

# A Data Mining Approach to Incremental Adaptive Functional Diagnosis

Cristiana Bolchini, Elisa Quintarelli, Fabio Salice  
Politecnico di Milano  
Dip. Elettronica, Informazione e Bioingegneria  
Milano - Italy  
{firstname.lastname}@polimi.it

Paolo Garza  
Politecnico di Torino  
Dip. Automatica e Informatica  
Torino - Italy  
{firstname.lastname}@polito.it

**Abstract**—This paper presents a novel approach to functional fault diagnosis adopting data mining to exploit knowledge extracted from the system model. Such knowledge puts into relation test outcomes with components failures, to define an incremental strategy for identifying the candidate faulty component. The diagnosis procedure is built upon a set of sorted, possibly approximate, rules that specify given a (set of) failing test, which is the faulty candidate. The procedure iteratively selects the most promising rules and requests the execution of the corresponding tests, until a component is identified as faulty, or no diagnosis can be performed. The proposed approach aims at limiting the number of tests to be executed in order to reduce the time and cost of diagnosis. Results on a set of examples show that the proposed approach allows for a significant reduction of the number of executed tests (the average improvement ranges from 32% to 88%).

## I. INTRODUCTION

In the past decades, the adoption of artificial intelligence (AI) techniques to support automatic functional diagnosis has received a lot of attention (see [1] for a review). In fact, when dealing with complex boards, functional diagnosis is the only affordable approach, even if several issues need to be dealt with, when defining a relevant approach that allows to identify the most probable faulty (sub)component with a high confidence by performing a small number of tests (to reduce the overall time and effort). In the past, several solutions have been defined, among which we mention *rule-based* ones, that adopt rules stated as “if test output(s)  $\rightarrow$  faulty component” ([2]). Other approaches are based on a model of the system under consideration [3], possibly adding some reasoning-awed knowledge ([4]), requiring a good understanding and information of the relationship between the components and the tests. More recently, reasoning-based solutions have been presented, exploiting Bayesian Networks ([5], [6]), Decision Trees and Support-Vector Machines ([7], [8]). An analysis of the benefits and limitations of the application of different machine learning techniques to test data collection for functional diagnosis is presented in [9], to compare the solutions aimed at limiting the amount of tests being executed, especially if they do not add significant information to either speed up the diagnosis or to increase the accuracy. In general, two are the main issues when trying to improve functional fault diagnosis applied at high abstraction level: i) reduce the number of tests to be

executed to identify the faulty candidate, instead of collecting the complete failure responses and ii) quantify the confidence in the performed diagnosis.

Several statistical learning techniques have been explored in the literature to deal with these two issues, but, to the best of our knowledge, the use of data mining [10], has not been investigated to this purpose. Data mining (DM) research area focuses on studying algorithms and techniques to find interesting patterns representing implicit knowledge stored in massive data repositories; it has been applied to different fields, but is rapidly receiving interest to improve the quality of the manufacturing process. In the present application scenario, we extract knowledge from the system model that puts into relation faulty components and failing tests, in order to infer correlations in the form of *association rules* [11], [12], [10] to guide the diagnosis process.

In this paper we apply DM algorithms for inferring correlations among data in the form of association rules [11], [12], [10]. The correlations are extracted from test vectors to rapidly find out relationships between test vectors and the fault component they actually detect.

The presentation is organised as follows. The next section introduces the basic concepts related to functional diagnosis, that is the adopted system model, partial syndromes and a running example. The proposed approach is discussed in Section III, by referring to a running example. Section IV presents the experimental results achieved by applying the proposed methodology to a set of synthetic examples, validating the effectiveness of the approach. Finally, Section V draws some considerations and highlights on-going and future work.

## II. BACKGROUND

### A. Incremental Functional Diagnosis

An incremental approach to functional diagnosis has been presented in [5], with the aim of reducing the amount of tests being executed to identify the candidate faulty component of a complex system exhibiting erroneous behaviour. The methodology starts from a system model defined for diagnosis purposes, that expresses the relations between the components and the tests being executed. At each incremental step, the methodology selects the test to be executed so that

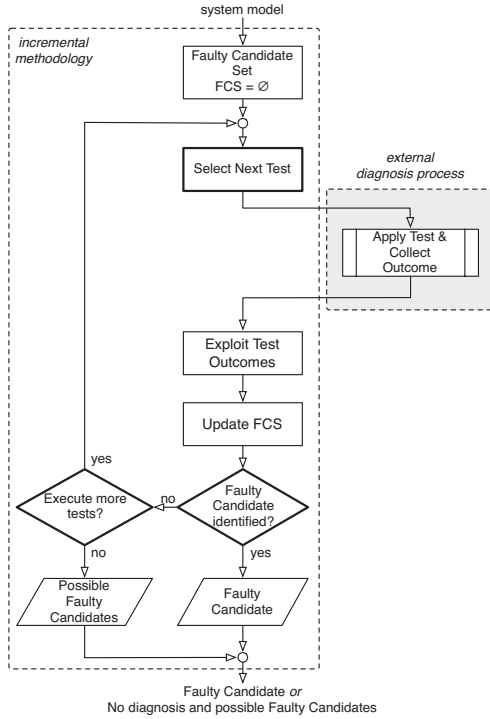


Figure 1. Incremental functional diagnosis methodology

the outcomes would point out the candidate faulty component, without running the complete test sequence (see Fig. 1).

### B. System model

Let us consider a complex system constituted by various components. In general, the system can be a board, where each component is an IP, such as a microprocessor, an accelerator (e.g. an FPGA), the memory, as well as digital circuit described at the RT level, that is an adder, a multiplexer, a block of glue logic. Whatever the case, for each one of these components  $C_i$  a set of test patterns has been defined  $T_j$ ; when applying test  $T_j$  (the corresponding sequence of input vectors) either the expected output (sequence of outputs) is obtained, that is the test PASSES, or not (i. e., the test FAILS). Indeed, in complex systems, for each component  $C_i$  several tests  $T_j, T_k, \dots$  are designed, to exercise the different functionalities of the component, or to target different classes of faults. In general, it is expected that when component  $C_j$  is faulty, test  $T_j$  fails. However, in a complex scenario, the interaction among the numerous components of the system introduces some uncertainty and some controllability/observability issues, such that, the test engineer can only give a qualitative measure of the probability that the test would fail, being the component faulty. Moreover, there are situations when a test designed for a component fails even if the component is fault free, because a different faulty component is on the controllability/observability path. Thus, the model of the system we adopt, is the following one.

Consider system  $S$ , constituted by  $n$  components  $C = \{C_1, C_2, \dots, C_n\}$ , and a suite of tests  $T = \{T_1, T_2, \dots, T_m\}$ ,

		Tests				
		$T_1$	$T_2$	$T_3$	$T_4$	$T_5$
Components	$C_1$	0.9	0.1	0.1	—	—
	$C_2$	0.5	0.1	0.5	0.1	—
	$C_3$	0.1	0.5	0.1	0.9	—
	$C_4$	0.1	0.9	0.9	—	0.5
	$C_5$	—	0.5	0.1	0.5	0.9
	$C_6$	—	0.1	—	—	0.1

Figure 2. Sample CTM used as a running example

such that  $ctm_{ij}$  represents the probability that test  $T_j$  fails when component  $C_i$  is faulty. Rather than using a precise quantitative model derived from the fault coverage offered by tests (often measured in a stand-alone setting rather than in the final complex system environment), we adopt the discrete scale, such that the resulting model is a so-called ([5]) *Components-Tests Matrix* – CTM, where each entry is defined as follows (from [5]):

$$ctm_{ij} \in \{0.9, 0.5, 0.1, 0\} \quad (1)$$

An example of a CTM that will be used throughout the paper to illustrate the approach, is reported in Fig. 2.

### C. Data Mining to model components – test relationships

In this work we use association rule mining to infer correlations between failing tests and faulty components. Association rules [11] describe the co-occurrence of data items and are represented as implications in the  $X \Rightarrow Y$  form, where  $X$  and  $Y$  are two arbitrary sets of data items such that  $X \cap Y = \emptyset$ . For instance, a rule extracted from the running example CTM is  $\{T_4 T_5\} \Rightarrow C_5$ , stating that if both  $T_4$  and  $T_5$  fail, then  $C_5$  is the faulty component.

The *quality* of an association rule is evaluated by means of *support* and *confidence* measures. Support corresponds to the frequency of the set  $X \cup Y$  in the dataset; confidence corresponds to the conditional probability of finding  $Y$  having found  $X$ , and is given by  $\frac{sup(X \cup Y)}{sup(X)}$ . In this paper, the set  $X$  is composed of faulty tests and  $Y$  is the predicted faulty component. For instance, the rule  $\{T_4 T_5\} \Rightarrow C_5$ , with confidence equal to 100%, states that if both  $T_4$  and  $T_5$  fail then the faulty component is  $C_5$  with a 100% estimated probability.

### D. Contributions

This paper aims at exploiting an engine based on Data Mining, to carry out such incremental functional diagnosis process, and in particular the steps i) for the selection of the next test to be executed, and for ii) determining whether to stop or continue the analysis (steps with thicker borders in Fig. 1). More precisely, given the nature of the extracted rules, more than one test at a time can be selected to be executed, thus improving the interaction with the external diagnosis process (especially when methodology and diagnosis are two separately-operated application environments). Furthermore,

the information on support and confidence are used to determine whether the indication of the possible faulty candidate is accurate enough, thus halting the iterative procedure, or if more steps are useful to provide a diagnosis. In the next section, we present the details of the proposed approach.

### III. DIAGNOSIS USING DATA MINING

The proposed approach aims at exploiting a rule-based strategy to guide the incremental diagnosis procedure, in place of the Bayesian Naive Network engine used in [5]. The rationale is the complexity of building the initial CTM model; if the approach is effective, it will be then possible to adopt a different solution, where the rules are directly extracted from log data files, thus either completing, complementing or eventually avoiding the critical modelling step.

#### A. Rule extraction

The first step of the incremental diagnosis methodology consists in mining the rules from the available model. More precisely, from the CTM an initial set of *association rules* is extracted. For the adopted running example, 70 rules are extracted from the CTM.

#### B. Rule weights and ordering

When mining association rules from the the considered CTM, the faulty probability indicated in each cell of the matrix must be considered to improve the precision of the inferred rules. To this purpose, we mine special type of association rules called *weighted association rules* [13] (WARs), which consider also the importance of each item in the analyzed data. In particular, each item in the input data is associated with a weight representing its importance. The assigned weights are used to compute a weighted version of the support measure where the frequency of a rule and the weights of its items are combined. In this paper, we use the *quantitative* relationship between  $C_i$  and  $T_j$  expressed in the CTM to assign an appropriate weight to each item (using the 0.9, 0.5, 0.1 scale).

The weighted association rule mining process is divided into two subtasks:

- 1) find all the sets of items (*itemsets*) whose weighted support exceeds a given threshold *minsup* and
- 2) generate, starting from the mined itemsets, the rules with a confidence greater than a specified threshold *minconf*.

Fig. 3 reports some of the rules mined from the running example CTM in Fig. 2.

Note that for each rule  $\{T_1, \dots, T_n\} \Rightarrow C_j$ , we also compute the average and the variance of the quantitative relationships, obtained from CTM, between  $C_j$  and the tests  $T_i$  mentioned in the rule.

Once the rules have been mined, a ranking procedure is applied to identify the “best” predictive rules. In particular, a sorting order based on confidence, rule length (defined as the number of items – tests – in the antecedent of the considered rule), support, average and variance, is imposed on the mined rule set RS. The highest quality rules are those characterized by a high confidence. We recall that the

#	Rules	Conf.	W. Supp.	Avg. Weight	Var.
1	$\{T_4 T_5\} \Rightarrow C_5$	100%	50%	70%	400%
2	$\{T_1 T_5\} \Rightarrow C_4$	100%	10%	30%	400%
3	$\{T_2 T_4 T_5\} \Rightarrow C_5$	100%	50%	63%	356%
4	$\{T_1 T_2 T_5\} \Rightarrow C_4$	100%	10%	50%	1067%
5	$\{T_1 T_3 T_5\} \Rightarrow C_4$	100%	10%	50%	1067%
6	$\{T_3 T_4 T_5\} \Rightarrow C_5$	100%	10%	50%	1067%
7	$\{T_1 T_2 T_3 T_5\} \Rightarrow C_4$	100%	10%	60%	1100%
8	$\{T_2 T_3 T_4 T_5\} \Rightarrow C_5$	100%	10%	50%	800%
9	$\{T_3 T_5\} \Rightarrow C_4$	83%	50%	70%	400%
10	$\{T_2 T_3 T_5\} \Rightarrow C_4$	83%	50%	77%	356%
11	$\{T_2 T_3\} \Rightarrow C_4$	69%	90%	90%	0%
12	$\{T_1 T_3\} \Rightarrow C_2$	63%	50%	50%	0%
13	$\{T_4\} \Rightarrow C_3$	60%	90%	90%	0%
...	...	...	...	...	...
17	$\{T_1 T_4\} \Rightarrow C_2$	50%	10%	30%	400%
18	$\{T_1 T_4\} \Rightarrow C_3$	50%	10%	50%	1600%
...	...	...	...	...	...
30	$\{T_4\} \Rightarrow C_5$	33	50%	50%	0
...	...	...	...	...	...

Figure 3. Rule set mined and sorted from the sample CTM reported in Fig. 2

confidence value represents and estimate of the conditional probability that given the tests in the antecedent of the rule the faulty component is the one in the consequent of the rule. Hence, we sort rules based on confidence; when two rules have the same value, the shortest one is preferred, to limit the number of test to be performed. In this proposal, we associate a higher rank to shorter rules, because they require a smaller number of tests to be executed, in order to identify the possible faulty candidate. Clearly, a rule with one only test  $T_j$  in the antecedent requires little effort to be verified, however, it provides also little information unless there is a single component  $C_i$  tested with  $T_j$ . In fact, should this situation arise, a rule such as

$$\{T_h\} \Rightarrow C_k \quad (2)$$

with a 100% confidence and weighted support, being the first in the list. Indeed, if the test fails when applied, the component is identified as faulty, but if it passes, no useful information can be re-used for the remaining components. In fact, a system where tests provide good isolation (corresponding to an *identity* CTM) is characterised by rules in the form in Eq. 2, all top ranking, and the diagnosis requires on average  $m/2$  tests to be executed, and in the worst case all  $m$  tests need be executed. In general, as previously mentioned, because the access to a component in a complex system requires interacting with several other components, isolation seldom occurs.

Should the confidence and the length be the same, the weighted support is considered. Finally, the average and variance of weights are taken into account to order rules having the same value for the previously mentioned indicators.

The ranked rule set represents, combined with the CTM, the model exploited by our approach to select the subset of tests to be executed and predict the faulty component as it is described in the following paragraphs.

### C. The incremental and adaptive approach

The incremental method, at each step, selects the most promising test(s) to be executed, and based on the partial syndrome and the exploited association rules it determines whether there is a probable faulty candidate, or additional tests need to be executed. Alg. 1 reports the pseudo-code of the method, whose details are described in the next paragraphs.

---

#### Algorithm 1 Diagnosis

---

```

1: procedure DIAGNOSIS(CTM) ▷ Uses the system model
2:   RS  $\leftarrow$  extractRules(CTM) ▷ Rule Set extracted from CTM
3:   FC  $\leftarrow$   $\emptyset$  ▷ Set of Faulty Candidate(s)
4:   NFC  $\leftarrow$   $\emptyset$  ▷ Set of Not Faulty Candidate(s)
5:   PS  $\leftarrow$   $\emptyset$  ▷ Partial Syndrome - no tests executed
6:   while RS  $\neq$   $\emptyset$  AND FC =  $\emptyset$  do ▷ Nothing else to be done or FC
     identified
7:     EA( $R_i$ )  $\leftarrow$  SelectTopRankingRule(RS)
8:     PSi  $\leftarrow$  ApplyTestForRule( $R_i$ ) ▷ PS after additional test
9:     if  $R_i$  is satisfied then
10:       FC  $\leftarrow$  EA( $R_i$ ).Consequents
11:     else
▷ Propagate test outcomes to rules
12:       RS  $\leftarrow$  applyOutcome(PSi)
13:       NFC  $\leftarrow$  updateNotFaultyCompSet(PSi, CTM) ▷ Review rules w.r.t. NFC
14:       RS  $\leftarrow$  updateRules(NFC)
15:     end if
16:   end while
17:   return FC ▷ Faulty component or no diagnosis
18: end procedure

```

---

**Next test selection:** To select the next test to be performed, we consider the sorted list of mined rules presented above, taking the highest ranking one  $R_i$  (see Alg. 1, line (7)). For such rule we compute the set EA( $R_i$ ) of rules having the same antecedent of  $R_i$ ; EA( $R_i$ ) contains rules that correlate the same set of tests to possibly different faulty components. In the running example, the first rule is  $R_1: \{ T_4 T_5 \} \Rightarrow C_5$  (see Fig. 3). Since, this is the only rule with the antecedent  $\{ T_4 T_5 \}$ , EA( $R_i$ ) contains only  $R_1$  at the first iteration. Tests in the  $R_i$  antecedent are executed, one at a time, and at each outcome the partial syndrome is updated. If all the tests in  $R_i$  fail, then rule  $R_i$  is satisfied and the components in the consequent of the rules in EA( $R_i$ ) are added to the set of Faulty Candidates, FC. On the other hand, if one of the tests passes, then the rule is not satisfied, and the remaining tests in the antecedent are not executed. In particular, before considering the next (set of) rule in the ranked list, the rule set is pruned and a set of (surely) not faulty components is identified based on the partial syndrome.

In our running example, if both  $T_4$  and  $T_5$  FAIL then  $C_5$  is included in the Faulty Candidates set FC. Otherwise, the next rule in the ranking is considered.

Suppose that, after having considered the previous rules in the ranking, we reach rule #17. Since rule #18 has the same antecedent of rule #17 (i.e.  $\{ T_1 T_4 \}$ ), EA( $R_i$ ) contains both rules and considers them simultaneously. Still referring to the running example, if both  $T_1$  and  $T_4$  FAIL, FC =  $\{C_2, C_3\}$ , each one with an associated probability, because both rule #17 and #18 are satisfied.

**Rules pruning:** Once the outcome of one or more tests is available, this *evidence* is propagated to the list of rules, with a two-fold goal: 1) discard rules that cannot be satisfied, because at least one of the tests in the antecedent has passed, and 2) identify those components that can be considered not-faulty, because the only test in the antecedent has passed (rule in the form of Eq. 2).

These pruning procedures allow us avoid the execution of useless tests and limiting the number of components under analysis. The current partial syndrome PS can be used to prune part of the search space. In particular, some components can be excluded from the set of candidate faulty components.

Consider the first rule in Fig. 3. Based on it, the next test to be applied is  $T_4$ , and let us assume that the outcome is  $T_4$ =PASS. Since rule #1 cannot be satisfied, test  $T_5$  is not executed and the entire ranked list of rules is re-evaluated, before proceeding with new tests.

A first propagation of this outcome prunes from the list, all rules having test  $T_4$  in the antecedent, because the rule will never be satisfied. This allows for a reduction in the number of rules to be considered for the subsequent steps. Considering the list in Fig. 3, rule #3 can be removed, as well as rule #6 and so on. Furthermore, we can also exploit the information about the outcome of  $T_4$  to classify as fault-free some components. In fact,  $T_4$ =PASS causes rule #13 to be not satisfied, and *given the values of the weighted support*, we can predict that component  $C_3$  will not be faulty. In fact, the relationship expressed in the CTM assumes that the probability of  $T_4$  failing when  $C_3$  is faulty is high.

Based on this consideration,  $C_3$  is included in the set of not faulty components (see Fig. 1, line (13)), NFC =  $\{C_3\}$ . Moreover, as a consequence, all rules having  $C_3$  as consequent are considered not satisfiable in this current context and are pruned (see Alg. 1, line (14)). Fig. 4 reports the updated list of rules after pruning (only the top ranking ones). Note that, rule #30 is also in the form  $\{T_4\} \Rightarrow C_5$ , however, confidence and support are low, therefore the rule is pruned by the list (because it cannot be satisfied) but component  $C_5$  is not added to the NFC.

The applied pruning allows reducing the number of rules and potentially the number of performed tests. In the running example, the list of rules is reduced to 39 after the first test outcome is exploited.

### D. Faulty candidate identification

As mentioned above, one rule at a time is considered, according to the enforced ranking. As soon as the a rule is satisfied, that is all tests in its antecedent fail, the component in the consequent is identified as the faulty candidate, (Alg. 1, lines (9)-(10)).

In the adopted running example, after the execution of test  $T_4$ =PASS, the next considered rule is  $\{ T_1 T_5 \} \Rightarrow C_4$  (Fig. 4). If both tests fail, component  $C_4$  is inserted in the Faulty Candidate set, that is FC =  $C_4$ ; the partial syndrome associated with this diagnosis is F—PF.



#	Rules	Conf.	W. Supp.	Avg. Weight	Var.
1	$\{ T_1 T_5 \} \Rightarrow C_4$	100%	10%	30%	400%
2	$\{ T_1 T_2 T_5 \} \Rightarrow C_4$	100%	10%	50%	1067%
3	$\{ T_1 T_3 T_5 \} \Rightarrow C_4$	100%	10%	50%	1067%
4	$\{ T_1 T_2 T_3 T_5 \} \Rightarrow C_4$	100%	10%	60%	1100%
5	$\{ T_3 T_5 \} \Rightarrow C_4$	83%	50%	70%	400%
6	$\{ T_2 T_3 T_5 \} \Rightarrow C_4$	83%	50%	77%	356%
7	$\{ T_2 T_3 \} \Rightarrow C_4$	69%	90%	90%	0%
8	$\{ T_1 T_3 \} \Rightarrow C_2$	63%	50%	50%	0%
...	...	...	...	...	...
14	$\{ T_2 \} \Rightarrow C_4$	41%	90%	90%	0%
...	...	...	...	...	...

Figure 4. Updated rule set after the evaluation of  $T_4$  ( $T_4$ =PASS)

In case the top ranking rule is such that there is another rule with the same antecedent, all components in the right part of the rule will be added to the FC set. At this point, two strategies can be adopted:

- 1) consider the diagnosis concluded, presenting all components in the FC set to the user, each one associated with a probability value, computed on the confidence and support values, or
- 2) continue with additional tests, to identify the faulty candidate, if it is possible.

It is worth nothing that, in general, given a system model, it may happen that a complete syndrome does not actually allow for discriminating between two (or more) faulty candidates. In this case, the limitation is caused by the model expressed in the CTM, and not in the approach itself. Indeed, the approach can point out this critical situation to the test engineers, for an improvement either of the model (should it be not accurate) or to the device test solutions.

Strategy 1) constitutes an immediate approach; eventually the user can execute additional tests (this time without the methodology offering support in the choice) and then let the system verify the obtained (partial) syndrome.

Strategy 2) aims at offering an improved confidence in the diagnosis result, exploiting additional tests. However, as discussed in [14] with respect to the Bayesian framework, the identification of a criterion for determining whereas additional tests actually provide useful information is a critical activity, strictly related to the adopted reasoning engine.

In this proposal, we adopt the former strategy, leaving the more refined one for future work.

#### E. Dynamic ranking

The complexity of this approach based on Data Mining resides in the preliminary weighted rule extraction step, performed once, at set-up time, when the system model is provided. Eventually, should mistakes or improvements be introduced, the activity needs to be performed again. The presented approach can thus be dubbed “static”, as it computes the set of rules once, then it updates the list by pruning it, but no further manipulation is performed.

However, when a component  $C_h$  is identified as not faulty, due to a passing test, if we consider the CTM without the

entries related to  $C_h$ , the weighted rule extraction would identify the same set of rules (without all the ones with  $C_h$  in the consequent), *but* with different support values. As a consequence, according to the same policy in the sorting order, rules may be ranked and thus processed in a different order. Differently from the initially extracted rules, the re-extracted rules are focused on the remaining potentially faulty components. Hence, the rules composed of tests able to identify the faulty components among the remaining ones have a better position in the ranking.

This approach has thus been labeled “Dynamic ranking approach”, because the various metrics are re-computed at each step. This update introduces additional complexity in the process, however allows for a further reduction in the number of executed tests.

The next section presents some experimental results achieved by applying the two versions (Static and Dynamic) of the proposed approach to a set of systems, for evaluating the quality of the proposed diagnosis strategy.

## IV. EXPERIMENTAL RESULTS

We performed a set of experiments on 5 synthetic examples (some of them similar to the one in [5]), each one referring to a board constituted by several IP components, and where  $T_j$  represents a set of tests devoted to testing component  $C_i$ . For each example board, a CTM has been defined, constituting the starting point of the methodology.

The analysis focused on the assessment of the efficiency of the methodology, that is the number of tests to be executed in order to identify the faulty component, and the accuracy of the diagnosis, in terms of the correctness of the identified candidate faulty component(s) with respect to the actual one, when the procedure ends by pointing out to the wrong candidate.

We applied both static and dynamic rankings, to evaluate the benefits and costs of the re-computation of rules’ metrics, after executing a test that passes. Results are reported in Table I.

The first part of the table provides information on the board under consideration, in terms of the number of components and tests, and the number of different syndromes that can actually occur. Then, we report the results achieved with the method presented in [5].

The first part of the results, under the “Static Ranking Approach” title, reports results related to the first presented approach, that applies rule pruning after each test outcome, whereas the second part (“Dynamic Ranking Approach”) refers to the improved version of the methodology, adding the re-computation of the rules metric to rules’ pruning, described in Section III-E.

For each experiment in both approaches we computed the minimum number of tests executed to identify the faulty component, the maximum number and the average one, to get an idea of the efficiency, on average, with respect to performing the entire test suite. Moreover, for the “Dynamic Ranking Approach” we also computed the average number of times per syndrome the rules need to be re-evaluated in

Table I  
EXPERIMENTAL RESULTS

Board	# Comp <i>n</i>	# Tests <i>m</i>	# Syn.	AFD approach [5]				Static ranking approach				Dynamic ranking approach				
				# Min Tests	# Max Tests	# Avg. Tests	Acc. %	# Min Tests	# Max Tests	# Avg. Tests	Acc. %	# Min Tests	# Max Tests	# Avg. Tests	#Avg. Rule Eval.	Acc. %
B1	10	14	48	1	10	6.43	100	2	12	7.08	100	2	11	6.08	2.73	100
B2	10	18	576	1	11	2.15	100	1	11	2.10	100	1	10	2.06	1.04	100
B3	6	5	11	2	5	4.00	100	2	5	3.40	100	2	5	3.40	2.50	100
B4	9	9	40	1	9	5.68	100	2	9	5.75	100	2	9	5.90	3.58	100
B5	11	9	35	1	9	5.49	98.6	2	8	5.39	100	2	8	5.67	4.47	100

terms of their confidence/support and related metrics (column “# Avg Rule Eval”).

Results show that the use of the proposed approaches allows for a significant reduction of the number of executed tests with respect to the full syndrome, also improving reduction achieved by [5]. The average improvement with respect to the execution of the entire test suite ranges from 32% to 88%. The improvement is higher for boards B1 and B2, where the number of tests is higher than the number of components. The motivation of this trend could be related to two reasons: (1) for each component there is a set of tests that fail only for that specific component (i.e., the tests are able to discriminate among the available components and the proposed approach is able to select and execute only the subset of needed tests) or (2) some tests are useless.

Columns reporting the maximum number of executed tests (# Max Tests) show that for the first two boards the maximum number of executed tests is lower than the number of available tests, when the number of tests is higher than the number of components, thus allowing for an isolation of the fault.

For all the five boards we achieve also a diagnostic accuracy equal to 100%; the proposed approaches reduce on the average the number of executed tests without impacting on the accuracy of the diagnosis.

A final note refers to the comparison between the Static and Dynamic ranking approaches. On the first board the dynamic approach performs better than the static one in terms of average number of executed tests (6.08 against 7.08), while for the other boards, the two methods achieve comparable results.

All experiments have been executed by means of a C/Java prototype tool, running on a 2.2-GHz AMD Turion Dual-Core RM-75 with 4.0 GBytes of main memory, running Ubuntu 12.04. The rule mining step, that is the most time intensive step, required from a few seconds to at most 30s for all the considered boards.

## V. CONCLUSIONS AND FUTURE WORK

This paper presents a methodology based on data mining for performing an incremental functional diagnosis of complex boards, to limit the number of tests to be executed to successfully identify the faulty candidate. Rules extraction, ranking and exploitation are presented, introducing two different strategies to drive the iterative process. Experimental results of a small set of synthetic examples show that we achieve a reduction in the number of tests ranging from 32% to 88%,

with a 100% accuracy. Based on this preliminary results, there are some possible improvements to be investigated, mainly related to the tuning of the stop condition and the ranking of the rules, such that only the most promising tests with respect to the information they provide are executed.

## ACKNOWLEDGEMENTS

This work is partially supported by the Cisco University Research Program Fund – Gift #2012-101762 (3696), an advised fund of Silicon Valley Community Foundation.

## REFERENCES

- [1] W. G. Fenton, T. M. McGinnity, and L. P. Maguire, “Fault diagnosis of electronic systems using intelligent techniques: a review,” *IEEE Trans. Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 31, no. 3, pp. 269–281, Aug. 2001.
- [2] J. G. Rowland and L. C. Jain, “Knowledge based systems for instrumentation diagnosis,” *Engineering Applications of Artificial Intelligence*, vol. 6, no. 5, pp. 437–446, 1993.
- [3] G. Friedrich, M. Stumptner, and F. Wotawa, “Model-based diagnosis of hardware designs,” *Artificial Intelligence*, vol. 111, no. 1-2, pp. 3–39, 1999.
- [4] C. O’Farrill, M. Moakil-Chbany, and B. Eklow, “Optimized reasoning-based diagnosis for non-random, board-level, production defects,” in *Proc. IEEE Int. Test Conference*, 2005.
- [5] L. Amati, C. Bolchini, L. Frigerio, F. Salice, B. Eklow, A. Suvatine, E. Brambilla, F. Franzoso, and M. Martin, “An incremental approach to functional diagnosis,” in *Proc. Int. Symp. Defect and Fault Tolerance in VLSI Systems*, 2009, pp. 392–400.
- [6] Z. Zhang, Z. Wang, X. Gu, and K. Chakrabarty, “Board-level fault diagnosis using bayesian inference,” in *Proc. VLSI Test Symposium*, 2010, pp. 244–249.
- [7] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, “Adaptive board-level functional fault diagnosis using decision trees,” in *Proc. Asian Test Symposium*, 2012, pp. 202–207.
- [8] —, “Board-level functional fault diagnosis using learning based on incremental support-vector machines,” in *Proc. Asian Test Symposium*, 2012, pp. 208–213.
- [9] H. Wang, O. Poku, X. Yu, S. Liu, I. Komara, and R. D. Blanton, “Test-data volume optimization for diagnosis,” in *Proc. Design Automation Conference*, 2012, pp. 567–572.
- [10] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd edition. Morgan Kaufmann Publisher, 2006.
- [11] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules in large databases,” in *Proc. Int. Conf. Very Large Data Bases*, 1994, pp. 487–499.
- [12] E. Baralis, L. Cagliero, T. Cerquitelli, and P. Garza, “Generalized association rule mining with constraints,” *Information Sciences*, vol. 194, pp. 68–84, 2012.
- [13] Wei Wang, Jiong Yang, and Philip S. Yu, “Efficient mining of weighted association rules (war),” in *Proc. Int. Conf. Knowledge discovery and data mining*. ACM, 2000, pp. 270–274.
- [14] L. Amati, C. Bolchini, F. Salice, and F. Franzoso, “A formal condition to stop an incremental automatic functional diagnosis,” in *Proc. Euromicro Conference on Digital System Design, Architectures, Methods and Tools*, 2010, pp. 637–643.