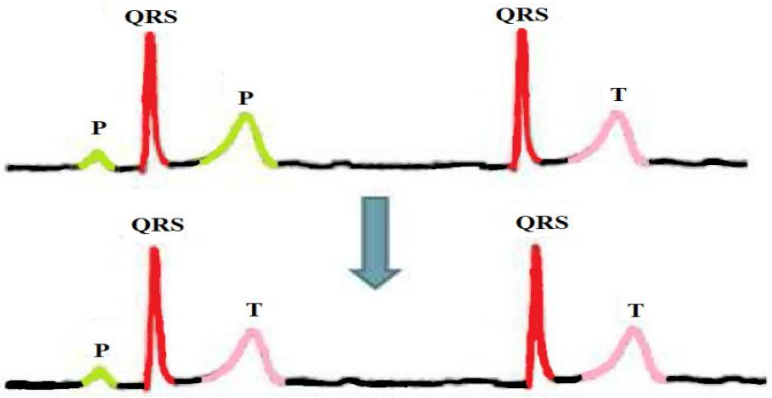


**Figure 6.** Crossover operation carried out in two chromosomes[9]

During the reproduction process shown in **Figure 6**, the algorithm identifies the pattern of *P* wave in the normal ECG from $g_{D\lambda}$, the first identifiable record is the *P* wave, representing the depolarization of the atria. The duration of the normal *R* wave generally obtained in $D2$ the ranges from 0.08 to 0.11s in adults[5].

According to Sanches[5], the *P* wave is rounded and single-phase most of the leads. Occasionally, they have small indentations and, in such cases, the distance between a peak and the other should not exceed 0.03 s. In normal individuals, the maximum amplitude of the *P* wave approaches the 0.30 mV 0.25 (2.5 to 3mm), observed in $D2$. However, the amounts commonly found ranging from 0.5 to 2 mm[5].

The *T* wave represents ventricular repolarization. The repolarization different portions of the right ventricle and the left ventricle occurs more heterogeneous so that the depolarization is recorded wider *T-waves* and smaller amplitude, *i.e.,* usually below 6 mm[5].

He rounded morphology and asymmetric, the first being longer than the second portion. Typically, the deflection of the *T-wave* has the same direction as the QRS complex, *i.e.,* multiple shunts the two phenomena positive deflections register[5].

The innocuous gene stores the patterns of waves, during the crossing process, the algorithm performs the comparison of the wave pattern established in performing the comparison and performs the crossover operation at that point in the ECG wave is the *T* wave according with the standards set by $g_{D\lambda}$.

Improvements can still be obtained by crossing operation through the sums of the adjustments to the chromosomes. However, when parent's chromosomes are less adapted to the children-chromosomes, they will be replaced by his descendants.

The mutation process will occasionally chromosomes having at least one gene of adaptation equal to "0". This operation will consist of an exchange of names of genes to try to increase the resulting adaptation of chromosomes, see **Figure 7**.

This operation is also designed to try to correct possible errors in detecting the waves, following the 50% probability of chromosome size is due to the fact that the mutations occurred in a chromosome of a given species are too large, then this chromosome would be repelled by the chromosomes of its kind, not to be considered more an equal to these[15].

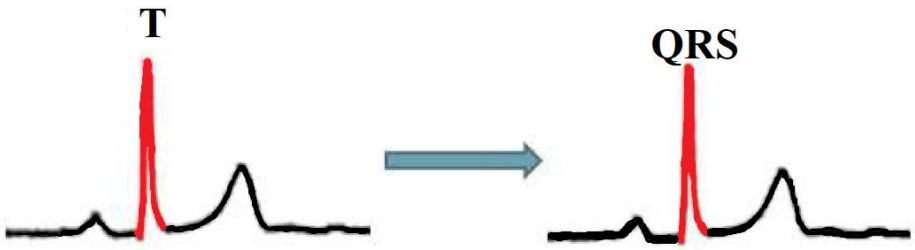


**Figure 7.** Example of name change held in mutation operation[9]

The stopping criterion adopted by CGAADT function is the maximum numbers of desired iterations and the value of the average adjustment of the current population defined by $P_{\text{point\_off\_cut}}$ (set of chromosomes below average) and are considered satisfactory for the result of the problem analysis.

These criteria are also part of the problem set requirements $R_{qD}$. To represent the most suitable chromosome, several experiments were carried out with values of 25 to 125 iterations, until the value of the fittest chromosome was not changed during the twenty iterations followed. Thus, we conclude that to reach 100 iterations, the population had more chromosome adapted to the problem.

### 4.2.4 Adaptation

The adaptation of a chromosome is an adaptation function of the following type:

$$adaptation: C_D \rightarrow K\ adaptation\ (c_D) \sum_{g \in c} \theta_{c,g} \text{ x } degree(g_D) \quad (11)$$

Where $\theta_{c,g}$ is the weight with which the g gene contributes to the adaptation of chromosome c[15].

To perform the crossover operation, we will need to be aware of two other functions, which are selection and reproduction functions. The mating operation receives two parent chromosomes, capable of mating, and returns a population whose chromosomes are formed only by the dominant genes of the given chromosomes.

The selection function filters the chromosomes that can cross and thus undergo a reproduction process, which occurs in the reproduction function, where they will be crossed. The function will return the dominant set of genes for all existing parent chromosome characteristics. The selection function receives a population $P$ and returns the $P_{DFRAG}$ subpopulation formed by the chromosomes that satisfy the requirements of problem $r$ (selection function), described by first-order logic formula, which indicates when a given chromosome is considered to cross, Further details on the selection, crossing and reproduction functions can be seen in Vieira[15] and Maciel[12].

## 4.3 Architecture CGAADT

The new version of GAADT in CUDA, called CGAADT[12], only the base abstract term runs in the CPU, while the other abstract terms called gene and chromosome run on the GPU.

This processing enables at least ten chromosomes are executed in parallel, in the case of fixed-size population according to size executed on the GPU grid of blocks, whereas the number of chromosomes is limited according to the size of the database performed in constant grid of blocks on the GPU. Therefore, the larger the size of the database and the grid of blocks, the greater the number of chromosomes in CGAADT run in parallel, which can be modified according to the adjustment algorithm architecture.

The **Figure 8** shows the architecture of CGAADT algorithm instantiated by GAADT, both algorithms work on an environment that can be modified according to the problem being addressed and how populations of chromosomes that will evolve.

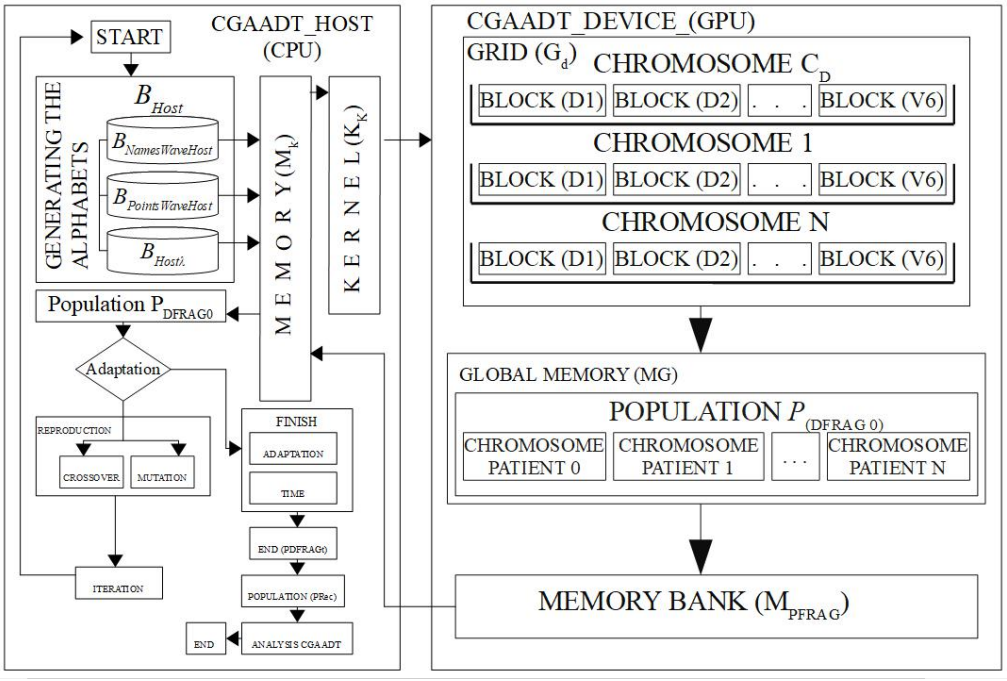


**Figure 8.** Architecture of CGAADT[12]

In the GAADT, the environment $A$ is made up of 8-tuple <P,IP,Rq,AFG,AFC,Tx,∑,$P_0$>, in this environment there is no concern with memory and all environment runs in CPU.

In CGAADT environment exposed in the CGAADT DataFlow (**Figure 3**), a genetic algorithm operates on populations of chromosomes which occur in the grid of blocks GPU (device) according to the characteristics of the environment A . This environment is presented by 13-tuples, $[P_{\mathrm{DFRAG}}, \mathrm{IP}_{\mathrm{DFRAG}}, \mathrm{Rq}_D, \mathrm{M}_m, \mathrm{G}_D, \mathrm{AGFD}, \mathrm{AGFD}, \mathrm{Tx}_D, \mathrm{SMs}, \sum, P_{\mathrm{DFRAG0}}, \mathrm{P}_{\mathrm{DFRAGt}}, \mathrm{P}_{\mathrm{Rec}}]$

where:

1. $P_{\mathrm{DFRAG}}$ is the population in GPU (device);
2. $\mathrm{IP}(P_{\mathrm{DFRAG}})$ is the set power;
3. $\mathrm{Rq}_D$ is the set of requirements (features expressed by formulas in a first-order language) the problem that influence the genealogy of the population $P_{\mathrm{DFRAG}}$;
4. $M_m$ is the set of memories $\{M_k, \mathrm{G}_M, \mathrm{M}_{\mathrm{PDFRAG}}\}$, where $M_k$ is the data transfer memory to the kernel, $GM$ is the global memory on the GPU (device) and $M_{\mathrm{PDFRAG}}$ is the population transfer memory $P_{\mathrm{DFRAG}}$ to the initial population $P_{\mathrm{DFRAG}}$ in the host, for algorithm execution;
5. $G_D$ is the thread block grid on the GPU (device);
6. AGFD is the axioms of set of formation of the population of genes in chromosomes $P_{\mathrm{DFRAG}}$ in the GPU (device);
7. ACFD is the axioms of set of formation of the population of chromosomes $P_{\mathrm{DFRAG}}$ in the GPU (device);
8. $\mathrm{Tx}_D$ is the set of pairs of chromosomes $(\mathrm{x},\mathrm{y})$, which x is a chromosome $C_D$ constructed from the chromosome $y$, by the action of crossing operation or mutation, thus registering the genealogy of chromosomes belonging to the populations generated by CGAADT during its execution;
9. SM is parallelization of chromosomes in streaming multiprocessor for scheduling the operations of the AG;
10. Σ is a set of genealogical operators acting on the population $P_{\mathrm{DFRAG}}$;
11. $P_{\mathrm{DFRAG0}}$ is a subpopulation belonging to $\mathrm{IP}(P_{\mathrm{DFRAG0}})$, called the initial population, with at least one chromosome.
12. $P_{\mathrm{DFRAGt}}$ is the most suitable people to be transferred to reconstruction by $P_{\mathrm{Rec}}$ the host;

13. $P_{\text{Rec}}$ is a population rebuilt the kernel.

The CGAADT architecture shown in **Figure 8,** the algorithm is divided by three functions: CGAADT-HOST, CGAADT-DEVICE (GPU) and ANALYSIS-CGAADT.

The CGAADT algorithm starts from the creation of alphabets: gene, basis and chromosome. The CGAADT-HOST function receives a so-called base $B_{\text{Host}}$.

This base is $B_{\text{Host}}$ comprised of three other elementary bases, which form the basic morphology of the ECG signal, stand out: $B_{PointsWavesHost}$ U $B_{NamesWavesHost}$ U $B_{hostλ}$, which are transferred to the memory $M_k$, aims to determine the profile of the data area in the kernel ($K_k$) where each thread must on the GPU, from CGAADT-DEVICE function.

To trigger the $K$ kernel, the CGAADT-DEVICE function initializes the parallelization of the database GD (grid blocks), from which is the grid blocks on the GPU. At this time, the basis $B_{PointsWavesHost}$ U $B_{NamesWavesHost}$ U $B_{hostλ}$ are converted into genes $g_D$, which represent a thread of sub-blocks formed by the sequence of elements $B_{\text{Host}}$ belonging to the set AGFD.

Then the genes $g_D$ are grouped into clusters to form chromosomes $C_D$ in the GPU, which complies with the conditions laid down ACFD.

These chromosomes $C_D$ are grouped in sets GD to form a population and this representation will ensure fairness in the evaluation of the chromosomes that comprise the population $P_{\text{DFRAG}}$ transferred to the global memory $G_M$. Each chromosome $C_D$ is a patient as the database characteristics. For each patient, represented by the chromosome $c_{\text{PATIENT0}}$ are assigned to the 12 (twelve) from the ECG leads, the defined set of chromosomes in the population is limited the number of patients in the existing database. These chromosomes $C_D$ form the initial population $P_{\text{DFRAG0}}$ will be transferred to the memory bank $M_k$, to be transferred to the kernel ($K$) and memory to boot the CGAADT algorithm. The CGAADT algorithm initializes its implementation as follows:

1. Initialize the function degree of gene adaptation is a function (equation 11) in the GPU;

2. Initialize the adaptive function $\text{adaptation}_D \leftarrow 0$. For every chromosome genes, select the dominant genes, is a function like $domi{:}G{\times}G{\rightarrow}G$. Dominant gene is a function dominant of type :

$$domi(g_{D1},g_{D2}) = \begin{cases} g_{\lambda} & if\ (g_{D1}, g_{D2}) \notin same \\ g_{D1} & if(g_{D1},g_{D2}) \in same \wedge degree(g_{D1}) \geq degree(g_{D2}) \\ g_{D2} & if(g_{D1},g_{D2}) \in same \wedge degree(g_{D1}) < degree(g_{D2}) \end{cases} \tag{12}$$

3. Initialize the function $domi(g_{D1},g_{D2})$, whereas two genes $g_{D1}$ and $g_{D2}$ refer to the same feature, $\text{degree}_{\text{D}}(g_{D1}) \geq \text{degree}_{\text{D}}(g_{D2})$ namely if return $g_{D1}$, otherwise $g_{D2}$;

4. Initialize the crossover function for all possible pairs of chromosomes $C_D$ with the population $P_{\text{DFRAG}}$ of more adapted chromosomes. Then select the dominant genes, forming all possible chromosomes with these dominant genes and include it in the population to carry the cross[12].

5. Initializes the mutation function, for all the chromosomes contained in the least adapted population, conduct exchanges of up to 50% of their genes, resulting in a more adapted chromosome $C_{\text{adaptation}}$ than the original chromosome $C_D$.

The CGAADT algorithm receives the population $P_{\text{DFRAG0}}$ and read the environment $A$ in order to submit it to the simulation of an evolutionary process, and returns a population $P_{\text{DFRAGt}}$, which will be rebuilt by $P_{\text{Rec}}$ the CPU after completion of the evolutionary process, according to the architecture in **Figure 9**. Initializes the calculation of the adjustment of the chromosome $C_D$ of the current population $P_{\text{DFRAG0}}$.

The CGAADT algorithm selects the most appropriate chromosome $C_{\text{Adaptation1}}$. If the fittest chromosome is $C_{\text{Adaptation2}} \leftarrow C_{\text{Adaptation1}}$ while the process of $\text{adaptation}_D(C_{\text{Adaptation2}}) \geq \text{Adaptation}_D(C_{\text{Adaptation1}})$ the algorithm selects a population of chromosomes adapted. Then the CGAADT, performs the crossover operation in the population

of more adapted chromosomes.

The population of chromosomes not adapted, will be selected for the implementation of mutation operator.

In this case, the chromosomes $C_{\text{Adaptation1}} \leftarrow C_{\text{Adaptation2}}$, chromosome new algorithm performs adaptation calculating and selecting $C_{\text{Adaptation2}}$.

Thus, the algorithm performs CGAADT new reading to the environment $A$, to form the new population $P_{\text{DFRAG}}$=$\text{P}_{\text{DFRAG}_{\text{CROSSOVER}}} \cup P_{\text{DFRAG}_{\text{MUTATION}}} \cup P_{\text{DFRAGt}}$. This new population is added to the environment A to completion of CGAADT algorithm.

The population of chromosomes $P_{\text{PDFRAGt}}$ are the chromosomes of population $P_{DFRAG0}$,$P_{DFRAG1}$,$P_{DFRAG2}$, ...,$P_{DFRAGt-1}$,that best meet the requirements of the problem $\text{Rq}_D$. It is said then that the population $P_{\text{PDFRAGt}}$ evolved the population $P_{\text{PDFRAG0}}$. The preservation and death of the chromosomes of the current population $P_{\text{PDFRAGt}}$ crafted by CGAADT is driven by a unary predicate called $P_{\text{cut\_off\_point}}$.

This predicate belongs to the set problem requirements $\text{Rq}_D$ on the GPU, which operates in conjunction with the SMs on the chromosome $P_{\text{PDFRAGt}}$.

The chromosomes that satisfy the predicate $P_{\text{cut\_off\_point}}$ will be part of the population $P_{\text{PDFRAGt+1}}$, while the other chromosomes of the population $P_{\text{PDFRAGt}}$ will die. The dead chromosomes can be retrieved via the taxonomy $\text{Tx}_D$ of the chromosomes of the current population to prevent them from reappearing in the next iterations of CGAADT function. This restriction serves the understanding of Darwin evolution[14], which does not contemplate the possibility of an extinct species appears again in another future time. Then CGAADT-ANALYSIS function, the population receives $P_{\text{PDFRAGt}}$ and guides the operator to rebuild the population $P_{\text{Rec}}$ in the origin data in the kernel format, which represents the algorithm the processing results involving genetic selection operations, crossover, mutation, reproduction, inclusion of offspring in the population and the waves of arrhythmias, atrial fibrillation and flutter[12].

# 5. Hardware and software infrastructure in the implementation of HOLLAND and CGAADT in GPU/CUDA

The HOLLAND and CGAADT algorithms required a minimum infrastructure for their implementation, execution and configuration, namely:

## 5.1 Implementation of CGAADT and HOLLAND in GPU (Graphics Processing Unit)

- GPU/CUDA GeForce GT 740 M card, NVIDIA, Resolution 1366 x 768, 60 Hz. -Software Matlab R2014a [18-20].
- Intel (R) Processor, Core TM i7 4500 U, CPU 1.80 GHZ, 8GB RAM; - 64-bit Operating System, Windows 8.1;

## 5.2. Performance analysis of the HOLLAND x CGAADT algorithm

In the genetic algorithm by abstract data types (GAADT) developed by Vieira[15], it defines the process of formation of the alphabets (base, gene and chromosome). The simulation performed with the CGAADT algorithm aims to perform a study of the process of cleaning the ECG signal[22-24]. The CGAADT was reformulated to manage data during the execution of the algorithm from the logistic regression model[21-22].

The regression model is designed to evaluate the computational effort of the CGAADT (GPU) and HOLLAND(GPU) algorithms as a methodology to evaluate the performance of these algorithms being processed in CPU and GPU.

# 6. Results

The database is composed of three different types of databases: the first database is formed by the specific signs of atrial fibrillation, the second by the atrial flutter signals and the third by the arrhythmia data. The age of these patients evolved in the range of 76 to 84 years and were submitted to the use of medications (**Table 1**). The electrocardiogram signals used in the experiments are composed of the leads: DII, V1, Avf (atrial fibrillation); DI, DII, Avf (atrial flutter), V5 and V2 (arrhythmia)[10].

| Databases | Sex | Age | Cardiopathis | Medicines | Exemples | Leads |
|---|---|---|---|---|---|---|
| 1 | F | 81 | FA | Atenolol, Monopril | 10000 | AVF |
| 2 | M | 76 | FLA | Lopressor | 10000 | DI, DII |
| 3 | F | 84 | A | Digoxin | 10000 | $V_5$, $V_2$ |

**Table 1.** Features ECG database. Legend: (F) Female, (M) Male. (FA) Atrial Fibrillation, (FLA) Atrial Flutter, (A) Arrhythmia[10,12].

This process allows the algorithm to perform the genetic material formation in different platforms, from the creation of the bases in the CPU and the genes and chromosomes in the GPU, so that the genetic material is fragmented and processed in parallel.

This process creates a new population, which is transferred to the CPU memory, in order to initialize the evolutionary process. If the evolutionary process is not met, the algorithm starts new processing, otherwise, the CGAADT sends the data outputs.

The first data output is represented by the current population, submitted to the reconstruction process, represented by $P_{Rec}$ so that the ECG signal is displayed. The second output is related to the log file that stores the processing time of genetic operations (selection, crossing, mutation, reproduction and insertion of descendants in the population).

The outputs of the CGAADT are sent to the analysis function (CGAADT ANALYSIS), to perform the data analysis process, from the logistic regression model to the collection of results[21].

The filtering process in the CGAADT is performed from the genetic operator of the crossing that represents the difference between $B_{Host}$ and the reconstructed population $P_{Rec}$.

The performance measure for evaluation of the filtering process in the CGAADT algorithm will be represented by the mean square error $e^2=[B_{Host}-P_{Rec}\ ]^2$, where $e^2$ is mean square error (EQM), $B_{Host}$ is system response containing the original data signal in the host and $P_{Rec}$ is response of the adaptive CGAADT filter by the crossover operation[12].

**Table 2** shows the comparative map for the processing results, with regard to ECG signal filtering. In this way, the CGAADT algorithm obtained a lower EMQ of 0.207. This result represents 66% accuracy in the filtering process, and this percentage is considered acceptable for the type of processing with a single GPU.

| DATABASES | CGAADT(GPU/MSE) | HOLLAND (GPU/MSE) |
|---|---|---|
| 1 | 1,010 | 1,020 |
| 2 | 0,011 | 0,018 |
| 3 | 0,026 | 0,032 |
| 4 | 0,074 | 0,149 |
| 5 | 0,063 | 0,174 |
| 6 | 0,054 | 0,207 |
| 7 | 0,008 | 0,257 |
| 8 | 0,261 | 0,510 |
| 9 | 0,264 | 0,513 |
| 10 | 0,296 | 0,546 |
| EMQ | 0,207 | 0,343 |
| Filtering | 66% | |

**Table 2.** Comparative map in the filtering process based on the EMQ algorithm CGAADT (GPU) and Holland (GPU)[12]

In **Figures 9** and **10**, they present the filtration process in the DI lead (atrial flutter) e lead V5 (arrhythmia signal) by the CGAADT. The signal highlighted in blue represents the original data base and the signal highlighted in red represents the filtering process after the crossover operation performed by the CGAADT.

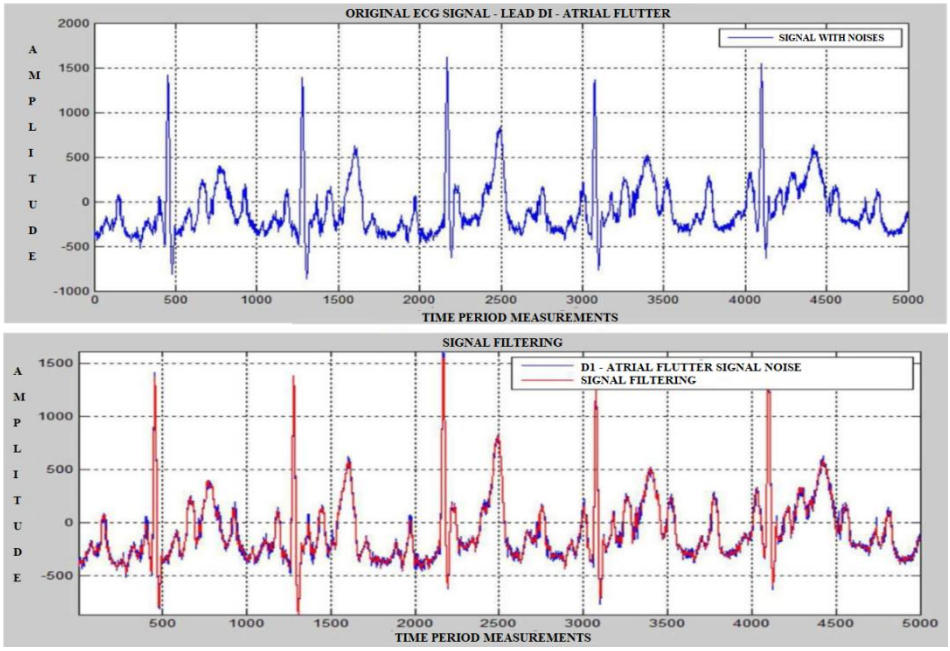


**Figure 9.** Filtration process (atrial flutter signal)[12]

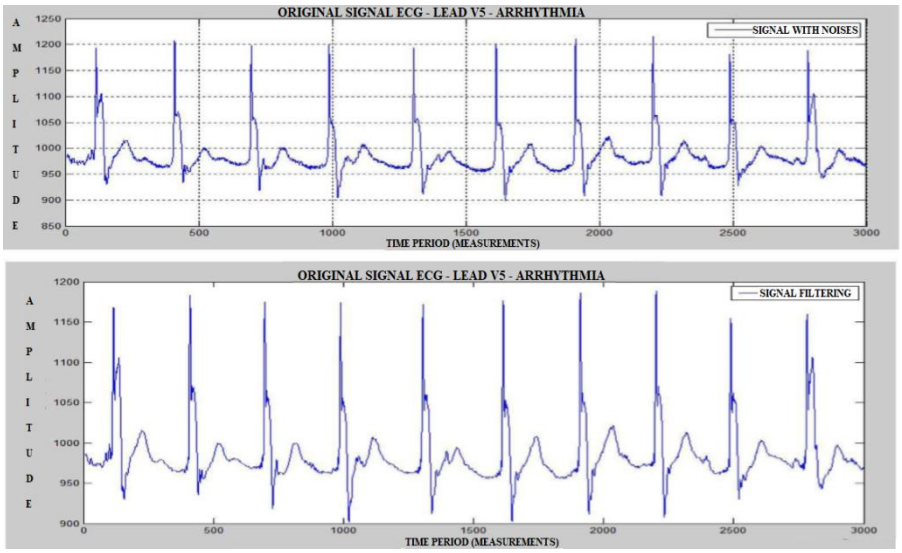


**Figure 10.** Filtering process (arrhythmia signal)[12]

| ALGORITHM | GPU (Seconds) |
|---|---|
| CGAADT(GPU) | 2.31 |
| HOLLAND(GPU) | 8.75 |
| Gain in Runtime (%) | 73.6 |

**Table 3.** Filtering process (arrhythmia signal)[12]

The CGAADT algorithm developed in GPU obtained an average real-time gain of processing based on the execution of 12 ECG leads per genetic operation per second, which results in the gain of 17.43% in the selection, 1.39% in the crossing, 1.12% in the mutation of 9.02% in the reproduction and 15.11% in the process of insertion of descendants in the population. These indexes represent a gain in the execution time of the algorithm of 73.6% related to the genetic algorithm developed in GPU, for example, Holland algorithm. The average transfer time of the abstract term $B_{Host}$ to memory at kernel firing is 5.0719 (seconds) in real-time processing (represents the execution of 12 ECG leads per converted genetic operation in seconds).

Considering that, for each database generated (**Table 1**), we get 10000 (data example), totaling 30000 data examples per cycle. Thus, 100 cycles were performed in the tests to compare the best results. Therefore, the parallel version of the CGAADT was able to reduce the execution time of the GAADT algorithm. The tests were performed on several types of ECG signals, including normal and arrhythmia signals.

# 7. Conclusion

The development of the genetic algorithm CGAADT obeyed the basic instantiation used by GAADT to use the representation of the stratified chromosome in three levels of perception (chromosome, gene and base). The difference between these two algorithms lies mainly in the way the manipulated genetic material is structured in the GPU, as specified by the CGAADT algorithm.

This algorithm obtained a gain of 73.6% in the real-time execution of the algorithm and in the process of collecting results. This gain is mainly due to the structuring of the base in the formation of the genetic material in the GPU. Among the operational advantages presented by this algorithm are: the filtering, cleaning, detection and diagnosis of arrhythmias using a single algorithm and the adoption of a representation for the population with fixed size of chromosomes, pre-established by the fragmentation of the based on the GPU.

The efficiency of the CGAADT algorithm in identifying biomedical signals and adopts the same modeling as the original version of GAADT. The algorithm is applied to any type of problem that uses fixed and floating point.

CGAADT models the chromosome representation independent of the adopted solution and the problem to be treated. The crossover operator constructs new chromosomes only with the characteristics responsible for adapting the parent chromosomes to the environment, which will be referred to as "dominant genes". The unadapted chromosomes before disappearing, being subjected to the action of genetic operators, as a way to ensure the presence of characteristics to the environment of these organisms in the next generations. CGAADT architecture is adapted to the environment in search of the best solution.

Among the operational advantages presented by CGAADT compared to the Holland (1975) algorithm are: chromosome adaptation is compromised with the relevance of GAADT encoded information. The GAADT search strategy is highly objective, due to the use, as a criterion of chromosome preservation in the next population, of a function based on the adaptive dynamics of the population. The explicit presence of the environment in GAADT functionality gives this algorithm the ability to handle highly dynamic problems. Applying GAADT to a problem requires defining the environment elements specific to the problem at hand, which must meet the properties set in the environment definition. The release of the use of representations suggested by the problem analysis and, consequently, the possibility of choosing different representations of the traditional fixed-size binary vector adopted by Holland and related models. The use of a representation for the population of varying size and not limited, preventing the loss of adapted chromosomes generated by the action of crossover and mutation operations, which may be discarded for exceeding a predefined size.

The definition of the weight function of a gene in the chromosome allows us to model the dependency relationships between the given gene and the other chromosome genes, thus ensuring the possibility of applying CGAADT to problems with epistasis. The behavior of the crossover operation ensures the transmission of characteristics relevant to the parent chromosome problem to the child chromosome, thus leading to a more objective search for promising chromosomes. This contributes to the average adaptation of the population is strictly monotonic. Applying the mutation operation on unadapted chromosomes prevents the problem-relevant genes belonging to these chromosomes from being lost, and ensures population diversity.

## Conflict of interest

No conflict of interest was reported by the authors.

## References


1. Oliveira WJ and Pedrosa LC. Doenças do Coração: Diagnóstico e Tratamento. 1ª edição, Revinter, São Paulo, p. 536, 2011.
2. Porto CC. Doenças do Coração: Prevenção e tratamento. Guanabara, 2005.
3. Moreira DAR. Fibrilação Atrial. 2ª edição, Lemos Editorial, São Paulo, p. 311, 2005.
4. Moffa P and Sanches P. Eletrocardiograma: Normal e patológico, 7ª edição, Roca, São Paulo, 2001a.
5. Moffa P and Sanches P. Eletrocardiograma: Uma abordagem didática. 1ª edição, Rocca, São Paulo, p. 356, 2010b.
6. Cunha PCN. Modelo de eletrocardiógrafo portátil de baixo consumo. Dissertação de Mestrado em Modelagem Computacional de Conhecimento, Universidade Federal de Alagoas, 2012.
7. Ramos VWJ. Uma aplicação do algoritmo genético baseado em tipos abstratos de dados ao problema de separação cega de fontes com não-linearidade posterior. dissertação de mestrado. Dissertação de Mestrado, Universidade Federal de Alagoas, 2011.
8. Melo BRP. An application of a new genetic algorithm to assist the detection of segments of an ecg signals. ISABEL' 11, ACM, Barcelona, Span, 2011a.
9. Melo BRP. Um Sistema Adaptativo para Detecção de Ondas de Eletrocardiografia, Dissertação de Mestrado em Modelagem Computacional de Conhecimento, Universidade Federal de Alagoas, 2011b.
10. Physionet, MIT-BIH Arrhythmia Databases. Disponível em: http://www.physionet.org/physiobank/database/mitdb. Último acesso: 28 de abril de 2019.
11. Porto CC. Doenças do Coração: Prevenção e tratamento. Guanabara, 2005.
12. Maciel A. Um filtro adaptativo de alto desempenho instaciado do algoritmo GAADT para o processamento de sinais de eletrocardiograma, Tese de Doutorado do Centro de Informática da Universidade Federal de Pernambuco, 2015.

13. Holland JH. Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. MIT Press., 1975.
14. Darwin C. The Origin of Species. 1859. Disponível em: http://darwin-online.org.uk/converted/pdf/1861_OriginNY_F382.pdf. Acesso em: 27 de março de 2019.
15. Vieira RVV. Um Algoritmo Genético Baseado em Tipos Abstratos de Dados e sua especificação em Z, Tese de Doutorado, Universidade Federal de Pernambuco, 2003.
16. Norvin P and Russel S. Artificial Intelligence: A Modern Approach, Prentice Hall, 1995.
17. Sanches P and Moffa P. Eletrocardiograma: Uma abordagem didática. 1ª edição, Rocca, 356 p, São Paulo, 2010.
18. NVIDIA CUDA C Programming Guide, Version 4.2, 1999.
19. OpenCL Programming Guide for the CUDA Architecture, V 2.3, 2009.
20. Math Works - Softwares Matlab R2010, R2012a, R2012b, R2013a e R2014a. URL: http://www.mathworks.com. Último acesso em 20/06/2019.
21. Valença M. Aplicando Redes Neurais: Um Guia Completo, p.57:58, Olinda, PE, 2005.
22. John G and Proakis V. Digital Signal Processing using Matlab. Third Edition, Cengage Learning, 2010.
23. Gao X. Diagnosing Abnormal Electrocardiogram (ECG) via Deep Learning. IntechOpen, 2019. Disponível em https://www.intechopen.com/online-first/diagnosingabnormalelectrocardiogram-ecg-via-deep-learning. Último Acesso: 15/06/2019.
24. Iqbal MN, Bomhara M, Alhassan H, Abd-Alhameed R, Eya N, *et al.* Real-time signal processing of data from an ECG. IEEE, Wrexham, UK, 2019.