**INTERNATIONAL SCIENTIFIC AND VOCATIONAL JOURNAL (ISVOS JOURNAL)**

**Vol.: 1 Issue: 1 Date: December 2017 Received: 20.12.2017 Accepted: 30.12.2017 Final Version: 31.12.2017**

ISVOS Journal, 2017, 1(1): 30-41

# Interlock Optimization of An Accelerator Using Genetic Algorithm

İbrahim Burak Koç [a], Anas Al Janadi [b], Volkan Ateş [b,1]

[a] Instute of Accelerator Technologies, Ankara University Golbasi Campus, Golbasi Ankara TURKEY
[b] Department of Electrical and Electronics Engineering, Kirikkale University, Kirikkale TURKEY

**Abstract**

Accelerators are systems where high-tech experiments are conducted today and contain high-tech constructions. Construction and operation of accelerators require multidisciplinary studies. Each accelerator structure has its own characteristics as well as similar features of accelerator structures. Control systems come to the forefront as one of the most important structures that make up accelerators. Since control systems have critical importance for accelerators, in such systems when a problem occurs, there is a danger of environmental and human safety as well as machine system. For that reason interlock systems are being developed in different structures. In the literature, FPGAs and PLCs in such interlock systems have been shown to be suitable for use in accelerators [1,7].

In this work, we describe an interlock system that evaluates the operation and protection modes of devices used in an electron accelerator. In order to ensure that this system can operate at minimum cost and maximum safety, the defined system is divided into 3 subsystems. The error messages from the control devices in the accelerator control systems are the input to the interlock system. The purpose of the interlock system that evaluates error messages is to ensure that the accelerator closes safely.

The purpose of this study is to specify which of the 3 interlock subsystems which are defined for minimum cost and maximum security should be connected to the fault outputs from the control devices. As an evaluation criterion, 6 features are defined for the control devices and each control device is weighted according to the importance of the task. In the solution of the problem, genetic algorithms were used for assigning 74 controller outputs to 3 interlock subsystems. Thanks to the Genetic Algorithm used in the study, 94.3% success rate was obtained in terms of cost and safe system.

***Keywords:*** "Optimization, Genetic Algorithm, Accelerator"

## 1. Introduction

In accelerator systems, high-tech devices are used for the control of high-energy electrons. Some of these devices are control devices for obtaining laser at the output of the accelerator, and some of them are intended to ensure that the system does not pose a risk to parameters that are important for human and environmental health as well as for healthy operation. Such advanced technology and control of high risk systems should be assessed both accurately and efficiently in terms of machine, human and environmental health as well as cost.

Control of electron electron beams in accelerators depends on many physical parameters. Control of these parameters is done through electronic systems or interfaces. In a structure where complex control systems and electronic systems exist, in the event that the control of the electron high energy beams can not be achieved, there are damages to the system, the environment and human health. This means that faults or failures in the system should be detected very quickly and should be avoided without harming the system, environment or human health.

Each electronic device used in the control of the electron packs in the accelerators has a predetermined operating range. These devices generate error signals when they start working outside of the specified range. This error signal enters a central system, which activates a shutdown procedure in such a way that it does not harm human and machine health. This central system is called the interlock system. The error output of each device is called interlock output. That is, the devices enter the interlock outputs as inputs to the interlock system and thus report the error to the interlock system. The interlock system activates different shutdown procedures by evaluating the error messages received at the input. The shutdown procedures vary according to the severity of the error signals entering the interlock system.

[1] Corresponding author. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000 .
E-mail address: volkanfire@hotmail.com

The priority of the error signals, which are input to the interlock system, is important in proportion to the damage it will give. It may be permissible to react more slowly to some error signals when it is necessary to react more quickly to some error signals. There is a need for a fast interlock system where rapid reaction is required and the system must be shut down quickly. Furthermore, the interlock system should have a very low probability of failure in continuous operation. However, connecting all the interlock outputs in the system to the fast and extreme reliable interlock subsystem increases the cost too much. For this it is the case that some of the interlock outputs have to be connected to systems that run slower and have sufficient reliability and therefore are less costly.

In this context, it is envisaged that the interlock system to be used in the accelerator will come to 3 subsystems. These are FPGA, PLC and software interlock systems.

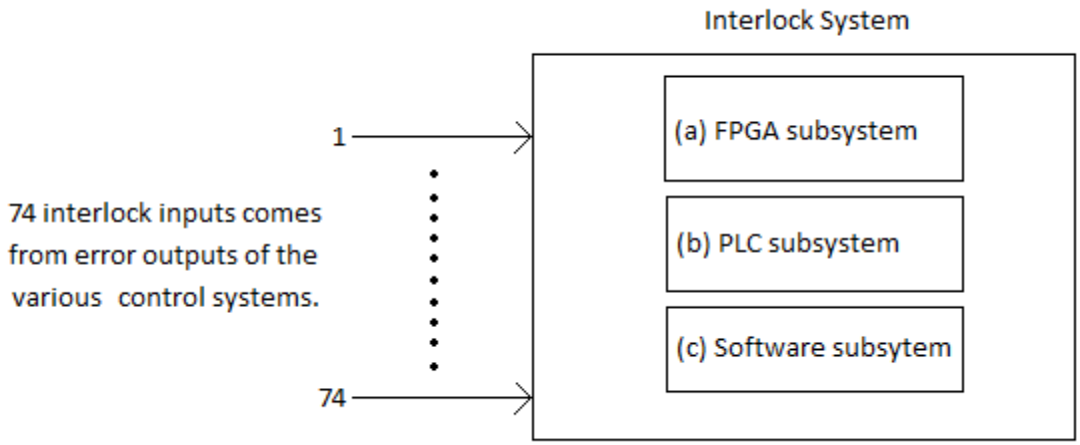



**Fig 1. General structure of interlock system**

The FPGA interlock subsystem is suitable for operations requiring very high speed shutdown procedures. The shutdown delay is around 1ms and the failure rate is extremely low, so it is highly reliable. However, it is the highest cost system.

The PLC interlock subsystem can be selected for operations requiring a high-speed shutdown procedure. The shutdown delay is around 50mS, which depends on the size of the application. Although the failure rate is low, it is higher when compared to FPGAs. Cost is high but is lower than FPGAs. It offers the optimum reliability and cost ratio within Interlock subsystems.

Finally, the Software interlock subsystem has the lowest cost but the highest probability of error. The shutdown delay is around 500ms.

Six evaluation criteria for 74 different error sources were introduced in order to predict which interlock subsystem the error outputs of the devices would enter.

***Occurrence probability:*** Likelihood of occurrence of the error.

***Risk:*** The cost of the fault when error occurs.

***Source speed:*** The speed of the source as the control system to be sent to interlock. In the other words the time elapsed between the occurrence of the error and the error output of the relevant control system device.

***Human health:*** Impacts on the human health if the accident occurs.

In some cases, it is necessary for the interlock devices' error outputs to be configurable as an input to the interlock system. For example, some error sources can be turned off during the development phase, or the protection intervals can be extended, providing flexibility during development. Such an authorization is only for high-level expert users.

Such authorizations are;

***Editable-treshold:*** The limit values for the error source can be set.

***Editable-enable/disable:*** Completely ignorability of the error coming from the error source.

There are 74 control error outputs that can cause an error in the accelerator application. For each control error output, the data set for the interlock system was created by normalizing and weighting the above 6 properties in the interval of 0-1. The data set was evaluated by expert personnel and the knowledge base was determined.

As an example of the systems in the accelerator, the simple structure of the Beam loss monitor system (BLM) is given in Figure 2. In Figure 2a, Electron beams passing through the cylindrical vacuum beam line, induce current on a wire detector located outside the beam line. According to the induced current value, it is learned how close the electron beams approach the wall of the beam line. In Figure 2a. and 2b, the minimum distance allowed to approach the wall of the beam line is shown by the broken line. As shown in Figure 2b, as the electron packages approach the allowable distance, the induced current on the wire increases. When the current value induced on the wire outside the beam line exceeds the permissible current value, the BLM device outputs the interlock output (error output).

Here the BLM system is an interlock source. The Blm device is an interlock device. The interlock device generates the interlock signal (error signal) according to the previously determined parameters in the interlock system.

The BLM system is considered as interlock source number 6 in Table 1.

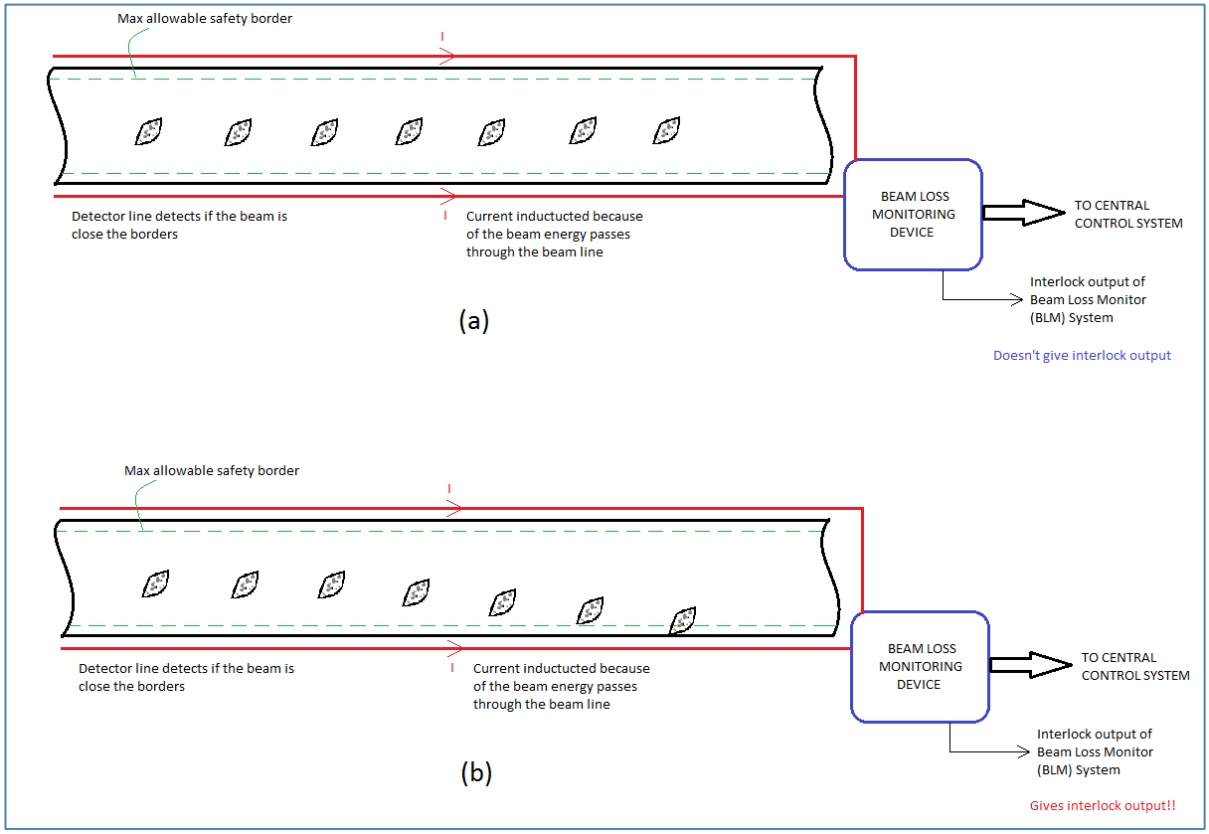



**Fig 2. Basic layout of the Beam Loss Monitoring (BLM) system as example of an interlock source**

There are many devices and parameters that allow control and adjustment of electron packs. If one of these devices goes out of control or if the parameters are selected incorrectly, the error probability increases. For this reason, the "Occurrence Probability" is estimated to be 0,6.

As a result of the high-energy electron packs striking the beam line, the vacuum line will be drilled and the vacuum will be broken. Vacuum distortion will affect many systems operating under vacuum in the electron field. Therefore, the risk factor is weighted as 0,7.

BLM system is a system that needs to react quickly as the "Risk" factor is high. The response time for system security should be around 50ms. When normalized between 0 and 1, the "Source Speed" value is 0.02 as seen from Table 1.

The maximum approach distance to the wall of the beamline must be adjustable or completely switchable in order to provide flexibility during the development phase if the expert staff uses the machine. This is shown in Table 1 by weighting the editable options (enable and threshold) to 1.

Although the BLM interlock system is risky for the machine, the human health effect is weighted as 0,3 so that there will be no human in the environment during the operation.

This system needs to be connected to the FPGA from interlock subsystems because the risk ratio is high and it needs to react quickly.

The other 73 interlock sources are similarly weighted. Some of these values are given in Table 1.

**Table 1. Some of the interlock sources and their weighted features**

| No | Interlock Source | Occurrence Probability | Risk | Source speed (1/ms) | Editable | | Human health | Knowlage Base |
|---|---|---|---|---|---|---|---|---|
| | | | | | Enable | Treshold | | |
| 1 | HV Spark | 0,8 | 0,2 | 0,002 | 0 | 0 | 0,05 | C |
| 2 | Beam Control volt down | 0,2 | 0,6 | 0,02 | 0 | 0 | 0,1 | B |
| 3 | Voltage divider down | 0,1 | 0,4 | 0,002 | 0 | 0 | 0,1 | C |
| 4 | HVPS down | 0,2 | 0,3 | 0,002 | 0 | 0 | 0,3 | C |
| 5 | Viewscreen wrong position | 0,4 | 0,6 | 0,02 | 0 | 0 | 0,3 | A |
| 6 | BLM | 0,6 | 0,7 | 0,02 | 1 | 1 | 0,3 | A |
| 7 | Aperture temparature too High | 0,8 | 0,3 | 0,02 | 0 | 0 | 0,2 | B |
| 8 | BPM | 0,7 | 0,7 | 0,02 | 0 | 1 | 0,4 | A |
| .. | …… | .. | .. | .. | .. | .. | .. | .. |
| .. | …… | .. | .. | .. | .. | .. | .. | .. |
| 74 | Beamline shutter valve open | 0,4 | 0,2 | 0,1 | 0 | 0 | 0,9 | A |

What is expected from the optimization algorithm is to determine which interlock subsystem will fit into 74 interlock sources with 6 different characteristics. In doing that, it should choose the minimum cost possible and the required speed and maximum security level.

## 2. Material and Method

Solving such a data classification problem, different methods could be used. These are Feature Selection Methods, Probabilistic Methods, DecisionTrees, Rule-Based Methods, Instance-Based Learning, SVM Classifiers, Neural Networks, Genetic Algorithms and etc [2].

Genetic Algorithm is one of the most populer hueristic algorithms for classification problems.

Several other people working in the 1950s and the 1960s developed evolution−inspired algorithms for optimization and machine learning [3].

Genetic algorithms (GAs) were invented by John Holland in the 1960s and were developed by Holland and his students and colleagues at the University of Michigan in the 1960s and the 1970s [3].

The genetic algorithm is an iterative and stochastic process that takes the individuals (populations) in a set into account. Each individual represents a potential solution for the given problem. At the beginning the population is randomly selected. Each individual in the population is given a value as a measure of the goodness by the help of of the fitness function This value is the information the algorithm uses to find the best solution for the given probing [6].

Selection, crossover and mutation operators are used in the genetic algorithm. Replacement is done with the generation of new individuals. This algorithm proceeds with the help of simple operators, creating better generations each time. To determine the better individuals in the population, the fitness value of each individual is calculated. The obtained fitness values are used to rank the individuals in the problem being solved. These values are then combined to form the fitness function. The population of the individuals obtained for the best fitness function is assigned as the solution of the problem [6].

Fitness Function:

The fitness function is used to determine how close the solution to the given problem is. The fitness function in the genetic algorithm varies according to the problem given. Extensive literature is available on the characteristics of the fitness function to expedite the search using GAs [8, 9, 10]. Often, in implementing a GA, some additional measures are taken into consideration. Among various such mechanisms, linear scaling, sigma truncation and power scaling are the most popular [8, 10]. Detailed information about the fitness function used in this study will be given in the following sections.

The operators used in the genetic algorithm can be summarized as follows:

*Selection (Reproduction):*

Reproduction is a process in which individuals are copied according to their fitness function. It means that individuals with the best fit will have a higher contribution to the next generation. When the selection process is performed, firstly the individuals above the median fitness value are collected in the matching pool. Individuals gathered in the pool can be selected using various methods. These methods can be Roulette wheel selection, random selection, rank selection, tournament selection, boltzmann selection, stochastic universal sampling [6].

*Crossover Probability:*

In order to increase the potential of the existing gene pool, the crossover operator is used to generate better-quality sequences than the previous generation. Random values are selected from the new population obtained as a result of reproduction. These values are subject to crossover processing. By using the crossover operator, individuals with higher compliance values can be obtained from the parents.

*Mutation Probability:*

The crossover operator searches for the current gene potential. However, if the population does not contain the information necessary to solve the given problem, a good result can not be obtained. For this reason, an operator with the ability to generate new sequences from the existing gene pool is required. Mutation operator increases diversity by producing new series. Mutation prevents the problem from converging to any local optimum.

*Elitism Probability:*

## 3. Implementation

In this study we used an application of genetic algorithm which K.C. Tan and friends proposed in data mining by evolving a set of comprehensive decision with high classification accuracy [4]. They called it Genetic Programming classifier(GPc) and we will say it TAN Algorithm in this study.

Tan and his friends have extended the idea of boolinezed attributes [11] in their study and have used the Michigan approach to encode the rules, where each individual encodes only one rule. This approach provides a simpler structure, easier development and better comprehension [4]. Along with this, they introduced a new mapping technique to make a more effective fitness evaluation. They also benefited from the covering algorithm, which preserves the rules that are good for developing classification rules and ignores bad rules.

The flowchart of the TAN algorithm is shown in Figure 3. The initial population starts with ramped half and half method [5]. In order to protect the individuals with the best evaluation value in the population and transfer them to the next generation, the elitism which is determined by the user has been applied. Genetic operators have been applied to individuals who are not elite

individuals. Tournament selection method and tournament size 2 were used as the selection method. Small selection of the tournament size provides better convergence by reducing the pressure on the tournament selection [12]. Individuals are subjected to crossover and mutation and then combined with elite individuals from the previous generation to obtain a new population.

In TAN's work, a concept map was introduced to improve the rules. In each GP cycle, a concept table is created over the training set and stored for future reference. This reduces the need to rescan the training set and makes fitness evaluation more efficient [4].

Individuals in the population enter the covering algorithm. The aim of this algorithm is to improve multi-rule structure by supporting diversity. Multiple rules support the same situations in the dataset, allowing for earlier convergence. Often, only a few of these rules are useful and many are unnecessary. The token competation technique was proposed by Wong and Leung (2000) and is used to eliminate unnecessary rules in the rule set. The pseudo code for token competation of TAN Algorithm is given in Figure 4.

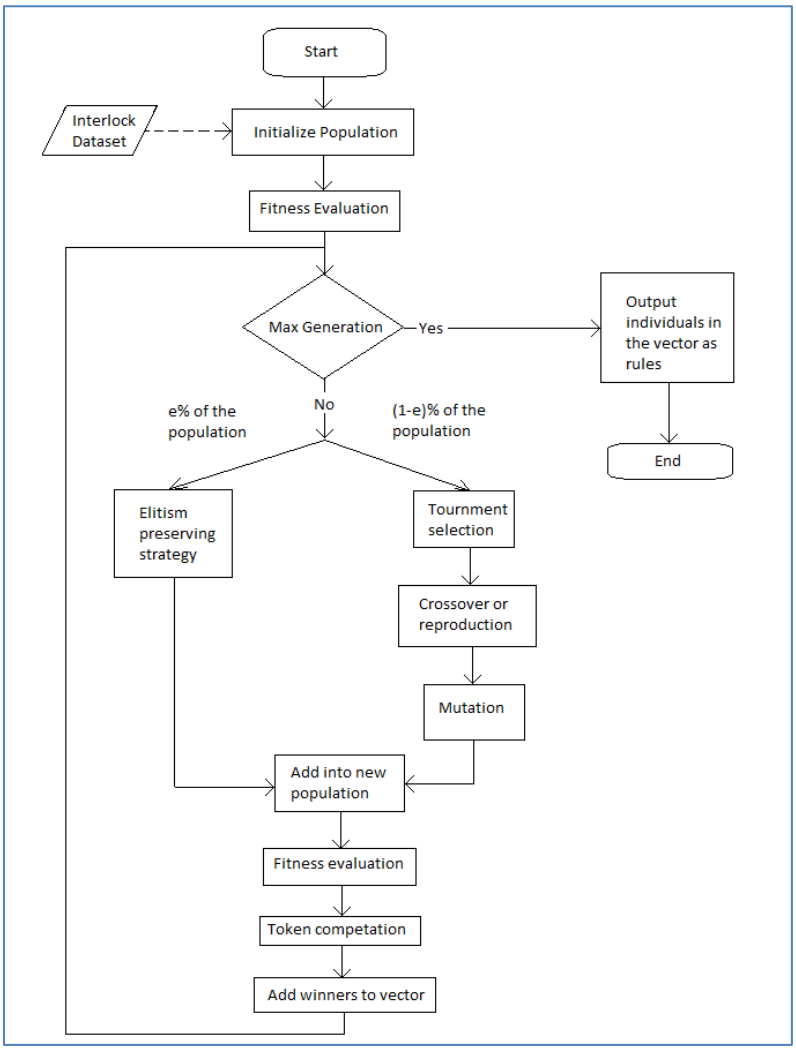



**Fig 3. Flow diagram of the TAN algorithm used**

```
Set the flag of all instances to-1
Set ideal_count and token_count of all individuals to 0
For all individuals in the population and individuals in the rule vector, do
        For all instances in the training set, do
                If instance is in the territory of individual
                        İdeal_count ← ideal_count + 1
                        If this instance haven't been seized by any individual
                                Flag as token
                        Else
                                Challenge the owner of the instance by fitness
                                Flag as token if fitter, otherwise leave it alone
        Repeat for the next instance
Repeat for the next individual
Calculate the token_count of each individual by scanning the flags
Adjust the fitness of all individuals
```

**Fig 4. Pseudo code for Token Competation of TAN Algorithm**

Since the fitness evaluation in the genetic algorithm varies according to the problem given, it is beneficial to open a paranthesis for fitness evaluation at this point.

Fitness function:

The fitness function is used to improve the quality of each rule or individual. The fitness function can be expressed as:

$$fitness = \frac{tp}{tp+\mathrm{w1}fn} x \frac{tn}{tn+\mathrm{w2}fp} \quad (1)$$

True positive (tp) and true negative (tn) are correct classifications while, false positive (fp) and false negative (fn) are incorrect classifications [4]. To explain the terms of the classification in this work:

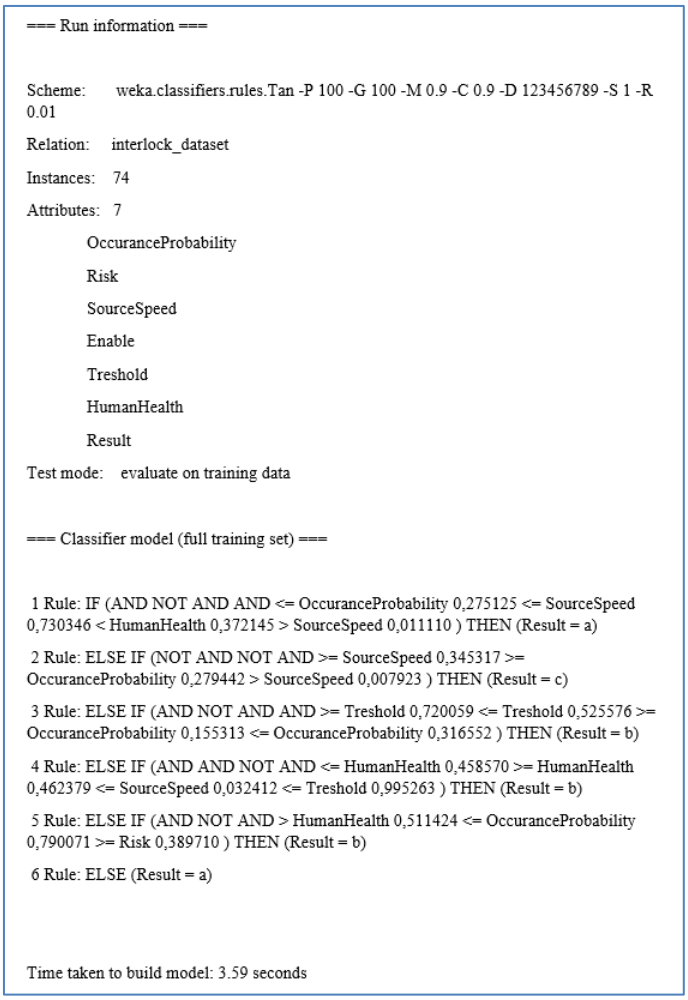
**Table 2. Confusion Matrix**

| | Predicted Class | | |
|---|---|---|---|
| | a | b | |
| Actual Class | tp | fn | a |
| | fp | tn | b |

The w1 and w2 weighting values are used to improve the prediction accuracy in the classifier tests in the Tan algorithm [4]. These values are predefined by the user. As can be seen from Eq. (1), the fitness function has the highest value at low values of w1 and high values of w2. However, excessive reduction or increase of these values may result in various over-fittings such as an increase in the number of rules [4]. Therefore 0,2<w1<1 and 1<w2<20 must be selected [4]. Tan and his colleagues used w1=w2=1 value in their work [4]. The first part of the fitness function is known as sensitivity and the second part is known as specificity [4].

In this study, the TAN algorithm is achieved by running the program in the WAKA (Waikato Environment for Knowledge Analysis) [13] program. The algorithm used is compared with different classification algorithms and the results are evaluated. It is seen that the algorithm used provides the optimum result in terms of reliability and cost within other algorithms.

Firstly, the interlock data set given in Table 1 has been adapted to the WEKA program. Later, the annex of the TAN algorithm was installed in the WEKA program. In the TAN algorithm used, the crossover probability value was chosen as 0.9, the mutation value as 0.9 and the elitism value as 0.1. In addition, interlock data set is used as training data. The outputs of the executed algorithm are given in Figure 5. and Table 3. The applied rules are shown in Figure 5 and detailed accuracy analysis according to the classes is given in Table 3.



```
=== Run information ===

Scheme:       weka.classifiers.rules.Tan -P 100 -G 100 -M 0.9 -C 0.9 -D 123456789 -S 1 -R
0.01
Relation:     interlock_dataset
Instances:    74
Attributes:   7
              OccuranceProbability
              Risk
              SourceSpeed
              Enable
              Treshold
              HumanHealth
              Result
Test mode:    evaluate on training data


=== Classifier model (full training set) ===


 1 Rule: IF (AND NOT AND AND <= OccuranceProbability 0,275125 <= SourceSpeed
0,730346 < HumanHealth 0,372145 > SourceSpeed 0,011110 ) THEN (Result = a)
 2 Rule: ELSE IF (NOT AND NOT AND >= SourceSpeed 0,345317 >=
OccuranceProbability 0,279442 > SourceSpeed 0,007923 ) THEN (Result = c)
 3 Rule: ELSE IF (AND NOT AND AND >= Treshold 0,720059 <= Treshold 0,525576 >=
OccuranceProbability 0,155313 <= OccuranceProbability 0,316552 ) THEN (Result = b)
 4 Rule: ELSE IF (AND AND NOT AND <= HumanHealth 0,458570 >= HumanHealth
0,462379 <= SourceSpeed 0,032412 <= Treshold 0,995263 ) THEN (Result = b)
 5 Rule: ELSE IF (AND NOT AND > HumanHealth 0,511424 <= OccuranceProbability
0,790071 >= Risk 0,389710 ) THEN (Result = b)
 6 Rule: ELSE (Result = a)



Time taken to build model: 3.59 seconds
```

**Fig 5. Output screen for TAN Algorithm to be applied to rule base in WEKA program**

As shown in Figure 5, 6 rules are used for 6 properties. Another thing to note is that the algorithm has arrived in 3.59 seconds as a result.

**Table 3. The values of TAN Algorithm in detail analysis in WEKA program**

| === Detailed Accuracy By Class === | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area |
| Class | | | | | | | | |
| | 1,000 | 0,056 | 0,870 | 1,000 | 0,930 | 0,906 | 0,972 | 0,870 |
| a | | | | | | | | |
| | 0,767 | 0,000 | 1,000 | 0,767 | 0,868 | 0,813 | 0,883 | 0,861 |
| b | | | | | | | | |
| | 1,000 | 0,080 | 0,857 | 1,000 | 0,923 | 0,888 | 0,960 | 0,857 |
| c | | | | | | | | |
| Weighted Avg. | 0,905 | 0,041 | 0,918 | 0,905 | 0,903 | 0,863 | 0,932 | 0,862 |

*Comparison with different algorithms:*

Comparisons were made with 5 different algorithms to determine how well the TAN algorithm used was appropriate for classification in this study. The results of the comparison are given in Table 4.

**Table 4. Comparison table for different algorithms**

| | Rules TAN | Naive Bayes | Meta MultiClass Classifier | Meta Bagging | Meta Attribute Selected Classifier | Bayes net search local K2 |
|---|---|---|---|---|---|---|
| **Correctly Classified Instances** | 67 (90.54%) | 46 (62.162%) | 67 (90.540%) | 66 (89.189%) | 67 (90.540%) | 64 (86.486%) |
| **Kappa statistic** | 0.8582 | 0.4039 | 0.8559 | 0.8388 | 0.8587 | 0.7991 |
| **Mean absolute error** | 0.0631 | 0.283 | 0.136 | 0.1088 | 0.0989 | 0.1631 |
| **Root mean squared error** | 0.2511 | 0.4208 | 0.2267 | 0.2325 | 0.2224 | 0.2702 |
| **Relative absolute error** | 14.381 % | 64.533 % | 31.011 % | 24.816 % | 22.559 % | 37.200 % |
| **Root relative squared error** | 53.644 % | 89.890 % | 48.430 % | 49.658 % | 47.509 % | 57.727 % |
| **Total Number of Instances** | 74 | 74 | 74 | 74 | 74 | 74 |

When the comparision table (Table 4.) is examined, it is seen that 3 algorithms are in the foreground in terms of percent accuracy. One of these is the TAN algorithm used in this study, while the others are the MetaMultiClass and MetaAttrubuteSelected classification algorithms. At this point, confusion matrices need to be examined more closely. Confusion matrices of algorithms used for comparison is given in Table 5.

**Table 5. Confusion matrix comparision table**

| Rules TAN | | | Naive Bayes | | | Meta MultiClass Classifier | | | Meta Bagging | | | Meta Attribute Selected Classifier | | | Bayes net search local K2 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***a*** | ***b*** | ***c*** | ***a*** | ***b*** | ***c*** | ***a*** | ***b*** | ***c*** | ***a*** | ***b*** | ***c*** | ***a*** | ***b*** | ***c*** | ***a*** | ***b*** | ***c*** | |
| 22 | | | | | | 1 | | | | | | | | | 2 | | | |
| 0 | 0 | 0 | 7 | 7 | 6 | 8 | 2 | 0 | 20 | 0 | 0 | 20 | 0 | 0 | 0 | 0 | 0 | **a=a** |
| | 2 | | | 2 | | | 2 | | | 2 | | | 2 | | | 2 | | |
| 3 | 3 | 4 | 0 | 5 | 5 | 2 | 7 | 1 | 6 | 2 | 2 | 6 | 3 | 1 | 6 | 0 | 4 | **b=b** |
| | | 2 | | 1 | 1 | | | 2 | | | 2 | | | 2 | | | 2 | |
| 0 | 0 | 4 | 0 | 0 | 4 | 0 | 2 | 2 | 0 | 0 | 4 | 0 | 0 | 4 | 0 | 0 | 4 | **c=c** |

Twenty-four of the 74 interlock sources used for training were identified as “a” (FPGA subsystem), 30 as “b” (PLC subsystem), and 24 as “c” (software subsystem). What is expected from the algorithms is to capture these values in the result of classification.

In general terms, transitions from “a” to “c” (Classified as Software while it needs to be classified as FPGA), transitions from “a” to “b” (Classified as PLC when it needs to be classified as FPGA) and transitions from “b” to “c” (Classified as software when it needs to be classified as PLC ) is regarded as unacceptable and faulty classification because it reduces safety.

On the other hand, in the opposite case, the transition from “c” to “a” (classified as FPGA when it needs to be classified as software), “c” to “b” transition (classified as PLC when it needs to be classified as software) and transition to “b” to “a” (classified as FPGA when it needs to be classified as PLC) acts to increase security. However, acceptance of such transitions is conditional on the fact that the number of transitions is not too great, as there is an increase in security as well as an increase in cost.

When we examine the confusion matrix of the MetaMultiClassClassifier algorithm, there are two instances where “b” should be “a” (Connected to the PLC while the FPGA needs to be connected). This is unacceptable because it reduces safety. There are two states connected to “a” when it is necessary to connect to “b”. Such a situation would mean an increase in safety and an increase in cost at the same time, but it is acceptable if such a transition is low by the means of number. There are two cases connected to “b” when it is needs to be connected to “c” in a similar way. This also means increased safety and cost, but it is acceptable because it is a small amount. The other case is a transition from “b” to “c”. This transition also reduces safety and is unacceptable because it is connected to the Software when it needs to be connected to the PLC. As a result, this algorithm was not found to be successful due to the fact that it contains two unacceptable situations in terms of security and additionally two conditions that would increase the cost.

When we examine the confusion matrix of the MetaAttributeSelectedClassifier algorithm, there is 1 case where c is needed to be b. This is unacceptable and is considered an error. On the other hand, there are 6 situations in which a must be. Although this is an acceptable value since it is in the direction of increasing safety, it is an imbalance that, as mentioned earlier, it will increase the cost excessively if it goes above a certain amount (The fact that 6 of the 7 faulty classifications are in the direction of increasing the cost causes to depart from the optimum solution).

When we look at the confusion matrix of the TAN algorithm, there are 4 cases where c should be b. This is unacceptable and is taken as the error value. On the other hand, there are 3 cases which are predicted as “a” needs to be “b”. These three cases can be accepted as they are in the direction of increasing safety. When examined in terms of cost, it can be said that the best solution is better approximated by looking at the ratio of these 3 values in the 7 values compared to the MetaAttributeSelectedClassifier algorithm

In this case, it is seen that the TAN algorithm is better than other algorithms in terms of security and cost. Although classification accuracy is given as 90.54% in Table 4, classification accuracy is 94.3% when 3 cases which increase safety are ignored.

The classification graph of the used TAN algorithm is given in Figure 6. Correct classifications and misclassifications are seen together in the graph.

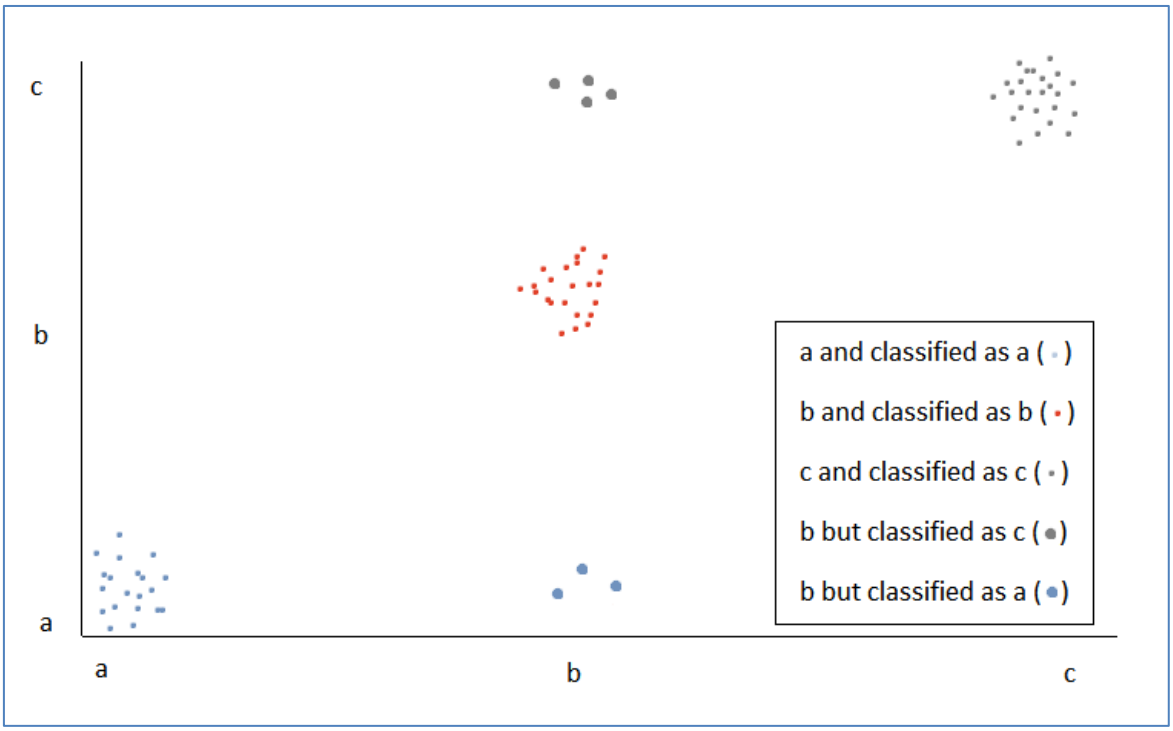



**Fig 6. Classification erros of TAN Algorithm**

Each of the seven misclassified cases can be viewed from the Weka status information screen. The information of three instances which are classified as 'a' needed to be classified as 'b' are shown in Figure 7. In Figure 8. information of 4 instances which are classified as 'c' needed to be classified as 'b' is shown.

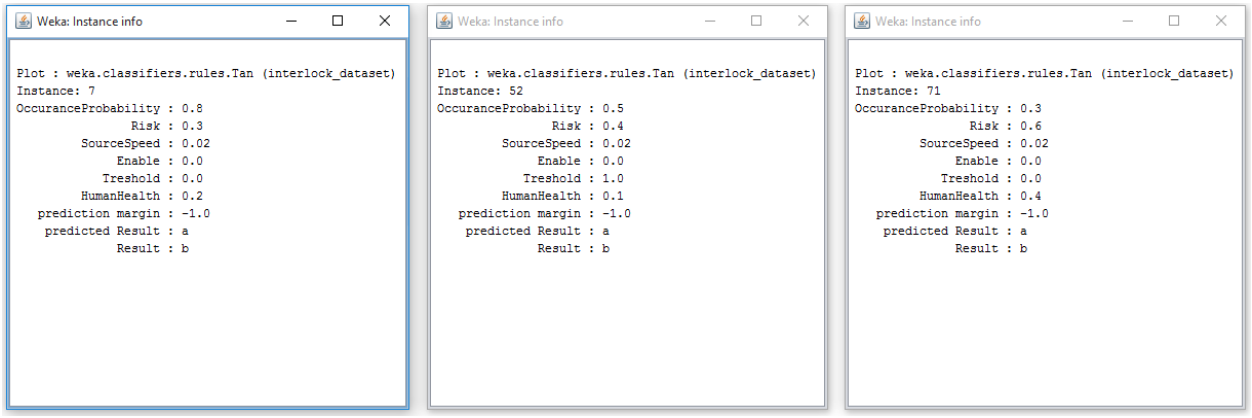



**Fig 7. Classification information for instances that can not be classified correctly (instances classified as 'a' when it should be classified as 'b')**

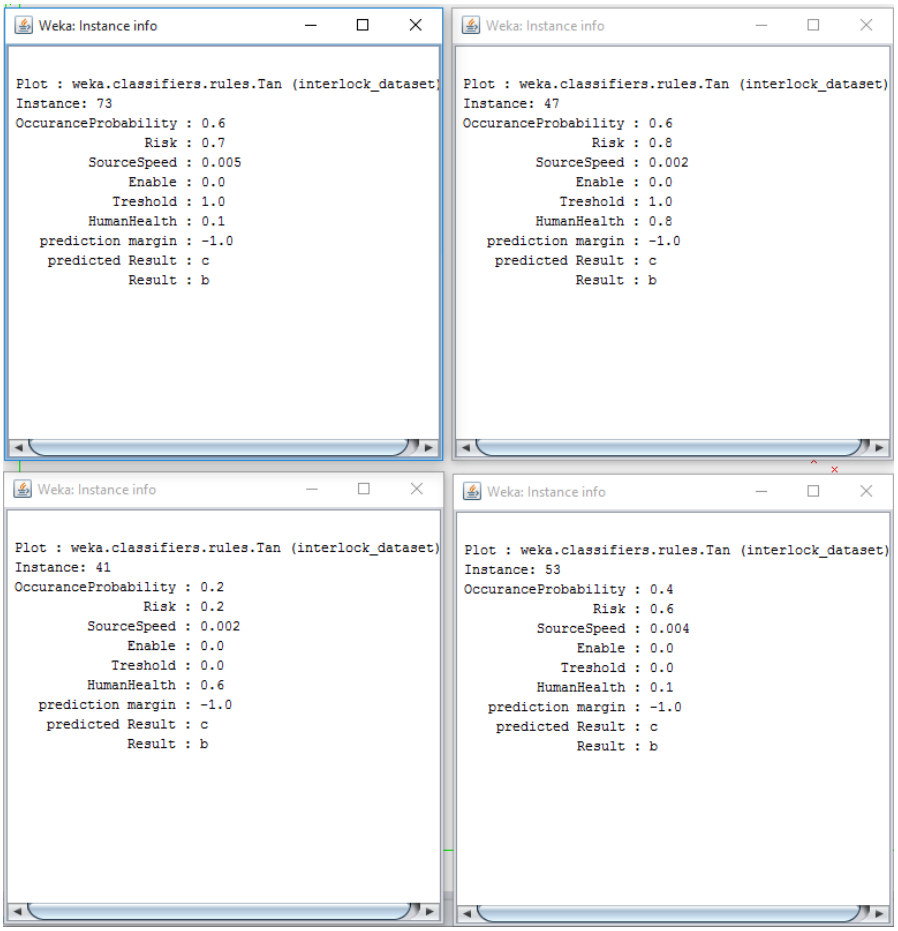



**Fig 8. Classification information for instances that can not be classified correctly (instances classified as 'c' when it should be classified as 'b')**

All tables should be numbered. Every table should have a caption. Headings should be placed above and middle of tables. Only horizontal lines should be used within a table, to distinguish the column headings from the body of the table, and immediate. Tables must be embedded into the text and not supplied separately. Below is an example which the authors may find useful.

## 4. Conclusion

In this work, an interlock system with three subsystems to evaluate the operation and protection modes of the devices used in the accelerator is defined. 74 systems with interlock error output used in the accelerator were identified and 6 different features were determined for these 74 systems. For each system, these 6 features are weighted 0-1 according to importance, and interlock

dataset is created. Through the generated interlock dataset, a genetic algorithm is used to determine which interlock source (system) will enter into which interlock subsystem. As the genetic algorithm, an algorithm developed by TAN and his friends and defined as TAN algorithm is used in this study. This algorithm has run in WEKA program and classification has been made. The interlock dataset was used for the training of the algorithm and as a result, 94.3% success rate was obtained in terms of minimum cost and maximum security. In order to measure the success of the work done, 5 different algorithms were compared. It has been seen that the TAN algorithm used in this study is more successful in evaluating these algorithms based on confusion matrices.

The work done in this paper can be a guide for the cost and security planning of interlock systems for newly installed accelerators. Considering the different needs of each accelerator, the systems to be used can be varied and weighted in different ways according to their safety need. The number of systems can be increased also. Optimized results can be obtained by conducting detailed analysis on the confusion matrix by running the algorithm.

An already installed accelerator can be configured with a minimum cost interlock security system by increasing the number of systems and weighting each system as needed.

It is hoped that this study will also be a resource for those who will work towards the development of new and more succesive algorithms for the given problem.

## References

[1] M. Kago, T. Matsushita, N. Nariyama, C. Saji, R. Tanaka, A. Yamashita, Y. Asano, T. Fukui, T. Itoga, System Design of Accelerator Safety Interlock for the XFEL/SPRING-8, Proceedings of IPAC'10, Kyoto, Japan

[2] Charu C. Aggarwal, Data Mining and Knowledge Discovery Series, CRC Press, 2015

[3] Melanie Mitchell, An Introduction to Genetic Algorithms, MIT Press, 1999

[4] Tan KC, Tay A, Lee TH, Heng CM. Mining multiple comprehensible classification rules using genetic programming. In: IEEE Congress on Evolutionary Computation, Honolulu, HI, 2002. p. 1302–7.

[5] J. R. Koza. Genetic Programming: on the Programming of Computers by Means of Natural Selection. Cambridge, MA: MIT Press, 1992

[6] Sivanandam S. N. Deepa S. N. Introduction to Genetic Algorithms Springer-Verlag, Berlin, Heidelberg, 2008

[7] R. Schmidt Machine Protection and Interlock Systems for Circular Machines—Example for LHC CERN, Geneva, Switzerland arXiv:1608.03087v1 [physics.acc-ph] 10 Aug 2016

[8] Sankar Kumar Pal Classification and learning using genetic algorithms_ applications in bioinformatics and web intelligence-Springer (2007)

[9] D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, New York, 1989.

[10] Z. Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin Heidelberg New York, 1992.

[11] C. C Bojarczuk, H.S. Lopes, and A.A. Freitas. Genetic Programming for Knowledge Discovery in Chest-Pain Diagnosis. IEEE Engineering in Medicine and Biology, July/August, pp. 38-44,2000

[12] Miller, Brad; Goldberg, David (1995). "Genetic Algorithms, Tournament Selection, and the Effects of Noise". Complex Systems. 9: 193–212

[13] I. H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. CA: Morgan Kaufmann Publishers, 1999.