Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Fault isolation by comparing alarm lists using a symbolic sequence matching algorithm

S. Charbonnier*, N. Bouchair, P. Gayet****

** Gipsa-lab UMR 5216 Université Joseph Fourier , Grenoble, France*
Sylvie.Charbonnier@gipsa-lab.grenoble-inp.fr
*** CERN*
Geneva, Switzerland
Nabil.Bouchair@cern.ch, Philippe.Gayet@cern.ch

Abstract: A fault isolation method using the Needleman and Wunsch algorithm (NWA), a symbolic sequence matching algorithm widely used in bio informatics, is proposed. A fault is represented by the chronologically ordered list of alarms raised by the control system on its occurrence. It is transformed in a symbolic sequence where each alarm forms the letter of a very large alphabet. NWA is used to align the k examples of the same fault stored in the data base and then to automatically extract the minimal sequence of alarms typical of the fault, named the fault sequence template. Any new alarm sequence is then compared to the fault templates by means of the similarity measure provided by NWA. The performances are compared with a pattern matching approach using Hamming distance and evaluated on a set of data obtained from a very realistic simulator of a CERN Large Hadron Collider process.

1. INTRODUCTION

To ensure system productivity and safety, modern industrial processes are constantly monitored by operators whose goal is to detect faults before they affect the process under control. To assist the operators, supervision systems, often based on SCADA (Supervisory Control And Data Acquisition) solutions, are implemented. Supervision systems trigger alarms when process variables are out of acceptable ranges. Alarms are just visible symptoms of a fault that may have occurred several hours before, they do not pinpoint the exact cause of the problem. When a fault appears, an alarm is never triggered alone. It is tens or hundreds of alarms that are rapidly raised by the system because of the different parts of the system being interconnected. Alarm avalanches often overload operators, thus slowing down the diagnosis process. It is, therefore, important to provide supervision systems users with hints on the source of the problem to enable them to select the appropriate recovery actions in a short period of time.

Fault isolation, i.e. finding the fault that generated the alarms avalanche, has interested the automatic control community for a long time. However, most of the methods proposed supposed that an in-depth knowledge of the system is available. The knowledge may be summed up by expert rules or by a model of the system. For fault detection, the model represents the normal behaviour of the system. For fault diagnosis, the model represents the behaviour of the system when a specific fault occurs. Various representations may be used to model a discrete event system: finite automata, Petri nets, fault trees, templates, chronicles (Cabasino et al 2011, Hurdle et al 2009, Pandalai and Holloway, 2000). However, the elaboration of a model may be a difficult task when the process is complex and requires involvement of experts. The model is dedicated to a given system and cannot be used on any other one.

Data driven approaches are an alternative to model based approaches. In this case, information on the process is provided by a set of historical data that are used to make decisions. Some solutions consist in learning from the data a model explaining the sequences of alarms observed. Bayesian networks, Hidden Markov Models or Abductive Reasoning Networks or even Petri Nets are popular methods (Lefebvre and Leclercq, 2011, Sun et al, 2012). However, these methods require a large amount of data to learn the statistical relationships represented by the models. These data are not always available, especially data recorded during faults situations that are hopefully relatively rare.

The approach presented in this paper is a data driven one, which can be applied to various systems, as opposed to model-based methods that suffer of a lack of adaptability. It is based on a pattern matching approach, less greedy in data than automatic model elaboration methods. The alarm sequence generated by an unknown fault is compared to a set of sequences of alarms stored in a case base by means of a similarity measure. Each alarm sequence of the case base was recorded on the occurrence of a single fault, which was diagnosed by an expert. The method makes use of a similarity measure from a symbolic sequence matching algorithm. Symbolic sequence matching algorithms are widely used in bio-informatics to compare gene sequences. They were already used to detect anomalous sequences with respect to a database of normal sequences, with application to aircraft security (Budalakoti et al, 2009). In this paper, the Needleman and Wunsch algorithm (Needleman and Wunsch, 1970), named NWA, is at first used to align several alarm sequences generated by the same fault and stored in a case base. The aligned sequences are used to extract a minimal alarm sequence, representative of the fault and named the fault template. Any new alarm sequence is then compared to the fault templates by means of the similarity measure provided by NWA.

The outline of the paper is as follows. In the second section, the mathematical formalism is at first defined, then the creation of the templates using NWA is described and strategies to assist the operator are presented. In the third section, the data set used to validate the method is presented. It is generated by a very accurate simulator of one of the CERN Large Hadron Collider systems, the gas system that provides the particles detector with a mixture of gazes. The tools to evaluate the performances of the method are described in section 4 and the results obtained are presented and discussed in section 5.

2. METHOD

2.1 Problem definition

Let N be the full number of alarms that may be triggered by the control system. Each alarm is represented by a letter of a large alphabet formed of N letters.

The alarm sequence produced when a given fault occurs is represented by a set of n letters from the N letters alphabet, ordered in time : S=<s_1 ,…s_k ,…s_n>. s_k is an alarm that appeared on a time period starting on the occurrence of a first alarm and ending when the alarm flood stops, ie when alarms are no longer triggered by the control system.

Let us suppose that a training set containing f faults is available. For each fault, k alarm sequences recorded during the occurrence of the fault in different running conditions are available. In this paper, we suppose that k, the number of examples per fault, is the same whatever the fault. This is done so as to ease the description of the method. However, the method remains valid if the number of examples per fault is different.

S is an alarm sequence produced by the control system, when an unknown fault occurred. The objective is to determine which fault i triggered the alarm sequence S.

Templates

A fault i does not always produce the same alarm sequence as the physical processes are not deterministic and because environmental conditions may differ when the fault occurs. Fault templates are thus created as a mean to represent a fault with a unique signature. The k examples of fault i are used to create a template of fault i. The k sequences are at first aligned using the Needleman and Wunsch algorithm. Then a minimal sequence is extracted from the aligned sequences.

The Needleman and Wunsch algorithm

The Needleman and Wunsch algorithm (NWA) is an algorithm designed to globally align two symbolic sequences in an optimal way. Given two sequences S^1 and S^2 of lengths n_1 and n_2, the algorithm aligns the sequences by adding gaps in them so that each symbol in one sequence has a corresponding symbol in the other sequence at the same position. Once the two sequences are aligned, their similarity can be calculated by summing the similarities of each corresponding elements. This requires the definition of a score/cost matrix $C(s_i,s_j)$ that gives the score/cost to substitute any symbol s_i by a symbol s_j. Another parameter to tune is the gap penalty, p, which defines how much it costs to insert a gap in a sequence.

For two aligned sequences $S^{1a} =< s_1^{1a},..., s_i^{1a},...s_n^{1a})$, and $S^{2a} =< s_1^{2a},..., s_i^{2a},...s_n^{2a})$ of length n, the similarity is defined as :

$$Sim(S^{1a}, S^{2a}) = \sum_{i=1}^{n} \alpha_i \tag{1}$$

with $\alpha_i = C(s_i^{1a}, s_i^{2a})$ if both s_i^{1a} and s_i^{2a} are symbols or $\alpha_i = p$ if either s_i^{1a} or s_i^{2a} is a gap.

NWA aligns the two sequences S^1 and S^2 so that the similarity of their aligned sequences is maximal. This is done using dynamic programming. The similarity is calculated in a recursive way. A n_1 by n_2 matrix, M, is built in an incremental way, increasing rows one by one and filling each row column by column. Each element M(i,j) measures the similarity between the sub-sequence of S^1, formed of elements s_1^1 to s_i^1 and the sub-sequence of S^2, formed of elements s_1^2 to s_j^2. M(i,j) is calculated from the element a column before, (M(i,j-1)), a row before, M(i-1,j), or a row and a column before, M(i-1,j-1), using the following formula:

$$M(i,j) = \max\begin{bmatrix} \left(M(i-1,j-1) + C(s_i^1, s_j^2)\right), \\ \left(M(i-1,j)+p\right), \left(M(i,j-1)+p\right) \end{bmatrix} \tag{2}$$

To calculate the similarity M(i,j), the algorithm selects the solution that is the less costly out of the 3 possibilities: matching element s_i^1 with s_j^2, adding a gap in S^1 or adding a gap in S^2.

The alignment score of S^1 and S^2 is the value obtained in the lower right hand corner of the matrix M(n_1,n_2). The aligned sequences can be traced back from the matrix M, starting at M(n_1,n_2) and retracing the optimal path up to M(1,1). The optimal path is found by stepping from one cell to the connecting cell whose value is maximal (connecting cells are a row ahead (M(i-1,j)), a column ahead (M (i,j-1)) or a row and a column ahead (M(i-1,j-1))). Moving from (i,j) to (i-1,j-1) means that elements s_i^1 and s_j^2 are aligned, moving from (i,j) to (i-1,j) means that a gap is added in S^1, moving from (i,j) to (i,j-1) means that a gap is added in S^2.

NWA thus provides, from two symbolic sequences, two optimally aligned sequences and their corresponding similarity defined in equation (1). In the following, the similarity from NWA is normalized by dividing its value by the length n of the aligned sequence, to make it insensitive to the length.

Template creation

To create a fault template, k sequences of different lengths must be aligned. Aligning k sequences could be done in an optimal way using NWA but it would require a very long time. Most often, sub-optimal but efficient hierarchical methods are used.

In this paper, an ascendant hierarchical clustering method is used to align several sequences together. C_k^2 combinations of two sequences out of k are first made and their corresponding normalized similarities using NWA are calculated. The two sequences the most similar are selected and clustered in a list of aligned sequences, which forms a node. Any gap added in the two sequences to align them is transformed into a new letter, named X, that is used as a neutral element. The cost to substitute X to any other letter in the alphabet is equal to the cost to substitute the letter with itself. Then, the similarities between the node of aligned sequences and the remaining sequences are calculated. The similarity between a node and a sequence is the highest similarity obtained between the sequence and each of the aligned sequences contained in the node. The similarity between two nodes is the highest similarity obtained by comparing each aligned sequence of one node to another aligned sequence from the other node. The hierarchical clustering is performed by grouping, at each step, the two sequences, the sequence and the node or the two nodes that have the highest similarity until a final node of k aligned sequences is obtained. At each merging step, the sequences are re-aligned in the following way. When merging a sequence and a node, the sequence and the aligned sequence in the node which are the most similar are aligned using NWA. Then, the gaps added in the aligned sequence from the node are added to the other aligned sequences from the node, using the rule “once a gap, always a gap”. Gaps are then transformed into X elements. The same process is applied when two nodes are merged. At the end of the hierarchical clustering, all the k sequences are transformed into k aligned sequences of the same lengths.

The minimal sequence that forms the fault template is extracted from the k aligned sequences by selecting only the alarms that are located at the same place on at least l out of the k sequences.

The method proposed to extract the minimal sequence requires the definition of the cost matrix *C* and the gap penalty *p*, which both tune the NWA and of the minimal number of alarms, l.

The fault templates extracted are formed of the alarms that are the core of the fault. Alarms whose frequency of appearance is smaller than l/k are automatically filtered. The frequently occurring alarms kept in the template are the ones whose order of appearance is the same in most of the examples of the fault. Template extraction is a mean to extract knowledge from the set of faults and may help the operators to better understand the process by showing him/her the alarms typical of the fault.

Decision

Once fault templates are defined, a new candidate alarm sequence *S* is compared to each of the fault template using the NWA algorithm. The candidate sequence is aligned with the fault template and the corresponding NWA normalized similarity is calculated. It is named $\mathrm{Sim}_{\mathrm{NWA}}$ in what follows.

$Sim_{NWA}(S,T_i) = \frac{1}{n}\sum_{j=1}^{n} \alpha_j$ with α_j the similarity of the two j^{th} aligned elements in *S* and T_i.

The fault isolation strategy consists in proposing a list of possible faults to the operator. A possible fault is a fault whose similarity is higher than a detection threshold, Th_d. A unique diagnosis is not proposed but the operator can to pick out a diagnosis from a reduced list of faults, which eases the diagnosis process.

3. PRESENTATION OF THE DATA BASE

The data used to evaluate the method are provided by an accurate simulator of one of the CERN Large Hadron Collider (LHC) systems, the gas system. The LHC, designed and put into operation by CERN (Organisation Européenne pour la Recherche Nucléaire) in 2008, is a large ring of 27 km, built at 100 meters below the earth’s surface, where 2 beams of protons can be accelerated at nearly the speed of light and collided in huge detectors where particles created by the collision can be identified. The LHC is monitored 24 hours a day, seven days a week by several teams of operators. Each team is in charge of monitoring one specific sub-system only, because of the LHC huge size and tremendous complexity. The choice of focusing on gas systems was directed by the fact that it monitored by operators who are non-expert people and therefore need help when a fault occurs.

The simulated process: the gas system

A multi-wire proportional chamber (MWPC) is a particle detector composed of a chamber filled with a mixture of noble gases and electrical wires under voltage (half Anodes and half Cathodes). When a charged particle crosses the chamber, it ionizes the gas separating the atoms into electrons and cations. Electrons are then attracted by anodes and cations by cathodes creating an electrical pulse in wires, the particle trajectory can be reconstructed from the electrical signals generated.

In LHC experiments, gas systems provide precise mixtures of gases to the hundreds of MWPC and regulate the pressure inside those chambers.

A gas system is basically a closed loop where a gas mixture circulates through different modules. A “Primary Supply” module provides pure gases to the gas system; it is composed of monitored bottles. In the Mixer module, up to 3 pure gases can be mixed according to a predefined ratio. In the distribution module, the gas is distributed into several racks (around 10), and then into channels (from 9 to 20 depending on the topology of the detector). The pump module circulates the gas into the loop. The purifier removes impurities from the gas (mostly N2 and O2). The analysis modules periodically analyze the quality of the gas mixture at several points of the gas system. It can stop the system in case of bad mixture proportion. The recovery module recuperates the used gases. It extracts valuable gases in order to reuse then.

The gas system simulator was built using Ecosimpro© software, a continuous and discrete simulation tool. All gas system components and devices such as pumps, pipes, and valves… are modeled with Inputs-Outputs differential and algebraic equations. The model not only describes the physical behavior of each component, but it includes also the model of the control actuators. The complete gas system model is the result of inter-linking all the devices. The complete model is composed of 10 subsystems modeled by 9195 equations.

The model was integrated with the control system. In the simulator, the supervision and control layers are the same as in the real system, only the field layer is replaced by the model. A C++ code of the model has been generated from Ecosimpro©, and linked to the Programmable Logic Controller (PLC) through an OPC server. PLC orders are sent to the model, and calculated sensors values are sent back to the PLC by the model. Hence, an opening order of a valve is sent to the model from the PLC, the simulator receives the order and opens the valve in the model. The model sends back the value of the simulated sensors after calculation. It is therefore possible to operate the simulator with the SCADA as operators would do with the real system.

Faults have been integrated directly into the device models in order to make the breakdown transparent for the control system. A Boolean variable added to the device model can trigger a fault once it is set to 1. This operation can be executed directly by setting this fault variable to 1 from the PLC. A set of 13 faults have been simulated: leaks on bottles (Argon, CO2), leak on Buffer, blockage of the two mass flow controllers in the mixer and the two in the recovery module, stoppage of the pump, leak in pump module, broken bubblers (safety device) in the distribution module and a sensor problem which lead to a regulation issue in the pump module.

The faults were proposed by the experts of the gas system because they are the most common ones. Most of them are quite similar and physically close, which makes the diagnostic complex. To validate the simulator, the faults were simulated and the corresponding event lists generated by the control system were given to the experts to analyze and validate the behavior of the faulty system.

Simulated Data

Each of the 13 faults was simulated 6 times in different operating conditions. This is to obtain different alarm sequences for each simulation, as it is expected on the real system. Therefore, the data base is composed of f=13 different faults, each fault being represented by k=6 alarm sequences. The number N of possible alarms the control system may trigger when faults 1 to 13 occur is 200, which sets the size of the alarm alphabet to 200.

The order of appearance of alarms is the order in which they appear in the alarm list provided by the control system.

4. PERFORMANCES EVALUATION

The results are validated using a 6 fold cross validation method. Each of the 6 subsets is formed with one example of all the 13 faults.

Similarity using Hamming distance

To evaluate the performance of the method proposed, the results obtained with Sim_{NWA} are compared to those obtained when a similarity measure using the Hamming distance is calculated. In this case, the sequence of alarms S is transformed into a vector of 2N rows, V, where each row corresponds to a letter of the alarm alphabet. If the letter appears in the sequence, the corresponding row is set to 1, else it is set to 0. This way, information on alarms appearing or disappearing during the alarm avalanche is kept but the information on the order of appearance is lost. To use the same information as in the sequence templates, a vector template of fault i, Pi, is built using the k examples of fault i. Pi is a vector of 2N rows where a row takes the value 1 if the corresponding letter appeared at least in 1 of the k examples. The Hamming similarity is calculated as follows :

$$Sim_{Ham}(V, P^i) = 1 - \frac{\sum_{j=1}^{2N} \left| v_j - P_j^i \right|}{2N} \quad (3)$$

Detection curves

The performances of the similarities are compared using ROC curves. To do so, the similarities Sim_{NWA} and Sim_{Ham} are calculated for each of the 78x13 pairs of (sequence, template) possible to form from the database. The sequence is selected from the validation set and the template is created using the training set. These pairs may be made of two elements from the same fault - in this case, they are labeled “same”- or they may be made of two elements from two different faults – they are labeled “different”. If the similarity of the pair is higher than a detection threshold Th_d, the pair is detected as “same”, else it is detected as “different”.

For a given value of the detection threshold Th_d, it is possible to calculate:

- the percentage of true detections (TD): the percentage of pairs labeled as “same” and correctly detected as “same” by the detector
- the percentage of false alarms (FA): the percentage of pairs labeled as “different” but detected as “same”.

The ROC curve is built by moving the value of the detection threshold Th_d and calculating, for each of these values, the percentage of true detections and the percentage of false alarms.

Confusion matrix

Classification results are analyzed using confusion matrices. A confusion matrix is a f by f matrix where a case (i,j) represents the number of examples of fault i classified as j by a classifier. When a detection threshold Th_d is fixed on the similarity (strategy 2), several fault classes, named possible faults, may be assigned to a candidate alarm vector. We propose to build the confusion matrix in the following way. Suppose the possible faults list {i, j, l} is assigned to an example of fault i. In this case, one unit is added to cases (i,i), (i,j), (i,l). When all the 78 candidate vectors are classified, the sum of line i divided by the number of examples per fault k, gives the average number of possible faults assigned to fault i. The number of cases in line i different from zero provides the list of possible faults globally assigned to fault i. The value of case (i,i) divided by k gives the percentage of examples of faults i for which the correct class i was present in the list of possible faults.

5. RESULTS

In this part, the score to match any alarm with itself is set to 1, the cost to substitute any alarm with another is set to -1 and the gap penalty *p* is set to 0. This way, it is always cheaper to insert a gap in a sequence than to match two different alarms. No difference is made between alarms. Let us note that the method offers the possibility to assign different substitution costs to the alarms, with the meaning that some alarms, though different, are more similar than others. This possibility is not used in this paper. The results are evaluated in function of the parameter l.

Figure 1 displays the average length of the sequence templates for each fault, in function of l. Because of the 6 fold cross validation method used, the value of l may vary between 1 and 5, since 5 sequences only are used to create the template. The length of the alarm sequences varies from about 15 for faults 1, 2, 3, 4, 10 and 12 up to 60 for faults 5, 6 7, 8, 11 and 100 for fault 9. Increasing the value of l obviously decreases the length of the templates. When l is equal to 1, the length of the templates is higher than the length of the initial sequences. Indeed, any alarm set off at least in one of the k examples of the fault is then added in the template. The length is drastically reduced when l is equal to 5, particularly in faults 5, 6, 7, 8, 9, and 11. This means that, thought the number of alarms set off during these faults is high, only a few are set off each time the fault occurs and in the same order.

The ROC curve obtained with Sim_{NWA} for l equal from 1 to 5 is displayed figure 2. About the same results are obtained when l is equal to 2, 3 and 4, with 90 % of true detections obtained with 5% of false alarms. The performances are decreased with l=1 with 90% of true detection obtained with 10% of false alarms. The worst results are obtained for l=5 with 90% of true detection with 22% of false alarms, showing that relevant information is lost when l is too high.

The ROC curve for Sim_{Ham} is presented figure 3. The influence of l on the performances is similar to Sim_{NWA}. Very poor detection results are obtained with l=1 and 5. The results are improved when l=2, 3 and 4 but they are always worse than the results obtained with Sim_{NWA}. For a false alarm rate of 5%, 78% of the same couples are detected with l=3, 72% with l=2 and 70% with l=4. This is about 15% less than when SimNWA is used. The order of appearance of the alarms is thus useful to discriminate some faults to others.

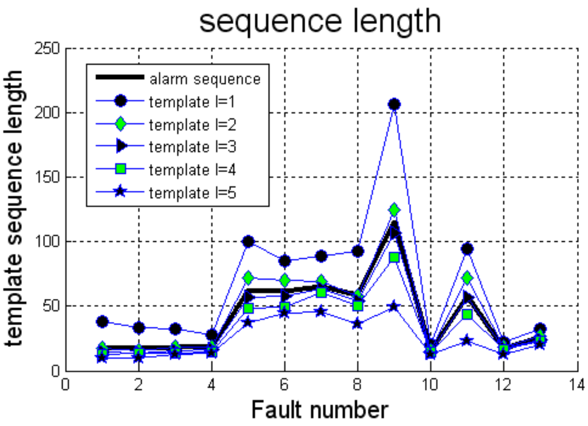

Fig. 1 Alarm sequence length in function of l

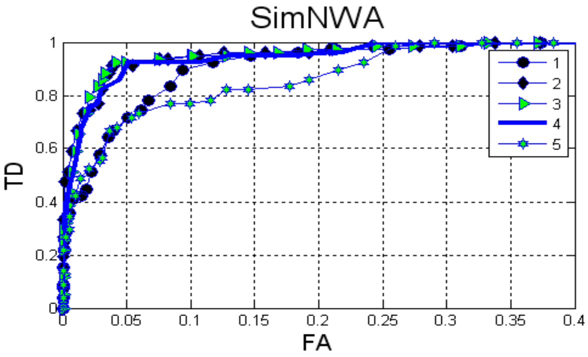

Fig. 2 ROC curve with Sim_{NWA}

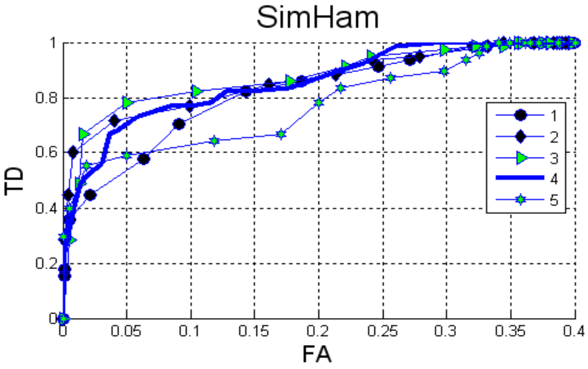

Fig. 3 ROC curve with Sim_{Ham}

The performances of the decision method using a threshold on the similarity are now analyzed. Results are presented for l=3. Let Th_d, the detection threshold, be tuned so that 72 out of the 78 examples are correctly classified (the list of possible faults showed to the operator contains the correct diagnostic in 72 cases out of 78 tests). The value of the threshold Th_d is different for Sim_{NWA} and Sim_{Ham}. The confusion matrices are presented in figure 4, as black and white images. Figure 5 shows the average number of possible faults proposed to the operator for each fault.

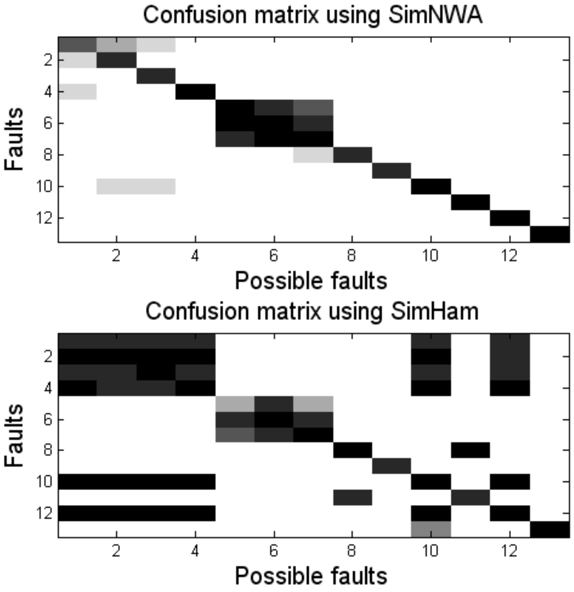

Fig. 4 Confusion matrices (black to white: 6 to 0)

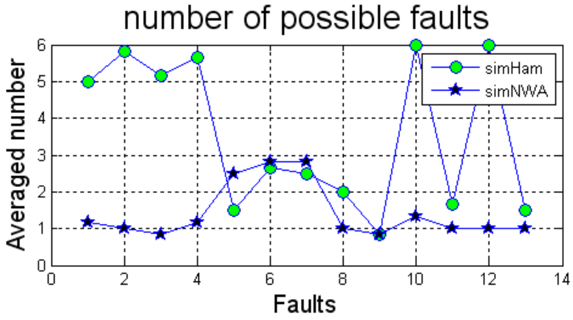

Fig. 5 Average number of possible faults proposed to the operator

It can be seen from figures 4 and 5 that the list of possible fault is drastically reduced with Sim_{NWA}. About one fault is proposed to the operator, except for faults 5, 6 and 7 where 3 proposals are made. When using Sim_{Ham}, faults 1, 2, 3, 4, 10 and 12 are totally confused while they are well discriminated with Sim_{NWA}. The number of alarms set off by those faults is small (about 20) but most of them are shared by all the faults, which makes it difficult to diagnose the faults just by analysing the list of alarms raised. However, the very few alarms specific to each fault change the order of appearance of the alarms in the fault sequence templates, which enables the faults to be diagnosed with $\text{Sim}_{\text{NWA.}}$ Faults 9, 11 and 13 are rather well discriminated with Sim_{Ham}. The list of alarms set off is specific to each fault and using the order of appearance with NWA improves only slightly the already good results. Faults 5 to 7, which generate about 50 alarms, are confused with both Sim_{Ham} and Sim_{NWA}. They are physically and geographically close on the process (problems on the mass flow controllers in the mixer and the recovery modules). Thus, they share the same alarms and their order of appearance is about the same.

6. CONCLUSION

This paper presents a fault isolation strategy using the Needle and Wunschmann algorithm, a symbolic sequence matching algorithm used in bio-informatics. Alarms set off by the control system form a very large alphabet can be seen a words. A pattern matching approach is proposed where each sequence of alarm set off by a fault is compared to a sequence fault template. Results obtained on data obtained from a very realistic simulator of a LHC process shows the interest of the method compared to a pattern matching approach when the order of appearance of alarms is not used. Faults represented by alarms shared by several faults can be easily discriminated.

7. REFERENCES

Budalakoti S., Srivastava A.N., Otey M.E. « Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety" IEEE Trans. On systems, man and cybernetics, part C : applications and review, vol. 39, n°1, 2009

Cabasino M.P., Giua A., Pocci M., Seatzu C."Discrete event diagnosis using labeled Petri nets. An application to manufacturing systems" Control Engineering Practice, Volume 19, Issue 9, September 2011, Pages 989-1001

Hurdle E.E., Bartlett L.M., Andrews J.D. "Fault diagnostics of dynamic system operation using a fault tree based method", Reliability Engineering & System Safety, Volume 94, Issue 9, September 2009, Pages 1371-1380

Lefebvre D. , Leclercq, E. "Stochastic Petri Net Identification for the Fault Detection and Isolation of Discrete Event Systems" , IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans Volume: 41 , Issue: 2 2011 , Page(s): 213 – 225

Needleman S., Wunsch C. A general method applicable to the search for similarities in the amino acid sequence of two proteins, Journal of Molecular BiologyVolume 48, Issue 3, 28 March 1970, Pages 443–453

Pandalai, D.N., Holloway L.E., "Template languages for fault monitoring of timed discrete event processes" IEEE Transactions on Automatic Control, vol. 45, n° 5, 2000, 868 – 882

Sun H., Guo Q., Zhang Y., Zhang B. "An online intelligent alarm-processing system based on abductive reasoning network", IEEE Power and Energy Society General Meeting 2012 , pp. 1 – 5