Politecnico di Milano
Dipartimento di Elettronica e Informazione

DOTTORATO DI RICERCA IN INGEGNERIA
DELL'INFORMAZIONE

Test and Diagnosis Strategies for Digital Devices: Methodologies and Tools

Doctoral Dissertation of:
Luca Amati

Advisor:
Prof. Fabio Salice

Tutor:
Prof. Donatella Sciuto

Supervisor of the Doctoral Program:
Prof. Carlo Fiorini

2011 - XXIV

Abstract

Given the growing complexity of electronic devices and systems, diagnosis is becoming a complex task concerning both the fault modeling and the computational effort of algorithms for automatic fault identification. The challenge for developing an effective methodology impacts the efficiency of the manufacturing process, and this is particularly true for the digital system field: the sooner a failure root cause is correctly understood, the more important is the reduction of diagnosis time, and the higher is the yield can be achieved.

In this thesis we present a methodology, called incremental Automatic Functional Fault Detective (AF2D), aiming at the general problem of system diagnosis. We propose a framework where a system under inspection is described at an high level of abrastraction, using Bayesian Belief Network (BBN) formalism. The adoption of a model at coarse granularity allows the description of complex, deeply interconnected systems. At the same time, AF2D provides both system and test engineers with modeling primitives to describe of relationships between system components, potential candidates for fault localization, and outcomes of diagnostic tests.

BBN probabilities are set with qualitative labels (high, medium, low) and not as quantitative values: this choice simplifies and accelerates the system and fault modeling process. Underneath, provided with the outcomes of executed diagnosis tests (syndrome), an inference engine computes the probability of each candidate to be the cause root of the observable symptoms.

This work covers three main directions of research for the application of the AF2D diagnostic methodology: the initial scouting of a system for fault detection, the exploration of the solution space for a fast identification of the failure cause, and the quantitative analysis of robustness for the generated models with respect to diagnostic precision and fault isolation resolution.

Concerning the first question, cost-effective policies to identify a good subset of tests providing good coverage of the system under inspection are proposed. Applying standard optimization techniques (Integer Linear Programming (ILP)) on the BBN model, a subset of tests within a complete test suite is selected, aiming at minimizing the effort required for fault scouting. The sets of tests obtained with this method are compared with other test sets, calculated with more fine-grained modeling on the same system, in order to justify the validity of adopting model at an higher level of abstraction. Also, an hill-climbing technique is proposed for sorting the initially executed tests, in order to further reduce the time

spent on detection and, consequently, to increase the amount of time available for the real diagnosis process. Efficiency improvement is expected in manufacturing lines since the probability of some specific failures with respect to others happens to be specific in different timing window of production.

A considerable effort is devoted to the adaptive part of the methodology, targeting the minimization of costs of each tests session without recurring to a static test sequencing approach. We propose a geometrical interpretation of BBN parameters and a quantitative evaluation through a metric distance within a vector space. This is used for an optimal step-by-step selection of tests to be executed, exploiting the information contained in the diagnosis history, represented by test outcomes collected during the diagnosis of the system. Optimal selection aims at maximizing the relative information that a non-executed-yet test outcome would apport to diagnostic conclusions, reducing the amount of redundancy with respect to previously executed tests, while minimizing the cost of executing of such test itself. A metric for the identification of a stop condition is also proposed, to interrupt the diagnosis when no further information from remaining tests would refine the diagnostic conclusion any longer.

The last part of this work investigates the correctness of the model based on BBN used for system diagnosis. Such validation is a complex task, and a statistical oriented approach is proposed for developing a quantitative comparison of the BBN model of a system with a fine-grained model counterpart, based on a well established and mature methodology. In particular, we target a stuck-at fault model on combinational circuits: a BBN-compatible model is extracted for benchmark circuits used for fault detection and diagnosis problems, and the results of AF2D methodology are compared with the results provided by an ATPG tool. The correlation found support the claims that high-level models can be adopted used for diagnosis and that the detailed information about the internal behavior of the system is not critical for the obtention of valid system diagnosis. Furthermore, we suggest an method to improve the diagnostic resolution of existing test suites, with a minimal modification both of number of tests (impacting diagnosis time) and of their specific coverage of components (impacting test development effort).

Both simulated synthetic systems and industrial case studies have been used to validate the robustness and the efficiency of the proposed methodology.

Contents

Introduction 1

Troubleshooting is described in [HMDPJ08] as the explanation or interpretation of initial failure symptom, where a sequence of diagnostic tests is selected efficiently to locate the root causes of failures. Any *diagnosis* strategy has to tackle some fundamental issues [SS91b] :

- **Detection**, as the capability of a combination of tests to identify the presence of a failure in a system.
- **Localization**, as the capability of a test to restrict a fault to a subset of pre-identified possible causes.
- **Isolation**, as the capability of a diagnostic strategy to restrict localization in order to allow the repair of a single unit (at maintenance level).

Diagnosis and unexpected faulty states in large and complex systems, usually based on the interaction of several heterogeneous components, is performed making *inferences* and conclusions from results of various tests, measurements, observations of the parameters of the system under investigation. Besides digital devices, diagnosis methodologies are designed to operate also in different fields, including medical diagnosis, airplane and automotive failure isolation, but also specific domains are error-correcting coding or speech recognition [Mac03]. Although methodologies are rather generic, as they find application in a wide variety of problem areas, we focus in their specific implementation on the area of digital systems management.

Greater system complexity, lower production costs, adoption of new technologies and shorter life-cycles have made the need for automatic tools with diagnostic purposes for electronic systems important [FMM01]. In an ideal scenario, failed products are diagnosed as a whole in a testing session during manufacturing, and field returns diagnosed and repaired in a cost effective manner.

However, as components, boards and systems become more and more complex, troubleshooting cost increases exponentially. Then test policies aiming at detecting and diagnosing failures early (locally) on in the test process (*component* or *structural* tests) [ME02], are more effective and therefore more accepted in industry; inevitably specific defects might escape those stages and cannot be detected until the system is completely integrated. The identification and characterization of these defects usually requires an *expert* (engineering skills), which depending on the specific architecture of the system might take long time to develop. And during the initial product stage, such expertise is needed but often unavailable. Therefore, tasks as fault diagnosis - detecting system problems and isolating their root causes - are increasingly important but at the same time an intrinsically difficult task.

Design of efficient diagnosis techniques plays an important role from an *economic* perspective, especially to improve product yield and accelerate time-to-market for manufacturing [Tur97]. Furthermore, diagnosis techniques are required to describe systems at *level of abstraction* which is more and more high, as they are required to deal with complexity both in terms of systems architecture and interactions. *Divide-et-impera* strategies, focusing at detecting faults independently on specific parts of the system, are deemed to fail in specific fields, where system hierarchy is deep and complex and there is an elevated level of integrations of heterogeneous cooperating elements (e.g., digital devices manufacturing [ZWGC10a]).

A good definition of the procedure of fault diagnosis is present in [FMM05] as the *isolation of a fault* is a system, obtained from analysis of the information collected during system observation and tests. Authors describe the diagnosis process as a three-step procedure:

1. generation of *system information*, the collection and analysis of the system parameters (through observable symptoms, measurements, diagnostic tests);

2. generation of *fault hypothesis*, the analysis of all potential locations

of failures, and the identification of *system information* which is consistent with each one of such failures;

3. *hypothesis discrimination*, taking place when multiple *fault hypothesis* are consistent with the system information; this step consists in the application of further testing, the comparison with previous diagnosis (historical), expertise or trial and error.

A capital achievement of any diagnosis procedure is *high accuracy* for localization and isolation, excluding all non-consistent failure explanations and focusing on valid possible failure cause(s). This achievement requires the knowledge of the outcome of a large number of tests, in the extreme case the execution of the whole set (*test suite*) of available tests. This might be an expensive cost so that even a methodology, proven to be valid from failure localization perspective, results to be unaffordable from an industrial perspective. An essential key-point for the development of new solutions for automatic fault diagnosis is the improvement provided in terms of scalability and cost-efficiency, by using only the most relevant measurements at any time point, i.e., by exploiting an adaptive (*context-* or *instance-specific*) inference policy to the current system state and observations.

1.1 Research and document organization

This Section presents the organization of the different contributions to fault diagnosis, throughout the Chapter of this document. We introduce in Section 1.3 an overview of the recent advancements in the field of fault diagnosis, after we provide a common background (Sections 1.2) to present and compare different methodologies, derived from digital device diagnosis area but also from other research sectors. The introduction of the different works retrieved in literature is connected to our research underlining the contributions to specific aspects of the diagnosis process (e.g., information modeling, adaptive test selection, …).

In Chapter 2, we present our incremental Automatic Functional Fault Detective (AF2D) reference framework. The methodology, based on Bayesian Belief Networks (BBNs), aims at providing test engineers with a unified procedure to describe the system under inspection with an high-level abstraction model, with the definition of *primitives* (nodes connections, coverage coefficients) to describe even complex relations between the system constituting elements and diagnostic test results (outcomes). The framework is provided also with an efficient inference

engine to evaluate diagnosis from a partial or complete set of outcomes retrieved from an instance of a system under diagnosis.

In the following Chapters we develop three main aspects of the methodology, namely the *fault scouting phase*, the *optimal test sequencing* and the *model robustness* analysis. In Chapter 3, we tackle the identification of a metric for evaluating the cost of a test session. Given the cost model, test suites are evaluated with respect to their capability of *detecting* the manifestation of a fault within a system, exploiting the abstract model information only. Two main problems are analyzed and a solution is proposed: the stimulation of all elements of system (to avoid misdetection) in a cost effective policy, and the correct execution order of a test sequence in order to minimize cost stimulating the most frequent failure causes first.

Chapter 4 is devoted to the adaptive part of the methodology, aiming at the cost minimization of each test section thought the incremental test execution policy of AF2D. In particular, the complexity of the direct analysis of the BBN model is overcome through a geometrical interpretation of its parameters, making it possible a quantitative evaluation of each test session using distance metrics within a *vector space*. The problem of test selection and test session completion (stop condition) are formulated in such a context and proven to be efficient; later a validation is proposed on synthetic and industrial case studies.

Chapter 5 targets problems related to the robustness of diagnostic information obtained with a system model, designed at an high level of abstraction, dealing with complex and highly inter-correlated systems as modern digital devices. In particular, an analysis of the *accuracy* of the diagnostic conclusions obtained with the BBN model is carried out, and its statistical validation is proposed through comparison against highly detailed fault models; therefore, it is indicated the potential benefit of adopting AF2D where the detailed knowledge of the system is not available, or it could be obtained at unaffordable cost of time and resources. Accuracy is also tackled from the test suite designer point of view, and an approach is developed to evaluate quantitatively the quality of a set of tests for the diagnosis of the system, while underline is weaknesses from the failure discrimination perspective, and to propose an incremental test suite extension to improve its diagnosis capability at a minimum cost.

Chapter 6, eventually, summarizes the contribution of the research activities and it proposes an outlook of future extensions an research directions.

Contributions Some publications have been produced during the development of the research activity.

- Luca Amati, Cristiana Bolchini, Fabio Salice, F. Franzoso, and al. A incremental approach for functional diagnosis. In *IEEE Intl. Symp. Defect and Fault Tolerance of VLSI Systems.*, pages 392–400, 2009

In this paper, we introduced the AF2D methodology, providing the theoretical basis of the BBN modeling and an initial proposal for the *incremental* (adaptive) exploration of tests.

- Luca Amati, Cristiana Bolchini, and Fabio Salice. Optimal test-set selection for fault diagnosis improvement. In *Proc. IEEE Intl Symp on Defect and Fault Tolerance in VLSI Systems, DFT*, 2011

This paper provides an ILP-based optimization approach to identify an efficient test set, targeting a fast scouting of a failure in a system. The test set is retrieved by the high-level abstraction model but its performance are compared with the results of a more detailed model (of the same system).

- Luca Amati. Optimal Test-Suite Sequencing for Bayesian-Network Based Diagnosis. Technical Report 2011.3, Politecnico di Milano, 2011

This contribution tackles an optimal *sequencing* of the test suite. aiming at the minimization of the time to first fail. The method exploits the information of the BBN model and explores the solution space using an hill-climbing approach.

- Luca Amati, Cristiana Bolchini, and Fabio Salice. Test selection policies for faster incremental fault detection. In *Proc. IEEE Intl Symp on Defect and Fault Tolerance in VLSI Systems, DFT*, pages 310–318, 2010

In this paper, we introduce a geometrical interpretation of the BBN evolution in order to establish and evaluate a quantitative metric to lead the adaptive selection of the next test.

- Luca Amati, Cristiana Bolchini, Fabio Salice, and Federico Franzoso. A formal condition to stop an incremental automatic functional diagnosis. In *Proc. 13th EUROMICRO Conf. on Digital System Design - Architectures, Methods and Tools*, pages 637–643, 2010

This paper defines a stop condition for the incremental approach of AF2D, within the same geometrical interpretation.

- Luca Amati, Cristiana Bolchini, Fabio Salice, and F. Franzoso. Improving fault diagnosis accuracy by automatic test set modification. In *Proc. IEEE Int. Test Conference*, 2010

In this paper, we tackle the definition of a metric for evaluating the diagnosis accuracy provided by a given test suite. This is based on a distance metric based on the geometrical framework for the BBN analysis, and it is used to create an incremental approach for the *extension* of a diagnosis test suite for a system under analysis.

- Luca Amati. Parameters Sensitivity Analysis in Bayesian Network-Based Diagnosis. Technical Report 2011.3, Politecnico di Milano, 2011

This contribution is devoted to the sensitivity analysis of the BBN model of a system, obtained from a correlation analysis of the diagnostic conclusions of an alternative model of the same systems, containing an exahustive (*golden*) description of the fault-test relations. A validation is produced against combinational circuits, using the stuck-at fault model and the test suite obtained from an ATPG tool.

1.2 Systems

In this section we present some simple definitions to introduce the problem of failure diagnosis. Such definitions will be useful also for the analysis of the previous works, where terminology is quite inhomogeneous because of the different fields where methodologies have been conceived and implemented. This will provide both a uniform overview of the literature and a unique context to locate our contributions.

Definition 1. We define a *system* $\mathcal{S}$ as a heterogeneous collection of cooperating entities, designed to realize a specified group of *functionalities*.

While Definition 1 is quite generalist, it can be easily adopted to cover a set of different types of systems that can be encountered in the fields of engineering: from digital combinational systems to computer networks, to nuclear or chemical plants.

Definition 2. We define a fault $\mathbf{f}$ as a source of *misbehavior* of a system. The presence of a fault puts the system in a state where the execution of one or more of its functionalities is partially or completely different from the *expected* execution.

We denote also with $\mathcal{F}_{\mathcal{S}} = \{\mathbf{f}_1, \ldots, \mathbf{f}_n\}$ the set of all faults potentially affecting system $\mathcal{S}$.

In reliability literature [Ise06], there is a clear distinction among *defects*, which correspond to locations of a system instance containing a difference between specification design and implementation; *faults*, defined as the actual causes of a system misbehavior; and *failures*, representing the external observable effects of a fault, producing a misalignment of a functionality from its specifications. In some occasions, we will introduce an ambiguity using the terms *faults* and *failures*, where the difference of definitions is not essential.

The definition of a *diagnosis process* requires some additional definitions, covering both the target of the process (components) and the information required to be collected to execute it (tests).

Definition 3. A component $\mathbf{c}$ represents a *location* of a system under analysis which can be affected by a fault.

We denote also with $\mathcal{C}_{\mathcal{S}} = \{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$ the set of all faults potentially affecting system $\mathcal{S}$.

We refer to a component containing (at least) one fault as a *faulty component*, and we use *fault-free component* definition otherwise.

Definition 3 requires that the location is to be identified in *unequivocally* within the system, in a such a way that a "*fault* **f** *is found in component* **c**" is a consistent statement of logic. As a corollary, we extend Definition 2 requiring that a fault can affect only one component, and two components of a system cannot overlap. This is equivalent to consider that a system is the *set of possible faults* $\mathcal{F}_{\mathcal{S}}$ and components are a *partition* of such a set.

According to the proposed definition, a component could correspond to a physical device of the system, as for instance a memory chip in a digital system. However, this condition is not strictly required: a component could be also represent a group of physically connected elements within the system, or even a set of non-interconnected entities contained in a system (*virtual component*). Indeed, it is the capability of a system expert to describe and isolate the location of a fault to determine the subdivision of a system in components; this level of isolation corresponds also to the *maximum accuracy* (or level of detail) that could be attained by any fault localization algorithm, since no further subdivision can be obtained.

The probability of a component to contain a fault is an important concept.

Definition 4. We define the *a-priori (component) fault probability* $\mathbf{P}(\mathbf{c})$ as the probability that component **c** contains (at least) one fault, given no other information but the decomposition of the system in components.

This definition makes the *a-priori* probability value intrinsically related to the particular decomposition adopted to describe the system. Depending on the context, the interpretation of this value can be twofold.

Dealing with scenarios where both fault-free and faulty systems are present, the *a-priori* probability represents the *absolute* probability to discover a fault in the component of the system. This assumption is general, and a good estimation of such probabilities could be obtained considering, for instance, the *failure rates* of the devices used to implement the system under inspection [ME02].

Otherwise, the analysis could target a system which have been proven to contain at least one fault, for instance, a device which have been taken out of the production line [VWE+06] after failing some verification check. In this case, it is necessary to take into account the information of such

a fault to be present. Then, the *a-priori* probability of a component represents the *relative* probability the cause of the failure is located exactly in that component.

Different distributions can be used to extract the values for *a-priori* probabilities for all components. When no other assumption is done, the normalization of the failure rates of the devices contained in the system can be used. On the other hand, if a statistically significant number of past cases is available, it is possible to take into account this information as a correction of the original probability distribution [BS07b].

A common assumption adopted in diagnosis is the so-called *single fault hypothesis*, in order to reduce the complexity of considering an exponential number of fault combinations to describe all system failures. According to this assumption, any instance of the system can contain at most one *faulty component*. This particular scenario is described in terms of *a-priori* probabilities imposing a *normalization* of the values, to sum to the unity.

$$\sum_{\mathbf{c}_i \in \mathcal{C}_\mathcal{S}} \mathbf{P}(\mathbf{c}_i) = 1 \tag{1.1}$$

Typically, at design time, a system is defined following a hierarchical decomposition. Several independent and interoperating elements are developed to cooperate in order to implement the required functionalities while keeping design, development and maintenance simple enough.

Definition 5. A subcomponent **sc** represents a *location* within a component **c** of a system $\mathcal{S}$ which can be affected by a fault.

It is worth noticing that the relation between components and subcomponents is similar to the relation between system and components, since it introduces a *partition* of the faults set of each component. This definition is depicted in the schema of Figure 1.1.

We denote also with $\mathcal{SC}_\mathbf{c} = \{\mathbf{sc}_1, \ldots, \mathbf{sc}_n\}$ the set of all subcomponents belonging to component **c**.

Given the hierarchical subdivision, it is possible to associate a value for the *a-priori probability* of a subcomponent to contain a fault ($\mathbf{P}(\mathbf{sc})$). In order to be consistent with the *single-fault* assumption, it is required that a relationship exists with the probability defined at component level; in particular:

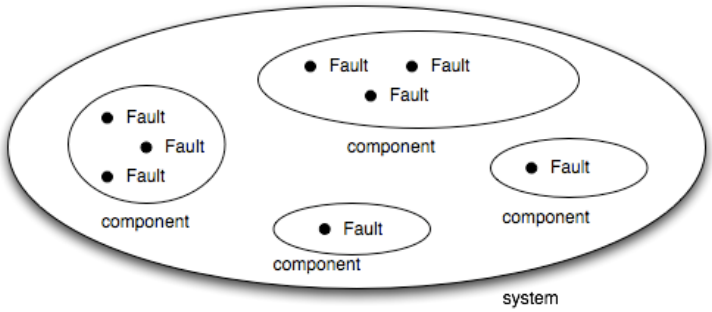


Figure 1.1: Relationship between system, components and subcomponents, faults.

$$\sum_{\mathbf{sc}_j \in \mathcal{SC}_\mathbf{c}} \mathbf{P}(\mathbf{sc}_j) = \mathbf{P}(\mathbf{c}) \tag{1.2}$$

Usually, it is not possible to obtain *directly* the value of the *a-priori* probability at the subcomponent level; for instance, this occurs when the decomposition in subcomponents is introduced to encapsulate different functionalities of an ASIC chip (as in [ME02]) in order to provide a clearer description of the behavior of system itself.

In other scenarios, a subcomponent disposes of its reliability information derived from historical data, and the component level representation is used to describe entities of the system under analysis which are intrinsically related from a *reparability* (or replacement) perspective [BdJV+08a].

1.2.1 Tests

In general, *testing* a system can be described as the operation of performing a measurement of a particular system property, in order to *compare* the result of the measurement with an expected value, usually derived from the specifications of the system [SS94]. In a diagnostic enviroment, measurements are designed specifically to identify a potential *symptom* of a failure, in order to make observable the presence of a fault within the system.

Being a measurement, a test is characterized by an *outcome*; this can be a detailed information about a system parameters (i.e. the value of a sensor measure [Ise05]) or the output values of a combinational circuit [ABF90]. In some cases, the information can be compacted in some form, in order to keep only its most significant part, the correspondence

between the observed value of the parameter and its expected one. When compacted at most, a test outcome is a binary variable assuming value *pass* or *fail*.

Definition 6. A (diagnostic) test $\mathbf{t}$ represents a *measurement* of a property of a system $\mathcal{S}$, targeting at revealing *symptoms* of the presence of a fault.

We denote also with $\mathbf{T}_S = \{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \ldots, \mathbf{t}_n\}$ the set of all *tests* defined for system $\mathcal{S}$, also dubbed as *Test Suite*. We generalize the possible outcomes $\mathbf{o}(\mathbf{t})$ for a test $\mathbf{t}$ using four qualitative labels:

- $\mathbf{o}(\mathbf{t}) =$ PASS, when the expected value of the measurement involved in test $\mathbf{t}$ corresponds to the value obtained from the actual system under analysis;
- $\mathbf{o}(\mathbf{t}) =$ FAIL, when the expected value of the measurement involved in test $\mathbf{t}$ is different from the value obtained from the actual system under analysis;
- $\mathbf{o}(\mathbf{t}) =$ UK, when test $\mathbf{t}$ has not been executed yet on the system;
- $\mathbf{o}(\mathbf{t}) =$ SKIP, covering all scenarios where test $\mathbf{t}$ cannot be executed.

Note For what concerns SKIP tests, this condition can depend on different reasons. In some cases, this outcome is due to the fact that some *preliminary configuration* of the system in order to run the test was not completed: for instance, the presence of a failure in the power system could prevent the execution of *any* operation in a digital device. In other contexts, the outcome of a test is completely deterministic given the result of a previously executed test: in a computer network scenarios, the absence of a link (path) between two hosts will produce a FAIL for each traffic test between them; while a *ping* command would be correctly described by a PASS or FAIL outcome, all traffic tests would be better described with a SKIP outcome, since the operations involved in the test are not usually executed because of the absence of the link itself.

◇

As for components, each test outcome can be associated with a probability value.

Definition 7. The *conditional probability* $\mathbf{P}(\mathbf{o}(\mathbf{t}) = \bar{o}|\mathbf{f})$ represents the probability of obtaining *outcome* $\bar{o}$ after executing test $\mathbf{t}$, given that fault $\mathbf{f}$ is present in system $\mathcal{S}$.

This probability value associated with tests is used to describe, in a unified framework, the *coverage* of a test with respect to a component it stimulates in order to retrieve a failure; such coverage indicates the

proportion of the component which is used carrying on the test and, at an high level of abstraction, this corresponds to the *probability* that the presence of a generic failure in a component results in a FAIL outcome when the test is executed.

The conditional probability described in Definition 7 is not to be strictly interpreted in a dynamic or temporal fashion: the outcome of a test is not produced out of a *stochastic process* for the same system under analysis $\mathcal{S}$; also, the execution of the same test $\mathbf{t}$ on $\mathcal{S}$ usually results in the same outcome $\bar{o}$. Rather, the conditional probability is to be considered with respect to the entire population of all instances of the same system $\mathcal{S}$, and it could be better interpreted as a *proportion of instances* containing a fault in a specific component $\mathbf{c}$ characterized with a FAIL outcome of a test $\mathbf{t}$ because of the presence of the fault in $\mathbf{c}$.

Determining the correct probability values for describing the behavior of a system, especially in presence of faults whose understanding is not deep and complete even for a system expert, is a non trivial task. Furthermore, when it comes to the definition of the probability values from test *coverages*, it is even harder to ensure that the overall coverage and the conditional probability values of a group of tests are *correlated*, i.e., that the probability of detection of a fault in a component is increased by the correct amount, following the increased coverage provided to the system under analysis. The main difficulty of this process is due to the presence of subtle or implicit correlations within the components (or subcomponents) of the system under analysis during the execution of different tests.

Example 1. Let us consider the simplified system described in Figure 1.2 (a). Component $\mathbf{c}_1$ is decomposed in 3 subcomponents $\mathbf{sc}_1$,$\mathbf{sc}_2$ and $\mathbf{sc}_3$, containing respectively 50%, 25% and 25% of faults potentially affecting $\mathbf{c}_1$.

$\mathbf{c}_1$ is tested using two generic tests $\mathbf{t}_1$ and $\mathbf{t}_2$. Making the hypothesis that *any* fault in $\mathbf{c}_1$ produces a FAIL outcome in $\mathbf{t}_1$ or $\mathbf{t}_2$, we obtain a 75% coverage for both tests.

The following table lists the probability of all possible outcome pairs $(\mathbf{t}_1, \mathbf{t}_2)$ when faults are *uniformly* distributed in $\mathbf{c}_1$; the first column indicates the probability of each pair considering only the *coverage* (conditional probability) of $\mathbf{t}_1$ and $\mathbf{t}_2$, while the second column takes into account the overlapping subcomponent ($\mathbf{sc}_1$) for $\mathbf{t}_1$ and $\mathbf{t}_2$.

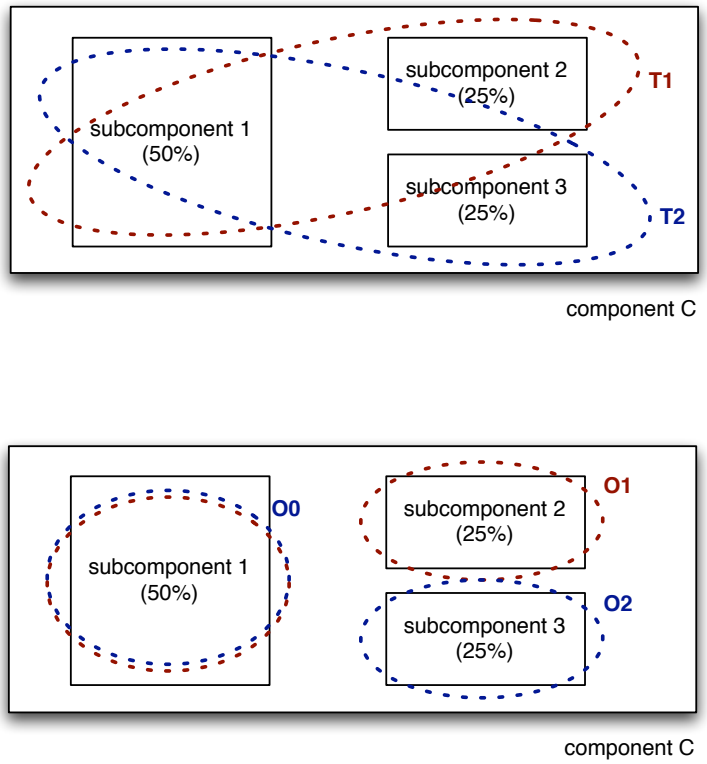


FIGURE 1.2: Description of coverage using tests and operations.

$(\mathbf{t}_1, \mathbf{t}_2)$	Prob.	Real
FAIL,FAIL	62.5%	75%
FAIL,PASS	12.5%	25%
PASS,FAIL	12.5%	25%
PASS,PASS	12.5%	-

◇

Such a task can be simplified a lot when a *divide-et-impera* approach can be adopted: when the effects of the faults can be decoupled for all tests in order to make them *independent*.

Definition 8. An *operation* **op** represents an atomic *task* executed on a specific location of a component, aiming at the computation of a result. Such result can be univocally defined as *correct* or *wrong*, accordingly to the system specification.

We denote also with $\mathcal{OP}_S = \{\mathbf{op}_1, \mathbf{op}_2, \mathbf{op}_3, \ldots, \mathbf{op}_n\}$ the set of all *operations* defined within system $\mathcal{S}$.

Through Definition 8, it is possible to specify a test as a composition of atomic *operation*. Given the fact that each operation can either complete correctly or exit producing unexpected results, a test would likely FAIL if at least one of its atomic tasks has introduced an error, and PASS otherwise. The introduction of the concept of *operations* allows a better resolution in the goal of evaluating how each test stimulate a component functionalities, and, more in general, taking into account correctly subtle components-tests correlations, as in the previous Example.

Any diagnostic methodology has the goal to execute some tests on the system, collecting their outcome and producing an *explanation* of the test outcomes.

Definition 9. A *(diagnostic) conclusion* **dc** referred to a system $\mathcal{S}$ is a *statement*, produced on the basis of the outcomes observed for some or for all tests in $\mathcal{T}_\mathcal{S}$, specifying the causes of misbehaviors $\mathcal{S}$, of which components of $\mathcal{C}_\mathcal{S}$ contain a fault.

We will refer sometimes to *diagnostic conclusions* specifying directly its nature of being a *subset* of faulty components. This will be dubbed Faulty Candidate Components (FCC) set.

Knowledge (Sec.)	Source	Robustness	Scalability	Novelty
Surface 1.3.1:	Expertise	Medium	Poor	Poor
Deep 1.3.2:	Structure	High	Medium	Difficult
Implicit 1.3.3:	Expertise, History	High	Medium	Automatic

Table 1.1: AI-based diagnostic methods.

1.3 State of the art

Artificial Intelligence (AI) has been widely applied to the problem of automating fault diagnosis. In literature, broad classifications of approaches using AI methodologies for fault diagnosis are proposed. Across different fields (as for instance electronic systems [FMM05], chemical systems [VRYK03], computer networks [SS04b]) we encounter minor differences in the accepted classification in *Rule-Based* methods, *Model-Based* methods, *Case-Based* methods.

AI-techniques have used to implement Expert Systems (ES), which try to replicate actions and decisions that a human expert would perform when solving problems related to a particular domain. In [Abr05] author recalls the advantages of ES-based diagnosis approaches:

- to capture human experience in a systematic, introducing a more structured consistency than human experts;
- to minimize and/or to remove the need human expertise, making it replicable and displacable;
- to find and to develop solutions faster than human experts.

A diagnosis ES is based on two entities: *knowledge base*, storing all relevant information (data, rules, cases) and *inference engine*, to seek information and relationships from the knowledge base and to provide answers and predictions, mimicking as close as possible a human expert task. The level of knowledge of an ES reflects the human expertise: it could be either a *surface-knowledge*, obtained from experience, or a detailed and *deep-knowledge* of the system behavior in presence of a fault or not. In some cases, the expertise is not directly, rather new diagnosis are produced *classifying* new instances on the basis of previous diagnosis, performed on similar systems and collected (*implicit-knowledge*).

Table 1.1 presents a taxonomy of AI-techniques, classified according to most significant key-features:

- **Source**: refers the behavior of the system, whether a failure is present or not, and its relation with the diagnostic tests, is dependent on engineering *experience* (direct system inspection, previous systems, qualitative analysis) or from *structural* properties of the

system.

- **Robustness**: refers to the capability of the system to produce the correct response during diagnosis, with an appreciable accuracy (from a statistical point of view), rejecting noise and uncertainties.
- **Scalability**: refers to the possibility to extend the analysis to larger systems, or the capability of the methodology increase the level of detail at which the system under investigation is described without affecting the quality of diagnosis response;
- **Maintenance**: refers on one hand to the ease to introduce changes in the system description and to re-use previous description, or to re-tune the diagnostic responses adaptively to take into account past observations (e.g, erroneous *diagnosis*).

1.3.1 Surface-knowledge methods

In this section we present an overview of methodologies for diagnosis based on a *surface-knowledge* of the behavior of the system under diagnosis. The main exponents of such methodologies are also known as *rule-based* approaches, defined using both deterministic and *fuzzy* logic [KY95].

Relying on description based on surface information only, those methods do not require systematic understanding of the underlying system architectural or operational principles. Those description are easy to develop for small systems, where they provide a powerful tool for filtering out quickly least likely hypotheses.

However, *surface-knowledge* ES systems possess a number of *disadvantages* that limit their usability for diagnosis in more complex systems: they include a poor adaptive learning from experience and inability to deal with unseen problems; this is correlated to the inability to update the system knowledge. Furthermore, *surface-knowledge* methods are inefficient in dealing with inaccurate information. In hierarchical systems, the lack of a reference with the system structure (especially for *rule-based* approaches) makes it very complicated the reusability of descriptions produced for similar or previous systems. Also, rules interactions may result in unwanted side-effects, difficult to verify and change.

Nevertheless, those approaches are important for historical reasons, since they are the first attempt to appear to solve diagnostic problems; furthermore, *rules* are the most immediate instrument to describe failure-symptoms *cause-effect* relationship, to produce a systematic description of engineering expertise. For instance in Abraham [Abr05] the knowl-

edge of the proposed *rule-based* system is a set of if-then rules, connecting together different facts relating observations (tests) and diagnostic conclusions (fault candidates). The inference is performed by a sequential rule interpreter, which *activates* rules consistent with observations, until a unique conclusions is reached.

In [SB06], authors reformulates the problem using rule based inference, which is done by reversing the information contained in a matrix connection model, in the form of logic implication: diagnosis conclusion implies a set of failures, and failing test requires at least on diagnosis conclusion to be true.

Deterministic set partition methods, which divide the components among faulty-candidates and fault free components on the basis of each test outcome and intersecting the resulting tests. Such approach is close to combinatorial group testing techniques, as in [DH00]. The limitation of this approach is related to the fact that single fault hypothesis is required to avoid and exponential growth of computational effort. This method is equivalent to decision tree construction, which is more common in when dealing with test sequencing optimization. Group testing is also analyzed in [NSL08], which covers some heuristics related to the group testing approaches.

In Dexter et al [DB97], some model-based schemes are used quantitative models to estimate the parameters of the system under investigation (a cooling module of an air-conditioning plant). Authors claim that a major problem associated with diagnosis of such system is the definition of a model that exactly matches the process behavior is hard to be defined. Because of this, mismatches between the behavior of the model and the system are unavoidable and they may lead to large differences lowering diagnosis accuracy.

In order to overcome the issue, the proposed approach builds a group of fuzzy relationships are used to define a surface model, in order to describe the fault-free system. Later, a list of all possible faults to be considered for diagnosis is defined by system expert, and an alternative fuzzy model is built for each fault. Fuzzy relationships are under the form of if-then rule, which may apply or not according to the working operating state. Rules are extracted from the system filtering measurements obtained from observations of both working systems and simulations. Since multiple models could be considered to be valid at the same time, a weighetening of rules in order to reduce ambiguity among different models is proposed, when no further measurements (new sensors or probe points) can be inserted due to cost constraints. Demptser's rule is applied also in this methodology to support adaptive strategies and integrating new evidences during testing.

In the approach described in [HN84], Hakimi proposes the idea of adaptive fault diagnosis at system level. The methodology is inspired from microprocessor grids testing, but it could be easily adopted in any *simmetric* scenario characterized by identical components and all disposing of the capability able to test each other functionalities. A test is modeled as the collection of stimuli operations from each entity; an interesting aspect of this work is the possibility that the test outcome can be affected by presence of a failure both on the tested and on the testing units. The work propose *adaptive testing* since tests to be run during diagnosis are not pre-determined, but they are progressively selected to discriminate fault-free elements only, which are capable of determining faulty ones with no uncertainty. The selection metric is based on an optimization on the system elements graph. While proven to be close to optimality in symmetric scenarios, the methodology cannot be extended to systems characterized by the presence of *non-homogenous* components.

1.3.2 Deep-knowledge methods

In this section we present an overview of methodologies for diagnosis based on a *deep* knowledge of the behavior of the system under diagnosis. Exponents of such methodologies are also known as *model-based* approaches. Table 1.2 presents a taxonomy of those methodologies.

The system model may describe the system structure (static knowledge, as in *fault-dictionaries* for digital circuits) and its functional behavior (dynamic knowledge, as in Bayesian networks). Thanks to representing a deep-knowledge of the system behavior, those approaches do not possess the disadvantages that characterize surface-knowledge systems. They have the potential to solve novel problems and their knowledge may be organized in an expandable, upgradeable and modular fashion.

However, the models may be difficult to obtain and keep up-to-date. One of the problems that the approach in with complex topologies with deep-hierarchy, as in [BRdJ+09], which points out a difficulty of developing efficient testing for complex systems: the knowledge about the system is spread over different engineers, then integrating and testing these systems is a time consuming, tedious and error-prone process. Considering also IEEE standards for testing hierarchical embedded (silicon) systems, there is no help in reducing the test complexity. Also, the modeling of hierarchical large embedded manufacturing systems is an aspect which is not covered. One solution proposed solution is to connect events between layers gradually increasing their level of abstraction, but reducing the robustness of the diagnosis.

Method	Family	Domain (Fault Type)	Contribution	Adaptive
Zou et al [ZCRT07]	Fault dict.	Circuits (stuck-at)	Dict. compression (hash-keys)	
Holst et al. [HW09]	Fault dict.	Circuits (stuck-at)		Adaptive simulation
Peng [PR87a]	Fault dict.	Analog circuit	Bayesian inference. Diagnosis conclusion pruning	
Zhang et al. [ZWGC10c]	Error dict.	Circuits (stuck-at)	Fault insertion at output pins	
Butcher [BS07a]	Bayes learn.	Synth. data	contrib	
Zheng et al. [ZRB05]	Belief net	Networks	Efficient entropy eval.	Adaptive probing (measurements).
Huang [HMDPJ08]	BBN	Automotive	Entropy heuristics	2 steps: Minimum cost detection, then fault isolation.
Sheppard et al. [She92]	Inf. flow	Board	Minimal model modification to learn from misdiagnosis.	
Sheppard et al. [SB07]	Inf. flow	Synth. data	Model learning (separability)	
Boumen et al. [BRdJ$^+$09]	Faulty signatures	Network	Hierarchical systems	Test sequencing (entropy)
Ruan et al. in [RZY$^+$09]	Hidden Markov	Sensors data	Dynamic system analysis	
Chen et al. in [CZL$^+$04]	Decision trees		Tree learning from data	
Silva et al. [SSB06]	ANN	Trasm. lines		
Isermann [Ise05]				

Table 1.2: Taxonomy of *deep-knowledge* methods.

1.3.2.1 Fault signatures

In fault dictionary based diagnosis we recognize two main approaches: cause-effect diagnosis and effect-cause diagnosis. For cause-effect diagnosis, all potential faults in a circuit are pre-computed; then, a codebook is build with all faulty responses generated in presence of each fault. Diagnosis is performed by looking up, in the codebook, for the response which is closer to the response of the circuit under analysis. This method saves computational time, since simulation is executed once and offline, but it requires large memory for storing the dictionary.

Methodologies to reduce memory requirements usually lead to diagnosis with weaker resolution and localization power.

Effect-cause analysis reverses the flow, by searching for the minimum size set of fault locations explaining the faulty response of specific test patterns: generally speaking, in a first step a set-covering problem is solved to find an explanation, for each pattern, of the faulty response; then the solutions of each set-covering are aggregated to identify the set of faults explaining both all faulty and fault-free responses. While memory requirements are lower, simulation time is a bottleneck especially for large circuits.

Pous et al. [PCM05] indicate the advantages of faulty dictionary: simplicity for generation (fault signatures) and ease of maintenance; for drawbacks, the fact that only previously defined faults can be detected and located. Shorter dictionaries (from dictionary compression or unmodeled faults) can reduce applicability.

The methdology proposed in [ZCRT07] adopts a compression of the dictionary used in cause-effect analysis, disregarding the resolution loss. The compression is done through a hash-key combining the indices of faulty response outputs. Instead of back-tracing simulation, a search in the dictionary is performed.

Holst et al. [HW09] analyze also fault dictionaries appraoches, pointing out the limitation of cause-effect diagnosis schema (its dependency on a fault model) the limitations resides on the fact that the fault model potentially reflects only a small subset of faults which actually can be found at debug phase.

In [DKW87], the model for diagnosis is proposed in form of properties of the system to be maintained: an example is proposed using an analog circuit, and properties as Kirchoff's or Ohm's laws. The *testing strategy* is defined as the measurement of a difference of an instance of the system with respect to the expected behavior. All potential faults are *pre-listed*

under the form of discrepancy of the system able to explain the difference of measurement. Faulty candidates are determined as the *minumum-size* intersections of the components of the system which are involved in each measurements, in such a way that a *minimalist* explanation can be found for all discrepancies.

The definition of a faulty candidate set for each set of measurements corresponds to the diagnosis. Faulty candidates set research is based on an *incremental test selection* taking into account a *diagnosis monotonicity* property. This requires that, once a component has left the faulty candidates set, because it cannot explain the observed set of measurements, it cannot be inserted again in a later stage of diagnosis.

Authors propose also an interesting function for the evaluation of the next test to be executed, based on *information entropy* of the component to be part of the candidates set and, indirectly, to determine the outcomes for the non executed-yet tests. The *minimization of the entropy* leads to minimum-length test sequences for real circuits. However, authors do not take into account test time as part of the minimization process, nor they consider the case where the execution some tests could be not feasible.

In [PR87a], a causal-effect scenario is designed for the definition of a consistent Bayesian two-level models which can be used for system diagnosis. In particular, it redefines the concept of parsimoniuous covering for failure explanations on the basis of the principles of minimality (the diagnosis should contain the minimum number of candidates to explain the outcomes of the tests), irredundancy (no candidate can be removed from the candidates set without making the diagnosis inconsistent) and relevancy (all candidates are expected to produce the observed misbehavior in the system).

Also, this work proposes a proof about the feasability of incremental diagnosis, supporting the claim of monotonic diagnosability for new outcomes insertion, excluding candidates which are not able to explain new findings.

This methodology is extended in [PR87b], with the goal to reduce the number of candidates evaluation for selecting a minimum number of hypothesis to be considered in a multi-fault scenario. In particular, the strategy is based on the construction of a graph of minimum-size explaining syndromes of test outcomes, based on the conservative methodology presented in [PR87a]. This method organizes the search using a greedy approach using only the most-likely explanation first, in order to build the tree of possible diagnosis for each syndrome.

Wang et al. [WWZP09] propose a methodology for fault localization in wireless sensor networks: in particular, they target a fault identifica-

tion methodology based on end-to-end message exchange, to be preferred to local, device-wise testing (because of energy constraint). The goal of the methodology is to minimize the expected cost of single-link test in terms of end-to-end message exchange and repair time. Authors focus on the identification of all faulty links/devices in a multi-sources / single-sink scenario. Methodology can be applied to both static routing and dynamic routing scenarios. In the case of dynamic routing, each node is associated with a *probability* to be part of the end-to-end communication, according to the routing policy.

Authors assume the binary non-lossy/lossy link as fault model. A diagnostic test is associated with the rate of received packets on a path; the test fails when the rate is below a given threshold. End-to-end information is equivalent to a faulty signature/faulty dictionary model, since a failure in a link is propagated outside the network as a failure of the entire path. Intermediate loss (i.e., transient faults) is not covered. As a consequence, the presence of a faulty link on a path is excluded whenever an end-to-end test is passed on such a path.

Authors propose a methodology to build a decision tree based on the network topology, including all links belongin to a lossy path and excluding links from non-lossy paths. The decision tree ranks the links to be tested locally selecting as first candidates the links which could explain the larger number of lossy paths, weighted by the cost of locally testing that link. Whenever a lossy link is identified, it is repaired and testing is run again; cost minimization is obtained by selecting the most stressed links first.

Neophytou et al [NM09] focus on the optimization of a fault-detection diagnosis dictionary-based technique, targeting test vectors for combinational and sequential circuits. Their approach tries to take into account fault equivalence (or ambiguity) propoerties as a way to optimize the size of the fault test-sets, exploiting the non-specified bits of test vectors.

An alternative to fault dictionaries is proposed in [ZWGC10c], along with an optimal methodology for fault insertion. Fault insertion is performed at hardware level by adding a particular circuit, selectively modifying output pins of modules; errors to be injected are selected in order to be the most representative of internal faults. While independent with respect to diagnosis strategy, paper uses a previous work of authors based on error-flow dictionary. Error - flow dictionary is a methodology extending the fault dictionary approach: errors are reported from register values, and also error order is taken into account.

1.3.2.2 Bayesian Belief Networks

Bayesian networks enable modeling system behavior [SBKM06], in particular for what concerns tracking failures.

One first disadvantage to applying Bayesian techniques is the computational complexity associated with the algorithm inferring conclusions from evidence; exact inference is complex even for bipartite networks, such as those used in the *Quick Medical Reference-Decision Theoretic (QMR-DT) [SMH+ 91]* system.

Sheppard et al. in [SBKM06] point out one limitation of approaches relying on *statistic* or *avarage* properties of the system for diagnosis (system model), obtained from populations of faulty systems. This occurs because, while it can be expected that average behavior to conform to these general statistics of the failures population, indidual instances of a faulty system can exhibit a significant variation from the expected average value.

In this paper, a bipartite bayesian network for diagnosis is used. First, the structure of the network is extracted from human expertise, which determines which fault is detected by which test. Then, the probabilities of such detection relationship are derived from a set of training data corresponding to actual test results, each of them associated with a valid diagnosis.

While the inference problem is not covered in the detail, the paper underlines that a 2-layered structure is simple to be built from human experts, but when it comes to learning probabilities from data such a structure is a non-realistic case to be encountered in real diagnosis. The model is built on the assumption that that test results are conditionally independent given the diagnosis. In fact, in many cases, we find that tests are highly dependent given the diagnosis they are intended to detect. In this paper, the the Tree-Augmented Bayesian network (TAN) is proposed.

The properties of the diagnosis using BBN with respect to the statistical parameters of the population are presented in [SK05]. In particular, this paper redefines the concept of PASS, FAIL test outcomes with reference of a general statistical test. Also, the probabilities of false alarms, and misdetections, are redefined with respect to a Bayesian context. Similarly, authors in [BS07a] propose a methodology for learning BBN-like models in order to produce a diagnosis of systems, along with descriptions described with dependency matrix models and a-apriori information about failures. In particular, the system model designed from an expert is used to create random data about test outcomes in a determin-

istic fashion, while noise is later added onto the synthetic data, in order to train the classifier correctly avoiding overfitting. A Bayesian network is used as a pure *classifier*, the analysis is performed on a dataset of synthetic data mimicking populations of real systems. Besides such simplifications, authors stress the importance of strategies to overcome bias introduced by low-probability fault outcomes.

In Sahin et al. [SYAU07] the construction of the BBN for a diagnosis methodology the scenario of airplane engines is made from a large dataset, and a strategy to exploit the model for fault diagnosis. Instead of using domain knowledge from experts to describe the system, the approach creates the model from sensor data readings. The methodology tries to learn both the bayesian network structure and the components to diagnostic nodes relationship from the dataset, using a particle swarm optimization as the heuristic search methodologies [Nik00].

Stainder et al. [SS04a] introduce a model-based methodology for the the computer network domain, relying on a fault propagation model representing causal relationships from faults to observable symptoms (tests). A symptom-fault map equivalent to a Bayesian bipartite directed graph is used to model, for every fault, a direct causal relationships between the fault and a set of observation related to it.

The method tackles in particular fault localization,the identification of the most likely set of failure explaining the syndromes, taking into account the multi-fault scenario. Such assumption is necessary since a single-fault constraint would be limitative with respect to scalability from medium-size to large systems.

While relationships between faults and symptoms are usually more complex than bipartite graphs capability, it is to be considered an extreme simplification (for instance, it cannot describe indirect effects, or chains of unobservable events). At the same time, the advantage to adopt a bipartite graph is the reducted computational effort for such context; also, the derivation of the model from external observation of the system is feasible, while a more complex structure requires a requires a profound knowledge of the underlying system.

Diagnosis is performed evaluating the *belief* (or likelihood) for each diagnosis hypothesis; hypothesis are groups of candidates containing at least one explanation (fault) for each observed symptom (failure). The approach is incremental in nature because, whenever a new observation becomes available, it verifies the quality of all previous explanations, and extends them of the minimum number of faults, according to the assumption that the minimum number of faults is the most effective explanation in most scenarions. The work presents the requirement of minimality for

the explanation extensions and it proposes a greedy heuristic solving the problem in a polynomial time.

The approach considers in the first place only symptoms associated with misbehaviors (tests with FAIL outcomes), but it is also extented to take into account also PASS tests and, in the network scenario, random loss of observations (SKIP tests).

Zheng et al. [ZRB05] underline that probabilistic quantities as conditional entropy and information gain have not been extensively covered in literature for what concern BBN. Some attempts have been done towards *most-informative* test selection, but disregarding computational complexity. Probability inference is done through *belief propagation*, an algorithm proven to be an *heuristic* non-exact inference on BBN.

Paper suggests to evaluate the information-significativity of each test using an *entropy measure*. The approach proposes to greedy select a test locally minimizing its conditional entropy, given the evidence of previously executed tests. The belief propagation algorithm is modified to compute the entropy at the same time while updating the Bayes Network nodes probabilities.

Methodology is applied on computer networks using information obtained from a simulation framework. Authors propose a condition to stop the selection of new tests based also on entropy; they do not presents their results in terms of savings for what concerns *testing time*, rather as *computational effort saving* of their algorithm in the computation of probabilities and entropy cost function.

In [RBM+05] and [Ris06] another approach based on two-layered BBN is proposed, with the explicit goal to formalize using information-theory concepts the analysis of adaptive testing. Also in this case, the system under analysis is a computer network scenario (where components are network hosts); it is modeled using a codebook approach, where a failure on each component of the system is related deterministically with each test. Authors underline that while the proposed approach targets multi-level hierarchical systems and multi-failure case, the proposed case-study is focused on a flat single-level component-to-test relation codebook, and the limitation to single fault is introduced for computational effort reasons. For what concerns the description of the system through BBN, authors adopt the *noisy-OR* approach [DD01] to decompose the impact of failures to each test in an independent fashion. Also, test outcomes are considered to be affected by noise, whenever misdetection or fault alarm outcomes occur. A parameter pair is used to describe the probability of such noise to occur in both events, and this

results in a unique probability values for all components and tests.

In [HMDPJ08] the focus on fault isolation produces a two step approach to diagnosis. A minimum number of tests is executed periodically in a system, in order to verify wheter a failure is present in the system or not. In the former case, two greedy approaches based on *entropy* are considered, namely incremental and subtractive search, respectively focusing on *isolation* of the faulty component first the former, and the identification of the larger number of non-candidate components first the latter.

In this paper, authors underlines difficulties in building models for systems:

1. with a large number of components and subsystems, whose interactions are potentially complicated;
2. where possible root causes are numerous, and the observations univoquely associating a syndrome with a direct cause limited, which leads to hard interpretation of fault symptom.

Methodology is based on a BBN, whose structure is mult-layered: 3 levels of nodes (from root causes to observations -tests, probes- to intermediate nodes describing common causes). Approach is interesting for the extraction of the model: the structure of the network is derived from a FMEA, while the coefficients of the a-priori are tuned by system experts. In order to create a systematic extraction of conditional probabilities, they are taken from a dataset of previously analyzed systems, obtaining probabilities from *normalized frequencies* of known past cases corresponding to each failures.

Zhang et al. use bayesian inference in [ZWGC10a] for failure classification and propose to create a model of the system of fault syndromes from fault insertion. In a second phase, a Bayesian framework is used to diagnose results for failing boards. The methodology propose a two step approach: in the first, the misbehavior of a faulty system is learned, and it creates a model for the different faulty scenarios. The model is as much accurate as the range of faults simulated at learning phase. A Fault Injection Technique (FIT) operating at pin level is used to create this knowledge. During fault simulation, observable measures on system are taken, in particular register values are extracted. Then, a fault syndrome is created under the form of a binary vector, where each position represents wheter there is match or a mismatch between the measured value and the expected one.

Concerning FIT, [ZWGC09] describes the revisited approach to Fault Insertion targeting high-density ICs and boards. Applicability is limited both by the huge dimensions of potential fault space and the practical difficulties of inserting faults in the system. Authors target a selection

of an effective group of faults in place of the complete set of faults, in order to fully exercise the system. In particular, faults are inserted at pin level: pins are selected so that a fault on them is able to represent the largest number of physical defects. To extend the generality of the approach, pins are selected both at chip and at sub-module (i.e., inside ASICs) level. The methodology exploits circuit simulation tool (RTL description) to simulate the injection of faults (stuck-at faults and flip). For each fault simulated (circuit defect), a correspondency table is built associating each output-pin fault to the defects potentially causing it. An Integer Linear Programming (ILP)-optimization methodology is introduced to obtain a minimum number of pin-level faults which covering the maximum number of interal circuit defects. The approach has as a secondary target the implementation of hardware fault-injection structures within the circuit for reliability purposes, so the ILP model take into account both the maximization of internal faults described represented though a pin-fault and the cost of implementation of an hardware fault injection on the target pin.

In [KDBK10], Krishnan et al. extends test methodologies for analog circuits are relatively firm bases in industrial scenarios, but diagnostic methdologies not have yet, while in presence of a large (academic) literature. Among the main reasons, the lack of a structured procedure for collecting appropriate information from diagnostic engineer, to develop a diagnosis tool for investigating the defective analogue products.

The model of the system is defined from a sufficient number of fail scenarios, representing the intrinsic relationship between the different functional blocks of the circuit, i.e., the reaction of the circuit during the application of test stimuli. This information is mediated by engineering expertise, in order to refine the real relationship at block-level, or at circuit level.

Diagnosis is derived from a BBN multi-layes structure, containing a node for each block of the system. Network blocks and connections identification is done from structural circuits. A-priori probability for each block, and conditional relationships are optimized from a large dataset of previous cases.

1.3.2.3 Information Flow methods

Simpson et al. in [SS91b] present an adaptive methodology diagnosis, known as *Information Flow* method. In this method, tests represent *observable measurements* performed on the system under analysis. Their outcome is reduced to the binary information, PASS or FAIL. The model is

not defined in terms of components but in terms of *diagnosis conclusions* or faulty states, in terms of sets of potential faulty candidates.

The name of information flow depends on the fact that human experts first have to define which test outcomes lead to *direct conclusions*, characterized with a specific component failure. Such information is organized in the form of an oriented graph structure where each node represents either a test or a conclusion, and direct relationships are edges connecting the corresponding nodes. Also, *indirect conclusions* can be drawn from the structure, making an inference on the transitive property of the oriented graph. The graph structure takes into accounts the fact that, in presence of a specific test outcomes, the execution of some other tests can be inhibited, or their outcome can be inferred in a deterministic fashion, making their execution *redundant*.

The methodology is designed for single faults scenarios, but it can handle also multiple fault scenarios if sufficient information is inserted into the graph in the form of multi-failure conclusions. The complexity of this extension depends on the capability of a test engineer to handle the modeling of diagnostic conclusions in multi-scenarios. Robustness of diagnosis with respect to imperfect testing, where a fault is not detected by a test designed to target it, as false alarms, where a test produces a FAIL outcomes is produced even if no fault is present in the system, is obtained through multiple redundant testing strategies. In this methodology, the order of execution of tests is subject to the constraint of following the arcs of the graph in order to diagnose an instance of a system. Fuhermore, the sequence of tests must begin from a root node for each instance. However, with the exception of *direct conclusions*, it is possible to re-order the test connections in order to modify test sequences, for instance, with the goal of minimizing the number of tests to reach specific conclusions. Regarding this aspect, authors introduce an information metric function, based on conclusions still valid derived from executed tests and test execution time, used to evaluate the quality of diagnosis test sequences.

Huang et al. [HMDPJ08] analyze the potential drawbacks of mantaining a strategy based on Information Flow diagram:

- a graph built from expert is usually derived from *if-then* reasoning, allowing only sharp PASS,FAIL conclusions; in real systems, such judgment about root cause should be driven using probabilistic assumptions.
- the sequence of testing is strictly defined, and usually there is no possibility to skip one of the tests of the sequence without loosing effectiveness of diagnosis;
- insertion of new knowledge is difficult, as graph maintenance;

- single fault scenarios are common defining the graph, making the method inefficient on multi-fault systems.

Sheppard et al. formalize diagnosis in [SB07] as pure pattern classification, and prove that the dependency matrix methodology is based on a model related to a linearly separable classification problem. They underline where such linearity property limits the diagnostic resolution of well-known inference algorithms. A test is a generic source of information showing a property of the state of the system (an specifically, failure symptoms). A conclusion is a statement about the element found to be faulty in the system, including no fault found. Also, they also propose a method for deriving optimal diagnostic strategies, proving the feasibility of the construction of a fault-tree in polynomial time.

Approach described in [She92] uses a model of the system to be diagnosed equivalent to information-flow diagnosis, which allows human experts to create a dependency between diagnostic test outcomes and diagnosis conclusions. In particular, authors focus on the *potential lack of dependencies* into the description of the system defined by test engineers which could be due to poor system behaviors understanding. The approach targets single-fault scenario. Diagnosis is performed through a set of inference rules which propagate the information available from tests, discarding inconsistent conclusions and leaving only valid diagnosis.

Instead of focusing on further analysis in order to improve the quality of the model, the methodology guides the execution of additional testing to discover what is correct diagnosis of the failure. The missing cause-effect relationships are extracted modifying the information-flow graph adding the minimum number of missing dependencies.

The *explanation of misdiagnosis* is taken into account in the diagnosis methodology, and it is used as a correction factor for future analysis and for defining a more accurate model of the system itself. Another assumption of the approach, it is required that the information carried by the test cannot be ambiguous, in the sense that a test is always able to detect whether a fault is present on a component by presenting a fail or a pass outcome. Simpson [SS91a] adopts this methodology on a standard Automatic Test Equipment (ATE). Fuzzy logic is used to identify the correct sequence of tests to be executed, and the approach is proven to correctly diagnose industry devices.

Method proposed in Butcher et al. [SS96a] adopts the information flow model but it relies on modeling of a system using fault dictionaries, specifically fault dictionaries designed for detection. Authors limit the scope to *combinational circuits*, and analyze scenarios of *single (stuck-at) faults*, while focusing their contribution on the analysis of potential

sources of error introduced by fault dictionaries, in particular for what concerns non-modeled faults. Test outcomes are analyzed incrementally, according to a set propositional logic rules derived according to Dempster-Shafer theory that combines information from multiple tests using Dempster's rule of combinations. Conclusions are evaluated using the credibility level obtained with Dempster inference theory, an different fault isolation conclusions are accepted or refused.

Alternatively, same authors in [SS96b] tackle the problem of performing fault diagnosis using imprecise models of a system under analysis, and on how to deal with circuit misbehaviors not listed in the dictionary. The algorithm proposed uses the test outcomes obtained from system under analysis and executes a nearest neighbor analysis on the fault dictionary entries. Furthermore, in [SS98] when new tests are introduced for system diagnosis the model is upgraded. Paper establish statistical bases for inferring new conditional probabilities consistent keeping coherency with existing model and minimally affecting the model itself. Conflict are handled through Dempfer-Shaffer inference theory, which degenerates to set covering for deterministic relationships.

In the work proposed by Beygelzimer et al in [BBMR05], they create a matrix based equivalent representation for the information flow model. Authors propose it to allow improved maintenance and more efficient exploration of optimal test sequencing. Their goal is to design a diagnostic strategy which perform fault isolation at *minimum cost*, using as cost metric the number of tests. The *translation* strategy from graphs to matrix is formalized; paper underlines that the flow representation is efficient when dealing with adaptive strategies evaluation, while the codebook representation is usually more efficient while describing the system behavior with respect to syndromes (model maintainance and correction).

A greedy approach for the simplification of a generic information flow graph into a tree structure, where the root represents the beginning of each test session, and each conclusion (fault isolation) can be reached with a unique path. This transformation creates an acceptable solution for adaptive diagnosis, and makes the computation of the expected test session cost straghtforward. Reversely, the translation from codebooks to tree representation allows author to extract important observation about the quality of the optimal adaptive strategy; in particular, authors claim that the minimum size tree can be built from a codebook by selecting at each step the test minimizing the *entropy* of the conclusions, i.e., the test that maximizes the separation between correct and incorrect conclusions.

In Boumen et al. [BDJV$^+$08b], authors focus the definition of testing and diagnosis approach, focused on maintenance phase (and not opera-

tional phase). In the first, some additional aspects are present as the fact that a fix operation can take place at any time (requiring a new execution of test session) and that the probabilities of faults are significantly different than operational scenarios.

The method is based on a dependency-matrix model, similar to fault dictionary / fault-signatures. Probability affects only the presence of a faulty state, but it cannot affect the outcome of the test (deterministic tests).

Authors in [BRdJ+09] propose a method to model and solve a test sequencing problem, based on a dependency matrix, relating faulty states to test outcomes. This approach is defined a hierarchical test sequencing problem, where individual test sequencing subproblems are resolved locally. Positive side-effects of hierarchy-based testing:

(a) the reduced complexity to build a local model (since not all test-components relations are to be designed) and
(b) a reduced computational effort (can produce suboptimal solutions).

Test outcomes are binary (PASS, FAIL). Deterministic relations are considered only; also, test suites are designed to have *maximum resolution* for fault isolation. A *virtual* test outcome (repair) is introduce to describe the situation where the faulty component is isolated after a test execution. Additional information are a-priori probability of faulty states and recurrent/non recurrent test costs.

The hierachical description of the system implicitly introduces the definition of a test at *different level of abstraction*: in other words, an atomic testing probe at an high level of the hierarchy can be seen as a sequence of independent test operations at a lower level. An innovation of this paper is the proposal of describing test-components relations at all levels at the same time, introducing detailed information local to a sub-component only when such information is available. The cost function is designed to minimize the expected cost test, given any faulty scenarios. The heuristic proposed for test selection, at each step of the test sequence, is an information gain, which is the tradeoff between cost test and the capability of the test to discriminate faulty components.

Other methodologies In [ETB07] authors propose a model-based diagnosis technique for wind-turbine. Interesting aspect is the derivation of the model of the system from reliability techniques as FMEA, Fault Tree Analysis.

The work proposed by Ruan et al. in [RZY+09] introduces a distinction between static and dynamic fault diagnosis, depending on the fact that diagnosis is executed after a whole set of test outcomes, or if new

evidences is added during time. Also, the non deterministic relationship between test outcomes and component failures are described in terms of *imperfect* testing, taking into account misdetection or false alarms.

The methodology naturally focuses on multi-fault scenarios, with a low rate of innovation about the information of diagnosis tests. An Hidden Markov Model is designed to model the dynamic time evolution of the system; this model contains a specified set of possible faults, and in each state a fault can be present or not. Information update is done at regular rate, and they are considered only in a limited time window. The Markov model requires all probabilities for test outcomes (considering also false alarms and misdetection) from each faulty state. Diagnosis is solved with a modified form of approximate belief inference. The methodology is based on a simulated annealing optimization problem solving. Results are generated and evalueted with respect to synthetic models of real devices.

1.3.3 Implicit-knowledge methods

In this section we present an overview of methodologies for diagnosis based on a *surface* knowledge of the behavior of the system under diagnosis. Main exponents of such methodologies are also known as *case-based* approaches. Regardless of the type of knowledge used by ES, the fault localization process is driven by an inference engine according to failure-symptoms correlation rules, expressed in the form of labeling. In particular, case-based systems are a special class of *ES* that base their decisions on experience and past situations. They try to acquire relevant knowledge from past cases and previously used solutions to propose solutions for new problems; through the learning of correlation patterns. When a problem is successfully solved, the solution may be used in dealing with subsequent problems.

However, case-based systems require an application specific model for the resolution process, and of a large-dataset of previous cases to avoid mislearning; furthermore, a broad set of different failure manifestations is necessary to avoid overfitting [HTFF05]. From a computational point of view, time inefficiency may make them unusable in real-time diagnosis scenarios. As observed by authors in [SB06], *implicit knowledge* approaches only are able to take into account extremely complex independency properties among faulty scenarios.

Chen et al. in [CZL+04] propose to adopt *decision trees* for failure diagnosis, as they are used as a representation of an expert knowledge to guide a user observing symptoms of failure toward locating the root cause of the problem. While presenting the drawback of poor prediction for

specific faults, the methodology opposes the yield to human-interpretable results, in order to make it more useful in a specific scenarios, as on-field maintenance. The structure of the tree is built from a large database of previous cases. The decision tree to produce a set of rules is designed to maximize the separation between causes: tests which creates a *clearer* separation of fault candidates are selected first, in order to create a *balanced tree*. The separation of candidates is evaluated through the information gain of the test, computed as conditional entropy. These transformations do not change the quality of the final diagnosis, but simply reduces the number of tests required to reach it. *Recall and precision* functions are used for the evaluation of results.

Artificial Neural Networks (ANN) are mainly adopted in scenarios targeting multiple faults diagnosis, especially in digital systems. For instance, in [AJA98] authors analyze combinational circuits; specifically, their methodology is based on training of a network on a pre-computed database of identified potential faults. ANN, as in general automatic pattern recognition approaches, require training data to define their coefficients; such data are generated by inserting one fault in the circuit followed by simulation. This leads to the construction of the learning database labeled with each possible single-fault scenario. Authors propose results for single and double fault scenarios.

Also authors in [OMCE05] propose a methodology based on ANN; meanwhile, they focus on the insertion of *local corrections* that could potentially occur while training the network coefficients using information from model-based simulations. Correction are related to case-based integration of misdiagnosis into the model-based diagnosis inference. Corrections are introducing a neural network intermediate layer into the diagnosis infrastructure: the training set for the neural network is built from a set of simulation from a model-based representation of the system, and a fault-dictionary containing only non correctly-diagnosed records.

Silva et al. [SSB06] propose a methodology for fault detection in localization transmission lines. Fault detection is done using a particular wavelet-transform, producing a signature for the event to be classified as a fault or not. This signature is used to decide whether to store or not the failure record in a case dataset. Such decision follows an if-then rules, designed by experts from a physical model of the transmission line. Once a case is classified as fault (signature), it is recorded in a previous case database. Fault diagnosis is performed using an ANN, trained on the case dataset. ANN are used also in [LHZQ09], combined with Rough Sets theory.

In [CF02] authors propose two diagnosis techniques based on a faulty dictionary, built collecting fault signatures in presence of failures. Dictio-

naries are used to train classifiers designed to infer diagnosis conclusions from system measurements; one classifier is designed as a fuzzy system to extract a set of if-then rules, while the second works on a radial basis function system.

In [PCMdlR03], Pous et al. propose an approach targeting at solving a limitation of fault dictionaries approach. In fact, given that the measurements are taken from a subset of all possible faults, diagnosis is performed by the neighborhood criterion, combined with a distance metric. Such flexibility is a drawback when there is an important difference between the population of faulty systems used to build the dictionary and the actual system under analysis population, producing problems due to non-previously recognized or mis-recognized faults. Dictionaries post-processing are necessary in order to solve ambiguity groups problem, and this computation can come at an high cost.

A Case-Based Reasoning (CBR) engine is proposed as an extension of dictionary-based approach, and it is applied for the diagnosis of an analog filter circuit. In particular, each faulty case is considered as a record containing a fault-class and a set of measurements (faulty component, test outcomes). CBR systems quality relies on a good case base, with significant and consistent case scenarios. In this work, the database of fault cases is built from circuit schematic, describing the system in a hierarchical way (from components to sub-components different sublevels). This description is then simulated, along with the injection of faults at any level of the hierarchy. A Monte-Carlo based sampler is then used to generate a uniform and continuous set of cases, which allows an explicit control of the parameters of the population.

During diagnosis of new instances of the system, the closest case retrieval strategy is based on a euclidean-like distance function; along with this metric, a weightening is proposed using standard filter kernels (gaussian, linear, exponential) in order to smooth the nearest neighborhood classification problems along boundaries. When the real diagnosis of a new instance of the system results in introducing an inconsistency for fault diagnosis produced from previously stored cases, case retaining is performed. Indeed this is the only scenario when a new case is carrying significant information with respect to the correctness of the classifier. Authors underline that the new case insertion could or could not have an impact on previously classified cases: authors propose to handle this scenario using two different case databases, a general purpose database (where new cases not affecting previous classifications are retained) and special purpose database (to held the remaining cases).

A similar approach is extended in [PCM05], introducing a *fuzzy* approach. Authors' target is an analog filter circuit. Fault data are ob-

tained from Matlab/SPICE simulation of the circuit, and also from a real circuit with fault injection. The measurements from the system are classified in a fuzzy classification; fuzzy-set ranges are extracted from Monte-Carlo simulation of the circuit. Then, a set of fuzzy rules for system modeling are proposed associating measurements with parameter values of the circuit under analysis (possible parameter values contain also faults).

In [Ise05], a *model-based* diagnosis approach is proposed for the automotive field. The model of the system is realized according to a block diagram describing dynamic systems. Tests are performed as measurements, and transformed into PASS, FAIL outcomes thresholding the measurement value into acceptable ranges. Diagnosis is applied as *classification*, on a large simulation dataset information, integrated with historical data where available. In particular, in this work diagnosis is based on the *extraction* of a set of diagnosis rules from a Fault Tree.

Case-based approach are indicated in [SB06], where the dataset of previously classified faulty system is used as the source of information for building the Information Flow model. In this work, authors formalize the translation of the diagnosis problem into a classification problem. Authors focus on the classification concept of *linear separability* of a model, which is redefined in terms of *diagnostic resolution* of the methodology, as indicator of the capability to discriminate faulty candidates according to test outcomes.

Butcher et al. in [BS09] present how bayesian networks for diagnosis can be either derived from domain knowledge or learned from actual test, extracting results from debugging or and maintenance data. In particular, for the latter, a stastically significant data-set can be used as an approximation of probability distributions of faults and test outcomes. However, heterogeneity can arise during the aggregation of all data, because of the fact that different components (with non-uniform failure rates) can be used in different production lines, or equipments used for testing slightly different. The process of aggregation itself result in averaging or lost of this information. Potentially, this loss of information can be a cause of inconsistency for the built model; being a Bayesian-based diagnostic engine similar to a classifier, it is as much accurate as there is a correlation between the learning population and future systems to be diagnosed on the field. In diagnostic terms, the learning dataset used to build the model have to reflect one one hand the statistical failure rates, and on the other hand the component to test test diagnosis relationships. The approach proposes to use the information-flow as diagnostic model, whose inferences are to be computed from a dataset of available observation. Conditional probabilities are modified to take into account the

corruption of data (misdetection, or false alarm). The learner to create the model is a naive Bayesian classifier.

Methodology described in [CSK+09] proposes a fault isolation technique on a case-based approach. In particular, a set of classifiers are learned using different pattern-recognition techniques, namely SVM, principal components analysis, nearest-neighbors.

1.3.4 Industrial techniques

1.3.4.1 Product cycle

Dependability, *reliability* and *risk analysis methodologies* cover a broad range of techniques implemented during the analysis of a life-cycle of a product, or applied in larger scale over an industrial process or a system [ALR04]. High quality and performance requirements, low level of failure and risk are always associated with the system functional life. Certification of standard specification conformance and compliance is required, and it can be obtained by applying several methodologies that converge towards this goal. Dependability methodologies are characterized by different features (approaches, strategies) but in particular for the output they provide, either *quantitative* or *qualitative* results.

It is always possible to identify three main steps in the life-cycle of a product [NASb, Pub11]: the *Conception and Design* phase, the *Development and Prototyping* phase and the *Production-Manufacturing and Maintenance* phase. This coarse classification is depicted schematically in Figure 1.3.

For what concerns *dependability*, when moving away on a timeline schedule from the conception phase (and successive phases are reached), the intrinsic cost for problem resolution is an increasing function. This has an impact on process or product *diagnosability*, for they do often require a deeper understanding of interactions among operations and their identification is not straightforward at any step of the development. On the other hand, as a counter part of the cost function, the designer degrees of freedom for correction choices is a decreasing function. This is also known as the *rule of ten* [ZWGC10a]. Most of those techniques are conceived to approach fault detectability as *earlier* as possible.

Conception At this level of evolution, available data covers information derived from similar projects; it is the case for the most of the new products, often targeting the same domain of another system that is already operative, from which it differs for new functionalities or im-

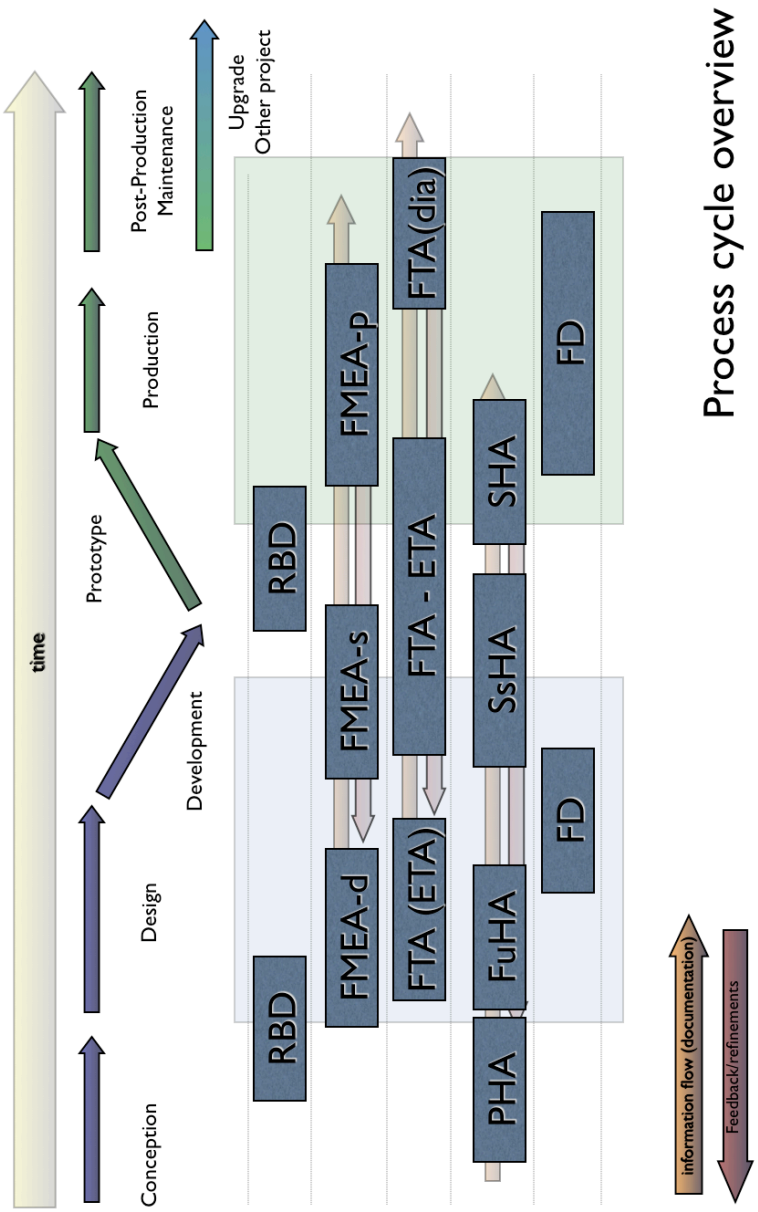


FIGURE 1.3: Simplified development process flow and dependability methodologies implementation.

proved performances. At this stage, it is impossible to apply quantitative techniques (risk estimation or reliability performances), because only generic data about components or subsystems that could be used in current project are available. On the other hand, *functional analysis* and *preliminary hazards analysis* [Eri05] can be applied in order to quantify in a coarse fashion the efforts for risk prevention and criticality issues. Those analysis follow an inductive, check-list approach. [PRO07].

Design In this phase, functional requirements and certification requirements converge together with system specification. As a consequence, organization level of detail regarding the new system increases. Two main strategies can be applied, and they provide complementary results: i) *model-based* metrics: techniques as Reliability Block Diagram are used to identify critical paths and critical items, both in *qualitative* way, as a tool to underline a potential dependability bottle-neck, and with analysis refinements in a *quantitative* way, since modular decomposition allows increased details; this latter kind of analysis is often combined with the use of ii) *statistical* metrics, where probabilistic behavior of elementary modules builds the overall performance estimation of the system. Simulation methodology, with *Montecarlo simulation* methods among the most known, are usually applied to quantify evaluation parameters.The detection of possible faults in the systems appears as a complementary issue but is not yet well defined, because in this phase and in its immediate successors, potential failures can be mere hypothesis and their analysis can be inferred only *by injection* in the system model.

Development and Prototyping Two orthogonal principles characterize [Pub11] this phases: i) functional, subsystem and component analysis become progressively more and more accurate, along with system development. At the same time, ii) precision required increases and cost associated with full analysis have to be contained to match budget requirements.Main focus is on failure modes, that are conventionally identified as misalignment with respect to nominal desired behavior. Causes of misalignments can be unpredictable events, for design mistakes are excluded. Those events are localized and mitigated, in order to quantify criticality (risks, performance loss) and unsafe events (hazards). Two main philosophies are commonly identified in this field, representing causal relationships from a different perspective, *inductive* and *deductive*. In the first family, mainly the *Hazard Analysis* family, focus on potential risks, by decomposing them in potential domino

effects and inducing a sequence of hazardous conditions requiring mitigation strategies. Besides, *FMEA/FMECA* approaches [MR04a] can be built to describe system misbehaviors, by creating an overview of possible effects induced from specific, a priori known, failures. Both techniques aim at reduction of subsystem complexity to small sets of information, that are propagates to higher levels, inducing this way possible sequencing of failure conditions. In the second family, *Fault Tree Analysis* is commonly considered as the reference methodology. This is because of its large application in industry, its wide acceptancy as an efficient tool and its multiple purpose approach, emerging in interesting results throughout several phases of product life. Main aspects in deductive approach cover *logical sequencing* of causes, with a strict direct influence approach in probabilistic sense. This approach reduces complexity for large and highly interconnected systems. Because of this, it makes it possible to perform repeatable quantitative measurements (especially for probabilistic assessment) and diagnosis analysis in terms of minimal explanation identification. Temporal and conditional sequencing are not directly covered by this technique. Some attempts exist to force the model to represent this kind of situations, but the application is not straightforward. Because of this, Event Analysis is preferred for it shares conditional causality but it does not focus on single event but on success/failure paths, or sequencing; this allows a better and deeper handling of failure development in a top-down decomposition approach, even if causality relationship for events is not elicited. In any case, for deductive techniques allow decomposition of the system faults in basilar causal elements (and theit ranking in importance order), those are the most probable candidates for implementation for testing strategies (as it is the case for BIST).

Manufacturing and Maintenance Product certification is required using both formal methods and process validation. In the first case, logical assessment of properties requires logical decomposition that is closer to Fault Analysis than other approaches. However, being this analysis highly dependent on analyst choices, it is also applied in order to diagnose the system to assess misbehavior that could arise in systems after the production phase is committed. As a difference with respect to the same analysis that (could) have been performed, in this case the model is used both for localization and analysis it-self checking, being the situation of Design phase reversed: the failure is an actual condition and its arising cause is to be detected and verified.

1.3.4.2 FMEA/FMECA

FMEA [Sta03] was first systematic techniques for failure analysis and nowadays one of the most widely used analysis technique. This is particularly true for initial assessments the development of a new system (product) [MR04a]. Failure Modes, Effects, and Criticality Analysis (FMECA), is derived from FMEA. It is used indeed to identify and analyze:

- All potential failure modes of the various parts of a system, both products and processes;
- The effects these failures may have on the system and its performance. Figuring out and understand root causes for system failures, it is particularly useful in order to get a better understanding the weakest location in design and realization.
- How to avoid the failures, and/or mitigate the effects of the failures on the system, in order to take by time necessary precautions.
- Possibly yield numerical values for risks evaluation.

As a corollary, FMECA helps organizing design efforts and development priorities. Moreover, the numerical estimation of the risk associated with the introduction and the use of the new system can be used to prove quantitatively the level of availablility or safey it can provide; this information can be communicated to a final user of the application and, in some fields, to a certification agency. Proposals for joint Bayesian and FMEA analysis have been done, as in [Lee01].

Standards Several standard were defined in order to organize FMECA procedures in industrial and production contexts. Among the most important ones [MR04a]:

- IEC 60812 "Procedures for failure mode and effect analysis (FMEA)"
- BS 5760-5 "Guide to failure modes, effects and criticality analysis (FMEA and FMECA)"
- SAE J1739 "Potential Failure Mode and Effects Analysis in Design (Design FMEA) and Potential Failure Mode and Effects Analysis in Manufacturing and Assembly Processes (Process FMEA) and Effects Analysis for Machinery (Machinery FMEA)"

Procedure FMEA-FMECA analysis is put into action by filling up a worksheet-like document. Main information collected regard:

1. A unique reference to each element (subsystem or component). This has to be consistent with respect to design documents, schematics, labels or tags.
2. An exhaustive list of functionalities provided by each item is elaborated. The completeness of the list is fundamental to ensure the validity of the entire FMEA/FMECA approach; usually, ad-hoc check-lists are available, to guarantee this property, especially with short-experienced analysts or large projects analysis, where several team-works has to be combined together.
3. Functional requirements must be fullfilled for the system to be considered ok, in case they are discriminated among all possible operational modes. If not, any exceptional event is listed into the item row as a *potential failure mode*.

For any failure mode identified with the procedure indicated, an analysis is performed, by covering both causes and failure generating processes (for instance, corrosion, fatigue, hostile environments). It is of critical importance to list, for any failure mode identified for a specific components, both local effects (effects for neighbor components or subsystems) and global effects (system failure, unnecessary system operational model switch).

1.3.4.3 Fault Tree Analysis (FTA)

FTA [LGTL85] is an important tool, largely recognized and adopted in industry [PRO07] and implemented in different development phases [NASa]. It involves a set of activities in reliability analysis, namely:

- visualizing *logic dependencies* between (undesired) Top Event, intermediate events or component(s) and initiating cause(s). So, it is self-evident with respect to the propagation of failures into the system.
- extracting priorities about the contributing cause toward the (undesired) top event; this is useful both with respect to vulnerable areas identification at early development steps (when the methodology covers the entire system), and towards cost and ressources minimization, when a mitigation or failure avoid strategy has to be put in place. Because of this, a quantitative performance evaluation can be extracted.
- evaluating design strategies, taking care of perfomances, even when specific data are not available; in this case, it is used as *comparative* tool been its approach general and system-independent enough; in particular, this tool can focus on *sensitivity*, thanks to its decomposition property (importance assessment for compoents towards

system performances). Finally, redundancy policies can be directly evaluated.
- diagnosing a failure cause when the system is in operation, when its localization results hard to detect; in this case, the well-designed FTA can validate all possible sequence of events leading to the system mis-behavior, and detection is performed using its implicit priority imposed on initiating events.

Information and results, and standards The methodology is based on a three-phased process: identification/definition, implementation, evaluation. In the former, a failure event is retrieved, both with its scope (contributors that can be included or not, system boundaries) and resolution, the level of details to which the analysis must be concluded). Critical aspects of the design of the FT are the correctness of the selection of a Top Event with respect to the problem that is to be solved. Correct definition of boundaries defines the roles that other interacting subsystems can have in presence of a failure. Moreover, it is important that events are considered as simpler as possible, to allow a correct and rigorous *decomposition* in the following (deductive) steps. During the second step, the tree is constructed by decomposing any node (event) into its possible immediate causal factors. Logic deduction is fundamental, and can be assisted by the system description (component, functionalities) and by complementary informations the actual development phase at which the analysis is performed can provide. Latter step covers a tree evaluation and interpretation. For evaluation, most common form is the *minimal cut sets*, i.e., the combination of events that could arise into top event failure. This evaluation is both in terms of dimensionality (minimal explanation) and probability (most likely event); this information is as most numerically fiable as far as the analysis was performed. FTA methodology is define in international standards (DIN 25424, IEC 61025 [IEC90]) and other literature as [NUR].

Comparing FTA with FMEA FT and FMEA differ mainly on the direction of the analysis flow [BPG05]. FTA begins from undesired events, and trace them backwards to their causes. Once the primary ones among them are identified (and quantified in terms of probability), the analysis can be considered over. Inductive methods, on the other hand, requires the identification of initiating causes, by applying the path of possible consequences. FMEA however does not define a logical

sequencing model regarding every step is perfomed. Thus, even if both analysis can converge towards a common level (functional failure, subsystem failure) presented in their results, there is no specific and repeatable way to match information from both approaches. However, there is no theoretical reason preventing mutual validation: an inductive approach can be used to verify that in a given system it is possible to reach an initiating cause that has been retrieved in a top-down FT decomposition. On the other hand, even if it is not possible to directly patch together multiple FMECA to obtain a FT, results from the first could be reanalyzed by a FT, verifying by this the likelihood of the inductive process.

1.3.4.4 AI-ESTATE

Sheppard et al. describe in [SW06] some advancements of the AI-ESTATE standard. This is the only standard explicitly aiming at providing a formal model for applications exploiting AI methodologies fault diagnostic domain, in order to guarantee a non-ambiguous exchange of information and a consistent cross-applications interface. In the standard, two elements play an essential role.

- the Common Element Model: it specifies common elements which are found in all reasoning approaches;
- the Fault Tree Model, Diagnostic Inference Model, and Enhanced Diagnostic Inference Model, which are designed to contain information of specific approaches to diagnosis.

They present also an overview of the insertion of BBN information model into the standard. In particular, authors underline the binary form of test outcomes (PASS, FAIL), the labeling of possible diagnosis conclusion for each component (good or fault-free, suspect, candidate). Finally, a subset of the standard is dedicated to improve the definition of 'test session' concept. In particular, the standard provides an organized approach to archive of past diagnostic sessions This is to be done for both online interaction of diagnosis tool and offline data analysis.

Bayesian Belief Networks for Diagnosis 2

We recall the preliminary elements related to the scenario of functional diagnosis of boards as well as some basic information on Bayesian Belief Networks (BBNs). Diagram representations of probability distributions offer different advantages for the analysis of complex scenarios, and they are widely used in AI applications in machine learning [Mit97], or statistical learning and pattern recognition [TK08].

As pointed out in [Bel06], those properties are summarized in the following categories:

1. **Visualization:** graphical models provide a simple while useful way to visualize the structure of a probabilistic distributions;
2. **Modeling:** inspection of the graph shows more directly the properties of the distributions (e.g., conditional independence), and it allows a more efficient handling of the impacts of its update (model maintenance);
3. **Inference:** graphical manipulations, associated with underlying mathematical expressions, often provide a more efficient way to carry along complex computations (inference, learning) in an implicit way.

When graphs are used to describe a probability distribution, *nodes (or vertices)* represent random variables, and *edges (or arcs)* represent probabilistic relations between variables (e.g. conditional dependency). Graphs are adopted as tools since they *provide a compact representation* of complex Joint Probability Distributions (JPDs) [Mur98]. This occurs because clusters of connected nodes represent the *joint probability distribution* of all random variables at it can be decomposed into a product of

independent factors, each relying the subset of the variables contained in the cluster).

Two main classes of graphical models exist:

- *Undirected graphical models*, also known as Markov random fields, which are models where links have no orientation and the relationship between connected nodes express soft constraints among random variables;
- *Directed graphical models*, also known as Bayesian networks, which are models where links in which the links have a particular directionality and they express causal relationships between random variables.

2.1 Overview of BBNs

According to the proposed classification of graphical models, BBNs are represented as Direct Acyclic Graphs (DAGs). A BBN containing n nodes can be transleted univocally with a JPD $\mathbf{P}()$ containing n random variables.

$$\text{BBN} \iff \mathbf{P}(\mathcal{X}) = \mathbf{P}(X_1, X_2, \ldots, X_n)$$

While the theory is general and regards both *continuous-* and *discrete-*valued variables, we will refer only to discrete-valued variables. Then a generic random variable X can only assume a finite set of K possible values $\{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_K\}$; we will implicitly consider the *binary* true/false case ($K = 2$, and $X = 0$ (false) or $X = 1$ (true)). Furthermore, for the sake of simplicity of notation, we will indicate with the expressions $\mathbf{P}(X_1, X_2)$ or $\mathbf{P}(X_1, \bar{X}_2)$ respectively the probability values $\mathbf{P}(X_1 = 1, X_2 = 1)$ and $\mathbf{P}(X_1 = 1, X_2 = 0)$.

2.1.1 Factorization

In set of variables $\mathcal{X} = \{X_1, X_2, \ldots, X_i, \ldots, X_j, \ldots, X_n\}$ it is possible to establish an order. Such order must respect the property that, if there is an arc from node X_i to node X_j, it has to be $j \geq i$ for the indices pair (i, j). Given those nodes X_i and X_j, X_i is generically defined a *parent* of X_j, and dually, X_j is a *child* of X_i. Furthermore, the set of parent nodes of X_j is denoted by the notation *parents*(X_j).

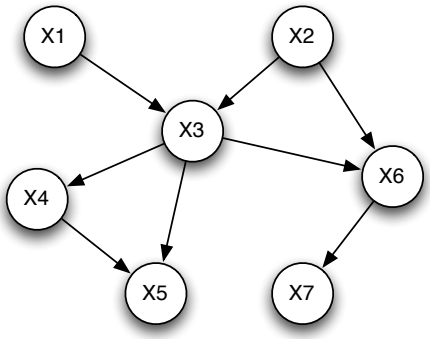


FIGURE 2.1: Example of BBN.

An example of a valid order is depicted in Figure 2.1. Any order, respecting this property, can be used to decompose (*factorize*) the JPD $\mathbf{P}(\mathcal{X})$ using *conditional probabilities*:

$$\begin{aligned}\mathbf{P}(X_1, X_2, \ldots, X_{n-1}, X_n) = \quad & \mathbf{P}(X_n|X_1, X_2, \ldots, X_{n-1}) \cdot \\ & \mathbf{P}(X_{n-1}|X_1, X_2, \ldots) \cdot \cdots \cdot \\ & \mathbf{P}(X_2|X_1) \cdot \mathbf{P}(X_1)\end{aligned}$$

Such *factorization* can be further simplified. According to probability theory, a conditional probability of three variables (A, B, C), e.g. $\mathbf{P}(A|B, C)$ can be simplified as $\mathbf{P}(A|B)$ when variables A and C are *independent* (or $A \perp C$).

$$A \perp C \implies \mathbf{P}(A|B, C) \equiv \mathbf{P}(A|B)$$

A BBN of n nodes is a valid representation of a JPD of n variables if, and only if, can be factorized in a set of n conditional probabilities; furthermore, if the conditional probabilities of each variable depends *only* on a subset of variables corresponding to the *direct parents* of the node on the graph. Variables having no parents in the graph, i.e. variables which are not dependent to any other one, are characterized by a probability expression in the form $\mathbf{P}(X_k|) = \mathbf{P}(X_k)$, indicated as *a-priori* probability.

Thus, the BBN allows the representation of the JPD of all variables as a product of a-priori and conditional terms, each depending only on their parents' variable. Those functions can be interpreted as the set of

parameters defining univocally the BBN model.

$$\mathbf{P}(X_1, \cdots, X_n) = \prod_{n=1}^{n} \mathbf{P}(X_i|\text{parents}(X_i)) \tag{2.1}$$

Example 2. Let us consider the BBN depicted in Figure 2.1, containing 7 nodes $\mathcal{X} = \{X_1, \ldots, X_7\}$. Trivial, non-simplified factorization of the JPD is the product of seven contributions $\mathbf{P}(X_k|X_{k-1}, \ldots)$.

According to the graph, we can obtain the simplification of these terms due to variables' independence: for instance, variables X_1 and X_2 are independent from all other variables, then are characterized by an *a-priori* probability. Or variable X_7 and X_4 depend only, respectively, on X_6 and X_3. Then

$$\begin{array}{rcl} 7 & : & \mathbf{P}(X_7|X_6, X_5, \ldots, X_1) \rightarrow \mathbf{P}(X_7|X_6) \\ 6 & : & \mathbf{P}(X_6|X_5, X_4, \ldots, X_1) \rightarrow \mathbf{P}(X_6|X_3, X_2) \\ 5 & : & \mathbf{P}(X_5|X_4, X_3, X_2, X_1) \rightarrow \mathbf{P}(X_5|X_4, X_3) \\ 4 & : & \mathbf{P}(X_4|X_3, X_2, X_1) \rightarrow \mathbf{P}(X_4|X_3) \\ 3 & : & \mathbf{P}(X_3|X_2, X_1) \\ 2 & : & \mathbf{P}(X_2|X_1) \rightarrow \mathbf{P}(X_2) \\ 1 & : & \mathbf{P}(X_1) \end{array}$$

And the JPD in the form of Equation 2.1 is a function of seven contributions (BBN parameters):

$$\begin{array}{rcl} \mathbf{P}(\mathcal{X}) & = & \mathbf{P}(X_7|X_6)\mathbf{P}(X_6|X_3, X_2) \\ & & \mathbf{P}(X_5|X_4, X_3)\mathbf{P}(X_4|X_3)\mathbf{P}(X_3|X_2, X_1)\mathbf{P}(X_2)\mathbf{P}(X_1) \end{array}$$

◇

2.1.2 Inference

Thus, BBNs are important as *visual* tools, allowing the identification of variables dependency and independency at a glance of the graph. This is usually more efficient than tedious equations inspection, especially for JPD of hundreds or even thousands of nodes, and the maintenance of the probabilistic model they describe is an easier task. Other properties of the JPD underlying the BBN can be extrapolated from the topology of the graph (e.g. *conditional independence* or *D-separation*), and a reference can be found in [Jen96].

Besides of this, BBNs are also important because they support a more efficient *probability inference* (or update), which is necessary when one or more *observations* are available about the *true value* of some variables (also indicated as *findings* or *evidences*). A well-known example in BBN literature regarding probability inference is the *rain-sprinkler-wet grass* example, in [Mur98].

Bayes' Theorem states that, for a pair $\langle X_i, X_j \rangle$ of random variables:

$$\mathbf{P}(X_i|X_j) = \frac{\mathbf{P}(X_j, X_i)}{\mathbf{P}(X_j)} = \frac{\mathbf{P}(X_j|X_i) \cdot \mathbf{P}(X_i)}{\mathbf{P}(X_j)} \tag{2.2}$$

In the context of BBNs, $\mathbf{P}(X_i)$ and $\mathbf{P}(X_j)$ are the *a-priori* probabilities of variables X_i and X_j, while $\mathbf{P}(X_j, X_i)$ and $\mathbf{P}(X_j|X_i)$ are respectively the JPD of those variables and the conditional probability defined by the BBN structure.

The *conditional* probability $P(X_i|X_j)$ of having X_i, given the *observation* of the true value of X_j is also referred to as *a-posteriori* probability of X_i. If we consider the variable order ($i \leq j$), Equation 2.2 can be reformulated as a function of known *a-priori* ($\mathbf{P}(X_i)$) and *conditional* probabilities ($\mathbf{P}(X_j|X_i)$):

$$\mathbf{P}(X_i|X_j) = \frac{\mathbf{P}(X_j|X_i) \cdot \mathbf{P}(X_i)}{\mathbf{P}(X_j|X_i) \cdot \mathbf{P}(X_i) + \mathbf{P}(X_j|\bar{X}_i) \cdot \mathbf{P}(\bar{X}_i)} \tag{2.3}$$

Equation 2.3 can be extended to take into account any set of conditioning variables. The task of *inference* on BBN is then the computation of all *a-posteriori* probabilities, given the set of true value for a subset of observed variables $\mathcal{X}_{obs} \in \mathcal{X}$. Probability inference is also the methodology allowing the adoption of BBN as a diagnostic tool [Agr96], as it is explained in the next Section.

2.2 Two-layer BBNs for system diagnosis

BBN models are merely an alternative representation of generic JPD of a set of random variables. Although, when a *semantic* is added the model and its parameters, a BBN represent also the notion of causality among the correlations of occurring events [Pea00] (*causal networks*).

In a diagnosis framework, each variable X_i is associated with an *event*. Each event may happen or not, and the corresponding random variable X_i reflect this condition, respectively with $X_i = 1$ or $X_i = 0$. Events are connected in causality chains (or in *trees*). The events at the root of a chain may happen according to their *a-priori* probability; furthermore, *parentless* nodes represent by construction events which are

completely *uncorrelated* (probability independence). All other events probability can be computed based on the occurrence of their parents $parents(X_i)$, through conditional probability values.

In our scenario, each variable X_i can be associated with either a *component* $\mathbf{c}_x$ event (with probability $\mathbf{P}(\mathbf{c}_x)$) or to a *test* $\mathbf{t}_z$ event (with probability $\mathbf{P}(\mathbf{t}_z)$). A component $\mathbf{c}_x$ *binary* event indicates if component $\mathbf{c}_x$ is either *fault-free* or *faulty*. A test $\mathbf{t}_z$ *binary* event indicates if test $\mathbf{t}_z$ outcome is PASS or FAIL.

Given a system $\mathcal{S}$ under investigation, and the suite of functional tests for performing the diagnosis process, the BBN model of the system provides all conditional probabilities $\mathcal{P}(\mathbf{t}_z|\mathbf{c}_{x1}, \ldots, \mathbf{c}_{xn})$, expressing the relation between the PASS (or FAIL) outcome of test $\mathbf{t}_z$ depending on the state of the components. According to BBN formalism, the conditional probability of a specific test depends exclusively on the *fault-free/faulty* status of the components it involves ($\{\mathbf{c}_{x1}, \ldots, \mathbf{c}_{xn}\}$).

In the diagnosis context, two further assumptions are usually introduced

- the set of *components* is complete;
- all *components* are mutually exclusive.

The former assumption guarantees that the model includes all possible causes of failure potentially affecting the system under investigation: *at least* one component *must* be faulty, in order to the BBN to represent a meaningful diagnosis process. This can be also reformulated as the consideration that there is no failure, among the all possible ones potentially affecting the system, that cannot be described with a *faulty* state on one or more components of the system itself. The latter assumption, instead, determines that failures affecting different components are independent, and they could occur concurrently. The diagnosis might end up identifing two faulty components, even though such a situation is unlikely to occur on real systems. Given those assumptions, the BBN is known in machine learning literature as Naive BBN [Mac03]. In general, the hypothesis of a single failing component is commonly adopted, nevertheless there are syndromes that cannot be explained with the failure of a unique component; it is important thus that used reasoning engine considers this less probable situation, should it occur.

2.2.1 Components-Tests Matrixs (CTMs)

Figure 2.2 depicts a generic BBN for system diagnosis. The model is a *bipartite* graph, composed of two groups of nodes:

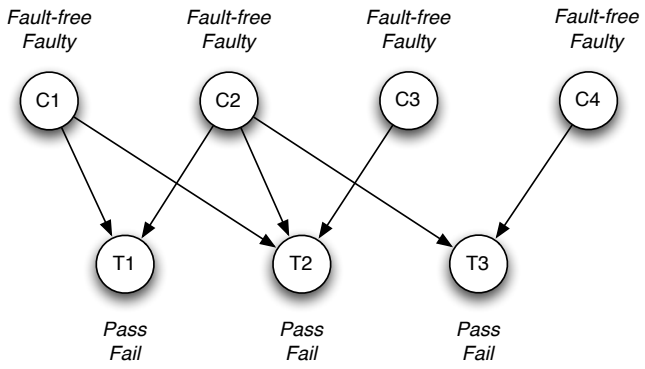


FIGURE 2.2: A BBN model for diagnosis.

- parent-less nodes, representing components; a unique parameter describe the node, its *a-priori* probability function;
- child-less nodes, representing tests; each node is requires a set of parameters to be described, contained in its *conditional* probability function;.

Components are described with an *a-priori* probability value in the range $[0, 1]$, representing the *absolute* likelihood of the component to be in a *faulty* state. In some cases, there is a close correspondency between components and actual physical devices instantiated in the system under analysis: therefore, *a-priori* values are derived from reliability information, failure rate indices from data-sheets, or other similar sources. Otherwise, if no precise information can be obtained, e.g. when components nodes in the network are used to represent either devices' blocks or a group of devices, then uniform or context-specific probability distributions can be adopted to tune *a-priori* probability values. In both cases, experience and previous diagnosis feedbacks are used as a source to *improve* the model over time, establishing a better correspondency between *a-priori* parameters for all components and the specific rate of occurrence of failures in the actual system.

Tests are described by *conditional probability functions*. Dealing with binary random variables, this function for each generic test $\mathbf{t}_z$ is univocally defined with a set of 2^{n_c} values in the range $[0, 1]$, where n_c represents the number of component nodes in the BBN. An alternative representation of this function is a probability table with 2^{n_c} entries.

Such function is *oversized* to univocally describe the causal relation between *faulty* components possible configurations and the outcome of test $\mathbf{t}_z$. The first simplification concerns in general the independence property underlying the BBN: components nodes which are not part of the test $\mathbf{t}_z$ node *fan-in* (components nodes not having an arc to $\mathbf{t}_z$) have no contribution for the definition of the conditional probability table of node $\mathbf{t}_z$. The second simplification regards the assumption that each component node of the fan-in of the test node contributes *independently* to the causal relation, following a conditional table decomposition approach which is known in bayesian literature as *canonical model* [DD01], and in particular the Noisy-OR (nOR) decomposition [Sri93].

Let us consider a component-test pair $\langle \mathbf{c}_x, \mathbf{t}_z \rangle$; their causal relationship is described by a pair of conditional probabilities:

1. $\mathbf{P}(\mathbf{t}_z = \textsf{FAIL} | \mathbf{c}_x)$, or the probability that the outcome of $\mathbf{t}_z$ is PASS, when $\mathbf{c}_x$ is *fault-free*;
2. $\mathbf{P}(\mathbf{t}_z = \textsf{FAIL} | \bar{\mathbf{c}_x})$, or the probability that the outcome of $\mathbf{t}_z$ is PASS, when $\mathbf{c}_x$ is *faulty*.

Dealing with *binary* variables, the probability of PASS outcomes can be obtained by difference. The former parameter is used to represent the probability of component $\mathbf{c}_x$ of being the cause of a *false alarm*, producing a FAIL outcome without any failure to be present in the system. The latter term, instead, is used to represent the probability of test $\mathbf{t}_z$ to *reveal* the presence of a failure on component $\mathbf{c}_x$, *whatever the nature of the failure is*. For this reason, this term is strictly correlated with the concept of test *coverage*.

We describe the composition rule of the nOR approach in the following Example 3. According to the nOR approach, only two parameters are sufficient to represent exhaustively each component-test pair relation; therefore, the number of parameters required to describe the overall conditional table of each test is *linear* (and not exponential) in the number of components the test involves.

Example 3. Figure 2.3 depicts a trivial BBN composed of two components $\mathbf{c}_1$ and $\mathbf{c}_2$ and a single test $\mathbf{t}_1$. The causal relationships within the pairs $\langle \mathbf{c}_1, \mathbf{t}_1 \rangle$ and $\langle \mathbf{c}_2, \mathbf{t}_1 \rangle$ are described independently in the tables on the right. **NF** and **F** labels on components represent respectively the *non-faulty* or *faulty* states, while the **P** and **F** indicate respectively the PASS and FAIL outcomes. Even if there are four entries per table, only two parameters describe completely each table: the *false-alarm* probability and the *fault-detection* probability.

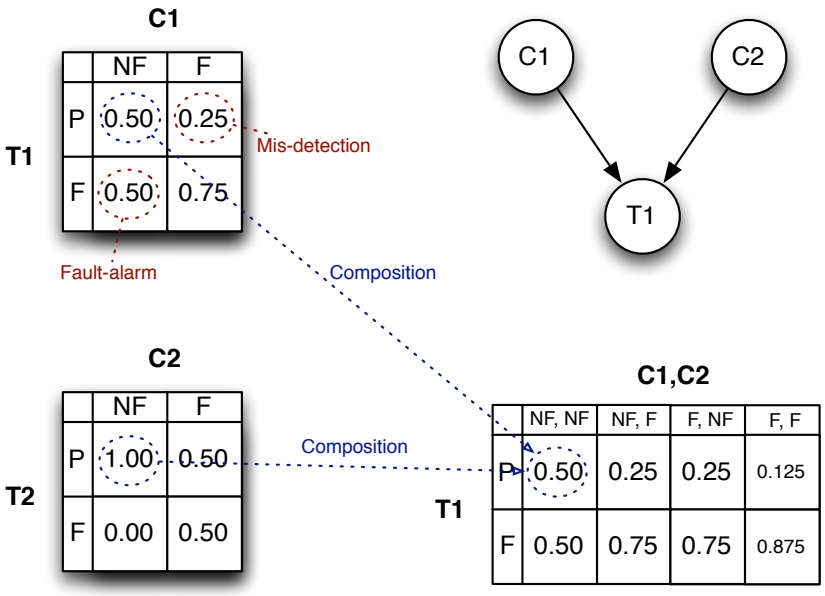


FIGURE 2.3: nOR decomposition example.

nOR decomposition is based on the assumption that the FAIL event can be caused independently from the two sources $\mathbf{c}_1$ and $\mathbf{c}_2$. Therefore, for a PASS outcome to occur, it is necessary that *both*, *independent* failure sources do not produce a FAIL event; the probability of this composed event can be computed as the product of single-event probabilities, because of independence, for each possible permutation of faulty states (**NF**,**NF**), (**NF**,**F**), ….

One of the two rows of each table can be obtained by difference.

$$\begin{aligned}\mathbf{P}(\mathbf{t}_1 = \text{PASS}|\mathbf{c}_1, \mathbf{c}_2) &= \mathbf{P}(\mathbf{t}_1 = \text{PASS}|\mathbf{c}_1)\mathbf{P}(\mathbf{t}_1 = \text{PASS}|\mathbf{c}_2) \\ \mathbf{P}(\mathbf{t}_1 = \text{FAIL}|\mathbf{c}_1, \mathbf{c}_2) &= 1 - \mathbf{P}(\mathbf{t}_1 = \text{PASS}|\mathbf{c}_1, \mathbf{c}_2)\end{aligned}$$

For instance, the probability that execution of $\mathbf{t}_1$ results in a PASS outcome when only $\mathbf{c}_1$ is faulty while $\mathbf{c}_2$ is fault-free can be computed as the probability of two independents events: no *false-alarm* due to $\mathbf{c}_2$, and *mis-detection* of failure on $\mathbf{c}_1$. Then:

$$\begin{aligned}
\mathbf{P}(\mathbf{t}_1 = \textsf{PASS}|\mathbf{c}_1 = F, \mathbf{c}_2 = NF) &= \mathbf{P}(\mathbf{t}_1 = \textsf{PASS}|\mathbf{c}_1 = F) \cdot \\
&\quad \mathbf{P}(\mathbf{t}_1 = \textsf{PASS}|\mathbf{c}_2 = NF) \\
&= 0.5 \cdot 0.5 = 0.25 \\
\mathbf{P}(\mathbf{t}_1 = \textsf{FAIL}|\mathbf{c}_1 = F, \mathbf{c}_2 = NF) &= 1 - \mathbf{P}(\mathbf{t}_1 = \textsf{PASS}|\mathbf{c}_1 = F, \mathbf{c}_2 = NF) \\
&= 1 - 0.25 = 0.75
\end{aligned}$$

◇

In our framework, we focus on diagnosis scenarios where *false alarms* probability is *null*; we consider that a test $\mathbf{t}_x$ can be repeated, producing the same outcome, therefore removing this potential source of ambiguity. Because of this, a single parameter (*fault-detection rate* or *coverage*) is sufficient to describe each pair component-test.

Summarizing the entire BBN model containing n_c component nodes and n_t test nodes can be represented using $(n_c + 1) \times n_t$ parameters, or formally with:

1. a vector of size $n_c \times 1$, containing a $[0, 1]$ *a-priori* probability value $\mathbf{P_{ap}}(\mathbf{c}_x)$ for each component $\mathbf{c}_x$ of the BBN;
2. a matrix of size $n_c \times n_t$ containing a $[0, 1]$ *coverage* value $\mathbf{cov}(\mathbf{c}_x, \mathbf{t}_z)$ for each component-test pair $\langle \mathbf{c}_x, \mathbf{t}_z \rangle$ of the BBN; this matrix is dubbed Components-Tests Matrix (CTM).

It is worth noting that the model extraction activity requires strong a test engineers' team activity. The component set can be derived from the specification of the system: for instance, in the case of a digital circuit, it results from a combination of a netlist inspection and the knowledge about the use of the system.

The test selection requires the intervention of the test engineers' team whose role is to identify the used test set, a subset of all possible tests, for system diagnosis; then, for each test, to specify the coverage level for all test-component. In order to face with the difficulty of computing an accurate value for each coverage probability, a *simplified qualitative* scale for the coverage is used: causal relations between each $\langle \mathbf{c}_x, \mathbf{t}_z \rangle$ pair are described through a *coverage* label $\mathbf{cov}_L$, belonging to the *coverage label set* $\mathbb{C}_\mathbb{L} = \{H, M, L\}$, indicating a *high*, *medium* and *low* coverage level for a test with respect to a specific component. Qualitatively, the coverage can be also considered as the amount (proportion) of functionalities of the component involved during the execution of the test; the smaller the *portion* of the component stimulated, the lower the probability that *any* failure results.

Table 2.1: CTM excerpt of system in Figure 2.4.

	$\mathbf{c}_1$	$\mathbf{c}_2$	$\mathbf{c}_3$	$\mathbf{c}_4$	$\mathbf{c}_5$	$\mathbf{c}_6$	$\mathbf{c}_7$	$\mathbf{c}_8$	$\mathbf{c}_9$	$\mathbf{c}_{10}$	$\mathbf{c}_{11}$
$\mathbf{t}_1$	L	-	-	L	-	-	H	-	H	M	M
$\mathbf{t}_2$	L	-	-	L	H	H	-	-	-	-	-

Another symbol (–) should be used to model the condition where a faulty condition of the component is not able to cause a failure of the corresponding test, since none of its functionalities are used during test execution.

The decision to express each component-test relation by means of a label derives from the need to simplify the work of the team modeling the behavior of each available test and its impact of the components; in fact each label expresses the qualitative relation between components and tests, moving the management of uncertainty to the adopted resolution strategy.

The modeling of test behavior and the impact on component through qualitative labels provide also a strategy to take into account bad fault modeling. A test engineer can exploit this flexibility, based on experience or expertise, and adapt the coverage of specific test-component pairs, in order to improve the model where diagnosis from test outcomes is unclear.

CTM and *virtual* components Two components $\mathbf{c}_{x1}$ and $\mathbf{c}_{x2}$ are different from the model point of view if and only if they differ for, at least, one element of the CTM. Motivations for this requirement are twofold: qualitative and quantitative.

From a *qualitative* perspective, the system analyst specifying the CTM can define the system model with multiple different tests covering the same subset of components with the same coverage labels. This situation corresponds to a replicated column in the matrix, and it is not a problem for the consistency of the model; for instance, two tests could stimulate a set of components with the same coverage levels, although interacting with different component functionalities. On the other hand, a row replication in the CTM is quite improbable, since it would model two (or more) components with identical partial and complete syndromes (for all tests the outcomes are identical). Such a pair (or set) of components, even if from an architectural perspective they can be identified

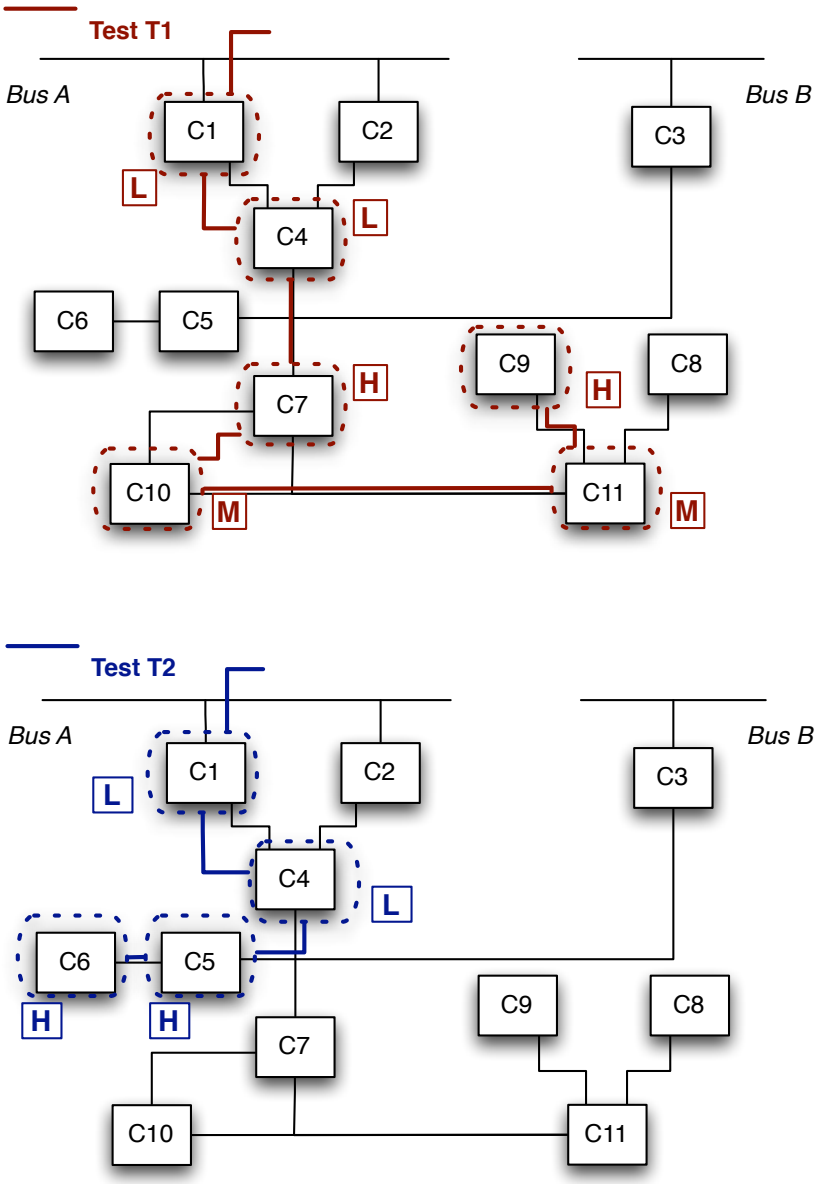


Figure 2.4: Sample board and component-test coverages.

with specific and separated elements of the system, should be considered as a *unique logical* component. Consequently, the responsibility of the failure would be assigned to this new *virtual* logical component rather than being distributed over the pair of components, with a lower and uniform degree.

From the *quantitative* perspective, the replication of a row in the CTM causes a duplication of a component node used for probability inference. Because of the independence property of the nodes of a BBN, two component nodes characterized by the same *faulty signature* will match their faulty probability value whenever the outcome a newly executed test is inserted, during the incremental test phase. This duplication is misleading from both a modeling and a computational point of view, because of the redundancy of the numerical processing (identical equations are solved twice at each step).

Therefore, without lack of generality but rather referring to a *reasonable* model, we require the CTM not to have two identical rows to avoid an increase in complexity and computational time.

Furthermore, it is worth noting that this assumption is not preventing the analyst to model intermittent failures, since this concept can be considered *orthogonal* to the coverage label ranking. In fact, it is possible to modify the coverage level in order to consider both pass and fail outcomes of a test to agree with the faulty component diagnosis obtained with the BBN.

2.2.2 Inference for two-layer BBNs

In literature [Mur98], several specific algorithms have been developed to extract node probabilities in efficient ways. They are broadly classified in terms of either *exact* or *approximate* inference methods, on the basis of exhaustive or heuristic resolution of *marginalization* of Conditional Probability Density (CPD) functions. By *marginalization*, it is indicated the operation of manipulation of the JPD in order to *eliminate* the dependence on one or more variables, in order to obtain a probability function of a *subset* of variables. The manipulation consist in a *sum* of all probabilities values of the JPD, for each possible value of the variable to be *kept*.

Example 4. Let us consider a JPD $\mathbf{P}_{\langle X_1,X_2\rangle}$ of two variables binary X_1 and X_2. In particular, let be $\mathbf{P}_{X_1,X_2}(X_1,X_2) = 0.45$, $\mathbf{P}_{X_1,X_2}(X_1,\bar{X_2}) = 0.15$, $\mathbf{P}_{X_1,X_2}(\bar{X_1},X_2) = 0.35$ and $\mathbf{P}_{X_1,X_2}(\bar{X_1},\bar{X_2}) = 0.05$.

Marginalization on X_1 produces the $\mathbf{P}_{X_2}$: $\mathbf{P}_{X_2}(X_2) = 0.45 + 0.35 = 0.8$ and $\mathbf{P}_{X_2}(\bar{X}_2) = 0.15 + 0.05 = 0.2$; marginalization on X_2 produces the $\mathbf{P}_{X_1}$: $\mathbf{P}_{X_1}(X_1) = 0.45 + 0.15 = 0.6$ and $\mathbf{P}_{X_1}(\bar{X}_1) = 0.35 + 0.05 = 0.4$.

It is worth noting that all $\mathbf{P}$ functions sum to the unity, *before* and *after* marginalization.

◇

One of the first algorithm for *exact* inference was proposed in the 1980s, known as *polytree* algorithm [KP83], is based on the concept of *message propagation*. A *message* is a local marginalization of the probability distribution, starting from parent-less nodes, and progressively propagated to children. While begin exact and having with a complexity polynomial in the number of nodes, this algorithm cannot be applied for networks having at least one node with multiple parents. Same author extended this exact inference algorithm for multiple-connected networks, dubbed *loop cutset conditioning* [Pea86]. This approach requires to instantiate a family of transformed *single-parent* networks, derived from the original one, for each multi-parent node contained in the graph. Each instance is resolved with the *polytree* algorithm and inference of the multi-parent network is computed by weighting. The complexity of this approach grows with the number of multiple-connection present in the graph. A similar approach, *variable elimination*, infers the probability value by the elimination other variables one by one by summing them out. The complexity of this algorithm depends on the number of multiplications and summations performed, depending on the the elimination order. Unfortunately, identification of minimum cost elimination order has been proven to be NP-hard.

Another popular exact inference algorithm is clique-tree propagation algorithm [LS88], also known as *clustering* or *Junction Tree* algorithm. It is a two-steps approach, which first transforms a multiply connected network into a clique tree (by clustering the triangulated moral graph [LS98] obtained from the BBN), then it performs message propagation over the clique tree. Efficient for sparse networks, the Joint Tree algorithm still can be extremely slow in dense graphs, since its complexity is exponential in the size of the largest clique of the transformed undirected graph. Many similar approaches to solve inference on clustering structures exist, and the most known in [SS08, JLO90, MJ98].

For our purposes, we require a light algorithm in terms of both:

i) reduced execution time, for a group of network updates needs to be performed whenever new evidence is available, and

ii) storage space, for our two-level structure is characterized by an high fan-in in test nodes inducing an exponential growth for CPD tables size.

An interesting approach, matching these requirements and fitting our fault causal relationship modeling is the *quickscore* algorithm, for NOR networks, described in [HL90]. The algorithm handles explicitly the independent concurrency of a test FAIL condition that is induced by component faults; this reduces the overall redundancy into inference propagation equations and it is possible to extract shared terms among different variables evaluations. As it is detailed in the referenced paper (notation is adapted to our case), we briefly recall probabilities equations for component and test nodes.

Given the set $\mathcal{C}$ of all components, and $\mathcal{T}^P$ and $\mathcal{T}^F$ respectively the sets of observed tests with PASS and FAIL outcomes, it computes:

$$\mathbf{P}(\mathcal{T}^P) = \prod_{\mathbf{c}_x \in \mathcal{C}} ((\prod_{\mathbf{t}_{zp} \in \mathcal{T}^P} \mathbf{P}(\mathbf{t}_{zp}|\mathbf{c}_x))(1 - \mathbf{Pc}_x) + (\mathbf{Pc}_x)) \tag{2.4}$$

$$\mathbf{P}(\mathcal{T}^F) = \sum_{\mathcal{T}' \in 2^{\mathcal{T}^F}} (-1)^{\mathcal{T}'} \prod_{\mathbf{c}_x \in \mathcal{C}} ((\prod_{\mathbf{t}_{zf} \in \mathcal{T}'} \mathbf{P}(\mathbf{t}_{zf}|\mathbf{c}_x))(1-\mathbf{Pc}_x)+(\mathbf{Pc}_x)) \tag{2.5}$$

which combines in a unique equation:

$$\mathbf{P}(\mathcal{T}^P, \mathcal{T}^F) = \sum_{\mathcal{T}' \in 2^{\mathcal{T}^F}} (-1)^{\mathcal{T}'} \prod_{\mathbf{c}_x \in \mathcal{C}} ((\prod_{\mathbf{t}_z \in (\mathcal{T}' \bigcup \mathcal{T}^F)} \mathbf{P}(\mathbf{t}_z|\mathbf{c}_x))(1-\mathbf{Pc}_x)+(\mathbf{Pc}_x)) \tag{2.6}$$

as the probability of observed test outcomes; then, for each component, it computes it probability to be *fault-free* as:

$$\mathbf{P}(\mathbf{c}_x = \mathbf{F}|\mathcal{T}^P, \mathcal{T}^F) = \frac{\mathbf{P}(\mathbf{c}_x, \mathcal{T}^P, \mathcal{T}^F)}{\mathbf{P}(\mathcal{T}^P, \mathcal{T}^F)} \tag{2.7}$$

where $\mathbf{P}(\mathbf{c}_x, \mathcal{T}^P, \mathcal{T}^F$ is obtained from Equation 2.6 replacing all terms $\mathbf{P}(\mathbf{t}_z|\mathbf{c}_x)$ with 1.

We adapted this approach, characterized by strongly reduced storage requirements, without the need for an explicit CPD representation. Indeed, in Equation 2.6 and 2.7 all contributions can be found either in the CTM or among *a-priori* components probability. Algorithm complexity is *constant* with respect to negative evidences (test with PASS outcomes), while exponential with the number $n_F = |\mathcal{T}^F|$ of positive evidences (tests with FAIL outcome), i.e. $o(2^{n_F})$.

Implementation For a CTM containing n_c components and n_t tests, two data structures are to kept all contributions required for the *quick-score* algorithm [HL90]: a $n_c \times 1$ vector, cumulating contributions for PASS test outcomes and a $n_c \times 2^{|\mathcal{T}^F|}$ matrix cumulating contributions for FAIL test outcomes.

Both structures are initialized setting all entries to 1. Two distinct operations are applied for new PASS and FAIL observations. Whenever a PASS outcome is observed, each entry of the PASS-vector is updated multiplying with the $\mathbf{P}(\mathbf{t}_z = \text{PASS}|\mathbf{c}_x = \text{faulty})$ referred to the corresponding component $\mathbf{c}_x$. For instance, Table 2.2 shows the evolution of the vector, from initialization, for the sequence of observations $\mathbf{t}_{z1} = \text{PASS}$, $\mathbf{t}_{z2} = \text{PASS}$, $\mathbf{t}_{z3} = \text{PASS}$. Whenever a FAIL outcome is observed, the entire content of the FAIL-matrix is duplicated; each new row is multiplied with the $\mathbf{P}(\mathbf{t}_z = \text{PASS}|\mathbf{c}_x = \text{faulty})$ referred to the corresponding component $\mathbf{c}_x$. For instance, Table 2.2 shows the evolution of the matrix, from initialization, for the sequence of observations $\mathbf{t}_{z4} = \text{FAIL}$, $\mathbf{t}_{z5} = \text{PASS}$. Terms $\mathbf{P}(\mathbf{t}_z|\mathbf{c}_x)$ in Table 2.2 and 2.3 are skipped for component-test pair $\langle \mathbf{c}_x, \mathbf{t}_z \rangle$ with *null* coverage.

The data structures presented contain all partial contributions necessary to compute inference through Equation 2.6. In particular, the equation denominator is calculated *traversing* all rows of Table 2.3; for each component, the FAIL-matrix entry is multiplied with corresponding entry in the PASS-vector (depicted in Table 2.2), and it is weighted with the *a-priori* probabilities. Row contributions, relative to each component, are multiplied together, and they are added or subtracted according to the number of FAIL observed in current Table 2.3 row. Inference for a specific component $\mathbf{c}_x$, according to Equation 2.7, is computed with the same algorithm *skipping* the column corresponding to $\mathbf{c}_x$.

2.2.3 Reinforcing single-fault hypothesis

In a BBN containing n_c component nodes, their *a-posteriori probabilities* evolve independently as long as new observations (test outcomes) are available after test execution. Therefore, the number of components identified as *faulty* by the BBN ranges potentially from 0 to n_c, spanning all 2^{n_c} possible diagnostic conclusions regarding components faulty states (**NF**: non faulty, **F**: faulty). If necessary, it is possible to set an explicit constraint on the BBN diagnosis conclusions introducing another topology of nodes in the network.

Table 2.2: Quickscore algorithm: PASS contributions vector.

	$\mathbf{c}_1$	$\mathbf{c}_2$	...	$\mathbf{c}_{n_c}$
(a)	$\emptyset$	1	1	1
(b)	$\mathbf{t}_{z1} = \mathbf{P}$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_2)$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_n)$
(c)	$\mathbf{t}_{z1} = \mathbf{P}$, $\mathbf{t}_{z2} = \mathbf{P}$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_1) \cdot \mathbf{P}(\mathbf{t}_{z2}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_2) \cdot \mathbf{P}(\mathbf{t}_{z2}\|\mathbf{c}_2)$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_n) \cdot \mathbf{P}(\mathbf{t}_{z2}\|\mathbf{c}_n)$
(d)	$\mathbf{t}_{z1} = \mathbf{P}$, $\mathbf{t}_{z2} = \mathbf{P}$, $\mathbf{t}_{z3} = \mathbf{P}$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_1)\cdot \mathbf{P}(\mathbf{t}_{z2}\|\mathbf{c}_1) \cdot \mathbf{P}(\mathbf{t}_{z3}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_2)\cdot \mathbf{P}(\mathbf{t}_{z2}\|\mathbf{c}_2) \cdot \mathbf{P}(\mathbf{t}_{z3}\|\mathbf{c}_2)$ ...	$\mathbf{P}(\mathbf{t}_{z1}\|\mathbf{c}_n)\cdot \mathbf{P}(\mathbf{t}_{z2}\|\mathbf{c}_n) \cdot \mathbf{P}(\mathbf{t}_{z3}\|\mathbf{c}_n)$

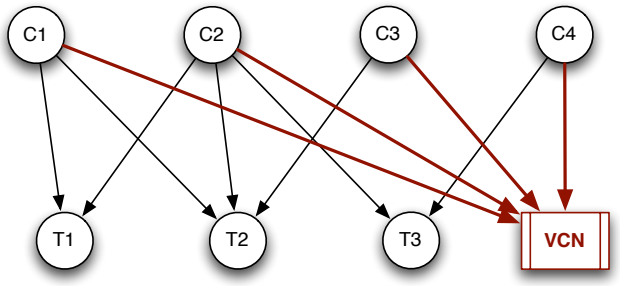


FIGURE 2.5: Sample BBN with VCN.

Let $\mathbf{xV}$ be a node of a BBN containing n_c component nodes. Let all component nodes to be part of the *fan-in* of node $\mathbf{xV}$. Let $\mathbf{P_{xV}}$ be the conditional probability distribution of $\mathbf{xV}$, having an entry for each one of the possible 2^{n_c} configurations $\mathcal{K}$ of faulty states (**NF**: non faulty, **F**: faulty) of component nodes.

We define $\mathbf{xV}$ a Valid Conclusion Node (VCN) when $\mathbf{P_{xV}}$ contains only binary entries; in particular, $\mathbf{P_{xV}}(\mathcal{K}) = 1$ if faulty configuration $\mathcal{K}$ contains exactly *one* faulty component, and $\mathbf{P_{xV}}(\mathcal{K}) = 0$ otherwise. A VCN is introduced into the BBN as depicted in Fig. 2.5.

VCNs can be considered to represent either an *observer* or a *controller* variables. In the former case, no information about the true

Table 2.3: Quickscore algorithm: FAIL contributions vector.

			$\mathbf{c}_1$	$\mathbf{c}_2$	...	$\mathbf{c}_{n_c}$
(a)	1:	$\emptyset$	1	1		1
(b)	1:	$\emptyset$	1	1		1
	2:	$\mathbf{t}_{z4} = \mathbf{F}$	$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_2)$		$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_n)$
(c)	1:	$\emptyset$	1	1		1
	2:	$\mathbf{t}_{z4} = \mathbf{F}$	$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_2)$		$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{cn})$
	3:	$\mathbf{t}_{z5} = \mathbf{F}$	$\mathbf{P}(\mathbf{t}_{z5}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z5}\|\mathbf{c}_2)$		$\mathbf{P}(\mathbf{t}_{z5}\|\mathbf{cn})$
	4:	$\mathbf{t}_{z4} = \mathbf{F}$, $\mathbf{t}_{z5} = \mathbf{F}$	$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_1) \cdot \mathbf{P}(\mathbf{t}_{z5}\|\mathbf{c}_1)$	$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_2) \cdot \mathbf{P}(\mathbf{t}_{z5}\|\mathbf{c}_2)$		$\mathbf{P}(\mathbf{t}_{z4}\|\mathbf{c}_n) \cdot \mathbf{P}(\mathbf{t}_{z5}\|\mathbf{c}_n)$
			...			

value of the binary value represented by the node is available, therefore the probability value of **xV** can be computed as for any other variable of the BBN. Because of the nature of its distribution $\mathbf{P}_{\mathbf{xV}}$, accepting only single-fault configurations, such probability computes a *credibility* measure of a single-fault diagnosis to be a correct diagnostic conclusion given the set of observed test outcomes.

In the latter case, the actual value of **xN** is known, and it is set to be *true*: such an observation on **xN** affects the probabilities of all component nodes. The presence of this observation introduces also a *semantic* change of the meaning of the probability associated with component nodes: for a given component node $\mathbf{c}_x$, $\mathbf{P}(\mathbf{c}_x)$ expresses the probability that the single-faulty component diagnosis, having $\mathbf{c}_x$ as faulty candidate, is true.

While interesting from a modeling point of view, the introduction of VCN suffers of the drawback of not being compatible with the inference algorithm proposed previously. In fact, such nodes fulfill the independence hypothesis for nOR decomposition. Therefore, to compute their probability values, or to solve the BBN when they are used as *controller* variables, it is necessary to use a more complex inference algorithm such as, for instance, a Junction Tree based algorithm [LS98], which we did not adopt in our framework for computational complexity reasons.

2.3 Four-layer BBNs

The BBN structure proposed in the previous Section 2.2 applies to a broad category of systems, while keeping a low complexity regarding the type and the amount of information required to test engineers to develop the model.

On one hand, model developers have a limited number of critical choices to make, namely the *decomposition* of the system under investigation in components and the *definition* of the *a-priori* failure rates; this limitation reduces the risk the model complexity to go out of control. On the other hand, test engineers are offered a large spectrum of possibility to describe interactions between components and diagnostic tests, and by exploiting the robustness of the BBN inference engine to tolerate *non-recurrent or local mistakes* potentially occurring during model development, they dispose of a detailed while simple tool to support automatic diagnostic investigations. Furthermore, the simplicity of the model is reflected in the possibility to implement an efficient inference engine, giving the opportunity of the BBN tool to be used in a quasi-real time fashion and making it suitable for industrial scenarios.

Unfortunately, the simplicity of the architecture proposed for the BBN carries a subtle side-effect, which is an intrinsic difficulty of describing specific correlations between events. In our diagnosis context this translates in the unfeasibility of forcing the model to consider as *impossible* some particular outcomes configurations (Example 5).

Example 5. Let us consider an excerpt of a system, as depicted in Figure 2.6 (a). The system is composed of a micro-processor $\mathbf{c}_1$, a memory composed of two banks (bank A $\mathbf{c}_2$, bank B $\mathbf{c}_3$), and a pair of bus interfaces $\mathbf{c}_4$ and $\mathbf{c}_5$ for each module. In the test-suite, two tests $\mathbf{t}_1$ and $\mathbf{c}_2$ are used to test respectively banks A and B, with a set of *write-read* loops. The BBN depicted in Figure 2.6(b) reports the CTM non-null coefficients, on the corresponding graph arc.

The failure is localized on a component ($\mathbf{c}_4$), which is stimulated by both tests; furthermore, the *coverage* is indicated as *low* because the functionalities of the bus interface involved in this specific *read-write* test are marginal, or in other words, the test is not designed to *detect* a large spectrum of possible failures affecting this specific component, being its target the verification of memory banks (A or B).

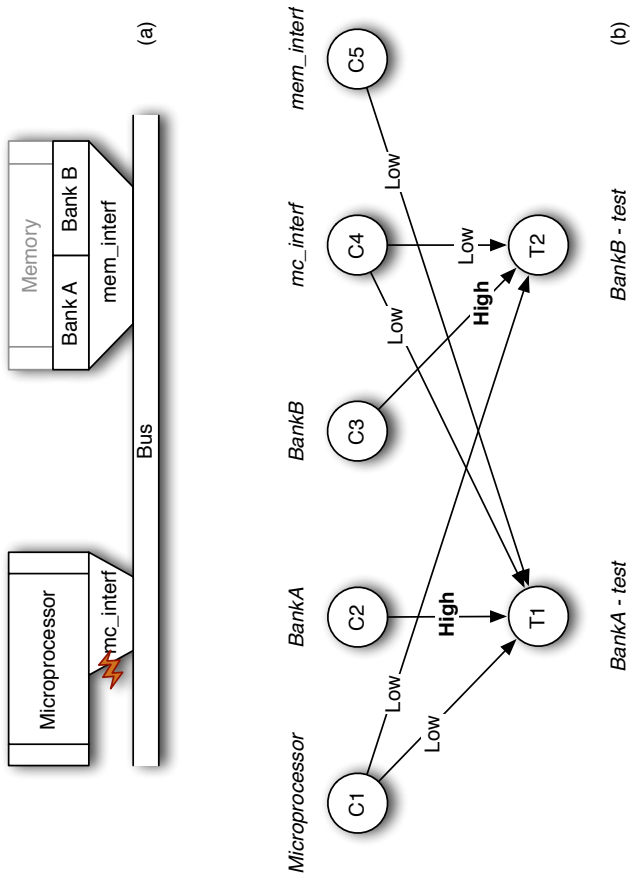


Figure 2.6: Microprocessor-memory system excerpt and BBN.

However, if we consider at the possible configurations of outcomes revealing a failure, we can observe that combinations $\langle \mathbf{t}_1 = \textsf{FAIL}, \mathbf{t}_2 = \textsf{PASS} \rangle$ and $\langle \mathbf{t}_1 = \textsf{PASS}, \mathbf{t}_2 = \textsf{FAIL} \rangle$ are valid *manifestations* of failures on banks A and B, but they reveal a bad modeling policy for failure in Figure 2.6 (a) because the functionality stimulated by the test is likely to be the same: the BBN proposed considers as valid outcomes those configurations, and this affects the valid estimation of probability through inference.

◇

According to definitions proposed in Section 1.2, we extend the model structure of BBN introducing a *4-layer* structure, renaming the nodes of the network according to the following:

- **Component** nodes: as in 2-layer BBN, they represent *logic* locations on the system under investigation; furthermore, partitions

of the system that are strictly and univocally related with *a-priori* probability of failure occurrence;

- **Subcomponent** nodes: each **component** of the BBN is linked with one or more subcomponents nodes; those are introduced in the BBN architecture to provide a better semantic to the model developer. With this nodes, it is possible to describe a *hierarchical* decomposition of each component in logic subblocks; furthermore subcomponents increases the resolution of the model developer to associate particular functionalities of each component to the tests of the test-suite, improving the quality of coverage and overcoming outcome constraints, e.g. the problems underlined in Example 5;
- **Operation** nodes: each **subcomponents** of the BBN support the computation of one or more operations, during the execution of tests; each operation is characterized by *locality*, in the sense that it receives information from and send information to other components. This family of nodes is introduced to decouple the concept of operation to the outbound visibility, wheter its execution result correspond or not to the expected-specified one;
- **Test** nodes: they are defined as collections of **operations**, providing a *cumulative* information about the success or failure of their execution, with the binary outcome PASS-FAIL.

All nodes of the 4-layer BBN represent binary random variables. In particular, *subcomponent* nodes possible values are **F** (faulty), **NF** (non-faulty), whether the node contains system failure or not. *Operation* variable assume the same values of *test* nodes (PASS, FAIL), representing respectively the condition where the operation produces correct or wrong results. Figure 2.7 depicts the element of a generic 4-layer BBN, indicating correspondences with the 2-layer model described in Section 2.2.

A-priori probability in 4-layer networks *A-priori* probability values are attributed to subcomponents, instead of components as in 2-layer BBN. Component nodes variables are computed according to a *deterministic* conditional distributions: a component is considered *faulty* if at least one of its subcomponents is *faulty*, and it is considered *non-faulty* otherwise. For instance, component $\mathbf{c}_1$ in 2.7 conditional probability distribution $\mathbf{P}(\mathbf{c}_1|\mathbf{sc}_{11}, \mathbf{sc}_{12})$ is depicted in the following table:

$\mathbf{c}_1$	**NF,NF**	**NF,F**	**F,NF**	**F,F**
NF	1	0	0	0
F	0	1	1	1

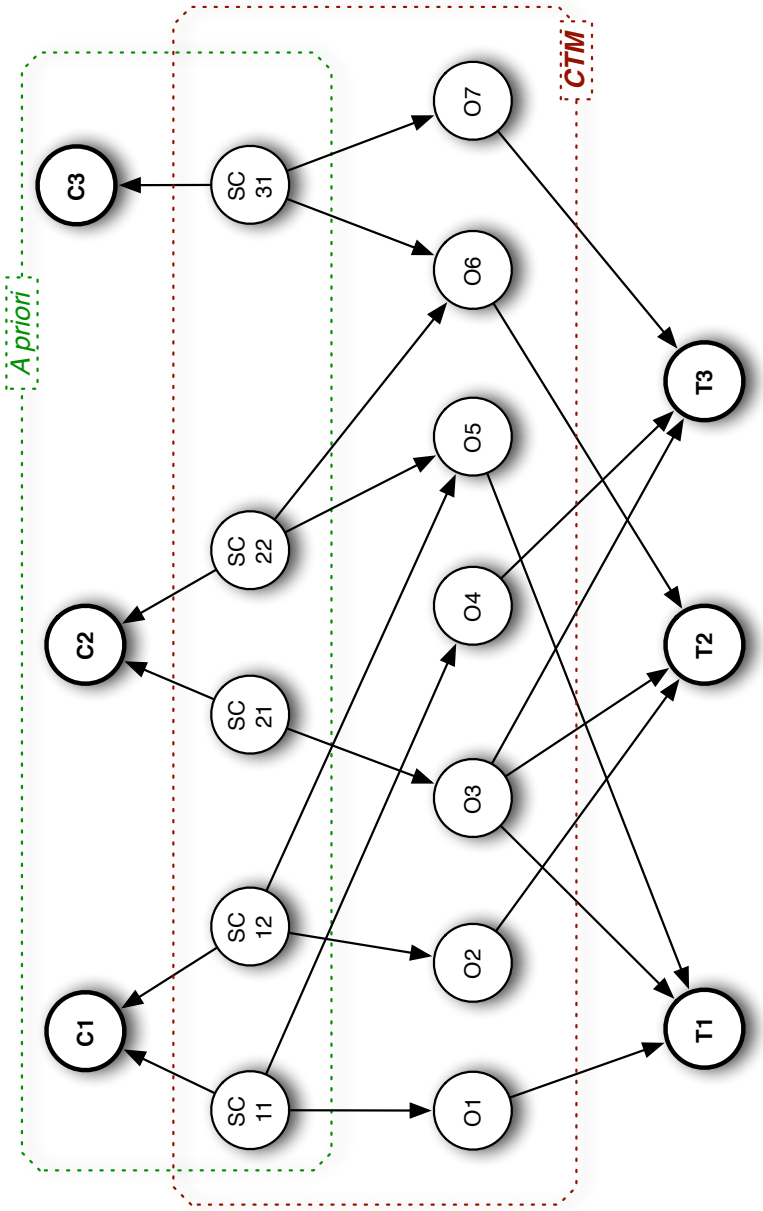


Figure 2.7: Example of 4-layer BBN.

When a component has exactly one subcomponent, they represent a pair of variable whose value will be always *equivalent*. Maintaining the redundancy increases the regularity of the network, which is an important property for both modeling and inference.

Being $\mathbf{P}(\mathbf{c}_1|\mathbf{sc}_{11}, \mathbf{sc}_{12})$ a deterministic function, the marginalization of components probabilities at the beginning of each test session, when no test outcome is available, correspond to the *sum* of all *a-priori* probabilities of its subcomponents. The procedure to be followed by test engineer to attribute *subcomponents a-priori probabilities*, once the failure rate of all components has been established by data-sheet inspection, is to *distribute* the probability to be *faulty* of each subcomponents proportionally to an evaluation of failure likelihood within each component.

Example 6. Let us consider the BBN shown in Figure 2.7. From datasheets inspection, *failure* probabilities of components $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ are respectively 50%, 40%, 10%. The probability of localization of a failure, within each component, is uniform for all subcomponents.

Then, the *a-priori* probability of subcomponents is 25% for subcomponents $\mathbf{sc}_{11}$ and $\mathbf{sc}_{12}$, 20% for subcomponents $\mathbf{sc}_{21}$ and $\mathbf{sc}_{22}$, and 10% for subcomponent $\mathbf{sc}_{31}$. In this example, $\mathbf{c}_3$ and $\mathbf{sc}_{31}$ are redundant.

◇

Test outcomes Test nodes have the role to collect the status of the single operation involved in their execution and to combine it in a cumulative, binary value (PASS, FAIL), representing the only observation available for system diagnosis. In order to avoid an excessive number of parameters to describe the BBN model, we adopt a deterministic function to describe the relation between a test and the operations it is composed of.

We derive from *Fault-Tree Analysis* [MR04b, NUR] formalism the concepts of OR gates, AND gates and K-OF-N gates and we translate them in conditional probability distribution to associate them with BBN nodes. *Fault-Trees* are graphical tools, especially used in Reliability contexts, designed to provide a visual representation of cause-effects relations between events. Basically, a *Fault-Tree* structure describes relations between events following an extended boolean fashion. Let us consider a set of N binary events $\mathcal{E} = \langle e_1, e_2, e_3, \ldots, e_N \rangle$ and a target event $\mathbf{e_T}$. An OR relation is used to describe a scenario where, when *at least* one event in $\mathcal{E}$ occur, it causes target event $\mathbf{e_T}$ to occur also (Figure 2.8(a)). An AND relation, instead, is used to describe the opposite scenario, where *all events* must have occurred simultaneously in order to cause the target event $\mathbf{e_T}$ to occur (Figure 2.8(b)). K-OF-N relations

describe the intermediate scenario where $K < N$ events are sufficient to *trigger* target event $\mathbf{e_T}$ (Figure 2.9).

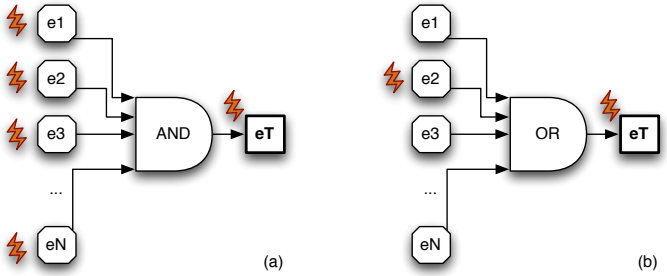


Figure 2.8: OR (a) and AND (b) gates.

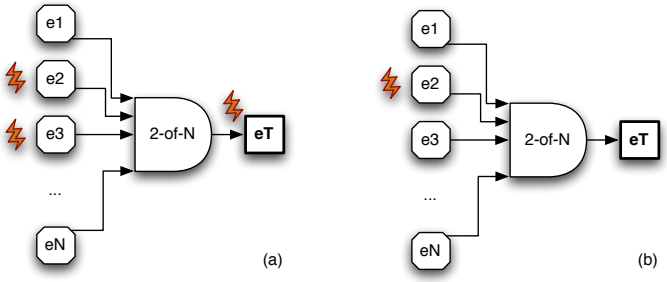


Figure 2.9: Activated (a) and not-activated (b) K-of-N gate ($K = 2$).

In 4-layer BBN, we translate the occurrence of causing events with the failure of an operation to compute correctly its results, and the triggered event as the FAIL outcome of a test. Conditional probability functions $\mathbf{P}(\mathbf{t}_z|\mathbf{o}_{k1}, \mathbf{o}_{k2}, \mathbf{o}_{k3})$ of a test $\mathbf{t}_z$ involving 3 operations $\mathbf{o}_{k1}, \mathbf{o}_{k2}, \mathbf{o}_{k3}$ are reported in Table 2.4 respectively for the OR, AND and 2-of-3 cases. The choice adopted to represent operation-test relations in our diagnostic framework is the OR relation.

CTM During the definition of the system model, the *coverage* relation is transferred from component-test pairs in 2-layer BBN to subcomponent - operation pairs in 4-layer BBN. In fact, 2-layer graphs can be considered as the extreme case of 4-layer ones, where each component has exactly one subcomponent and each test requires exactly one operation to complete. However, the latter structure provide enough semantic to discriminate between situation where the execution of two tests is

Table 2.4: OR, AND and 2-OF-3 conditional probability.

$\mathbf{P}(\mathbf{t}_z \mid \mathbf{o}_{k1}, \mathbf{o}_{k2}, \mathbf{o}_{k3})$	PPP	PPF	PFP	PFF	FPP	FPF	FFP	FFF
PASS	1	0	0	0	0	0	0	0
FAIL	0	1	1	1	1	1	1	1

$\mathbf{P}(\mathbf{t}_z \mid \mathbf{o}_{k1}, \mathbf{o}_{k2}, \mathbf{o}_{k3})$	PPP	PPF	PFP	PFF	FPP	FPF	FFP	FFF
PASS	1	1	1	1	1	1	1	0
FAIL	0	0	0	0	0	0	0	1

$\mathbf{P}(\mathbf{t}_z \mid \mathbf{o}_{k1}, \mathbf{o}_{k2}, \mathbf{o}_{k3})$	PPP	PPF	PFP	PFF	FPP	FPF	FFP	FFF
PASS	1	1	1	0	1	0	0	0
FAIL	0	0	0	1	0	1	1	1

characterized by full independence, and a scenario where *shared operations* would introduce an incoherence of a system description relying on component-test pair-based model only.

2.3.1 Inference for four-layer BBNs

The structure of 4-layer BBNs proposed in this section is even more complex than 2-layer structures proposed in Section 2.2; *exact* inference on this network result is solution which is not practicable. We implemented an *approximate* (*stochastic*) approach for inference evaluation on a 4-layer BBN.

Stochastic simulation algorithms, also called stochastic sampling or Monte Carlo algorithms, are the most well known approach for approximate inference. Those methods generate a set of *samples* of the network, attributing a true random value to each variable according to the JPD of the model. Then, they approximate probabilities of required variables computing the relative frequencies of appearances in the samples. The accuracy (and the complexity) of probabilities evaluation is as high as the number of generated samples, while it is independent of the *structure* of the network. Stochastic algorithms can be divided into two broad categories: *importance sampling algorithms* and Markov Chain Monte Carlo (MCMC) methods.

The simplest sampling algorithm is *probabilistic sampling*, described in [Hen88]. This is also known as *forward sampling* because it casts a random value for each variable only when all its preceding variables (in the DAG) have been sampled, exploiting the conditional probability function. Frequencies are evaluating through *counters* instantiated at each variables. The approach *discards* all samples which are inconsistent

with observed true values; therefore, as the number of potentially valid samples decreases exponentially with the number of observed nodes, the accuracy of the approach drops when the set of available observations is large.

Likelihood Weighting (LW) [DC93] has been conceived to tackle this limitation, which is prohibitive when dealing with large networks. Instead of counting/discarding a sample whenever is consistent/incosistent with the values of the observed nodes, the counter is updated *weighting* the sample with the likelihood of observation, conditional on the samples.

From accuracy point of view, Likelihood Weighting converges faster then logic sampling, but, as all stochastic sampling algorithms, it shows an extreme long converges time when *unlikely events* are observed. Improvements to this algorithm have been proposed, for instance in [DL97] to tackle specific topologies of networks or reduce computational efforts.

Even if more efficient techniques were reported in recent literature [Hec08,GH02], we focused our effort on an implementation based on Likelihood Weighting, as a good tradeoff between *accuracy*, *computational effort*, *real-time* constraints for the adaptive methodology (Section 2.4) and implementation complexity.

2.4 incremental Automatic Functional Fault Detective (AF2D)

2.4.1 Adaptive diagnosis

incremental Automatic Functional Fault Detective (AF2D) is a methodology we proposed in [ABS+09]. The flow of AF2D is summarized in Figure 2.10.

Offline operations The entry point of the methodology consists of two main pieces of information: i) the structure of the board under investigation (in terms of components or/and functionalities, which we will refer to as simply components) and ii) the set of tests and their relation with the components.

While the former can be directly derived from the specification of the system, the latter requires the intervention of the test engineers' team, whose role is twofold: they have to select, among all possible tests, the subset they deem representative to verify the functionality of the board, and they have to specify the qualitative indication of the relation between

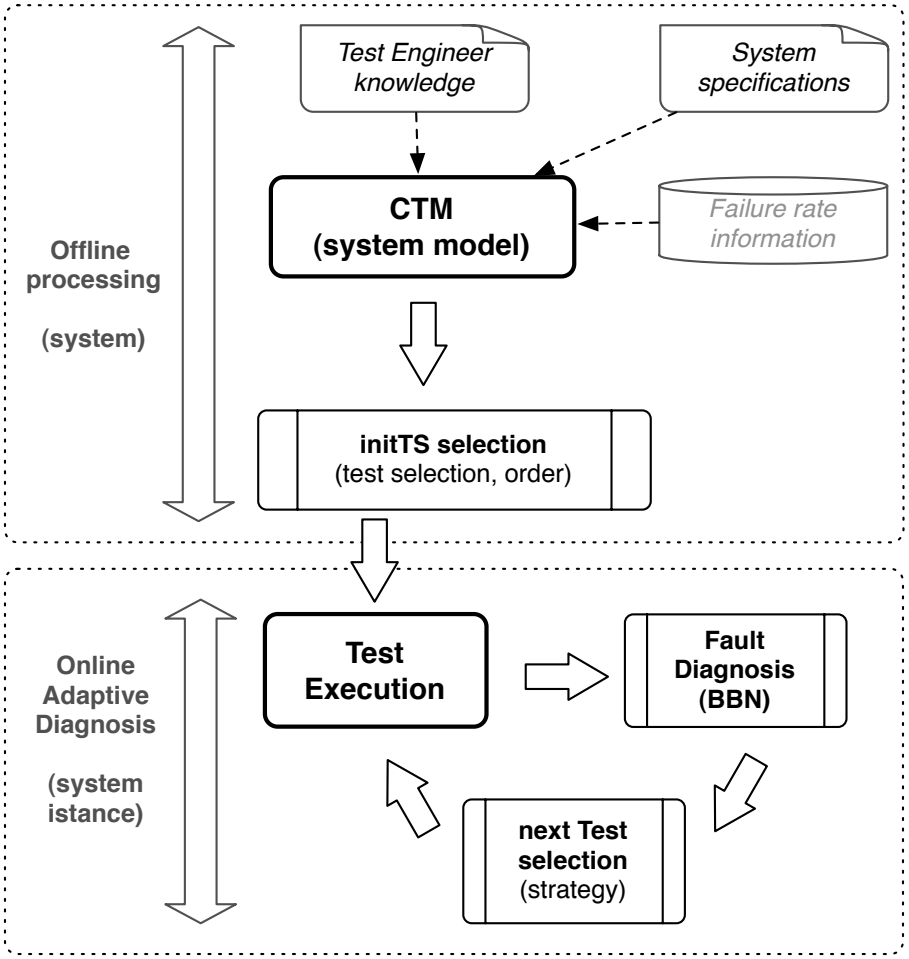


FIGURE 2.10: AF2D flow.

components and tests (the observability of the component's behavior on the test output), using the *coverage labels* presented in Section 2.2. The output of the test engineers' team activity is *the model of the board for the diagnosis*, the CTM.

Figure 2.4 (a) represents an hypothetic board, indicating the coverage of a test $\mathbf{t}_1$ involving components $\mathbf{c}_1$, $\mathbf{c}_4$, $\mathbf{c}_7$, $\mathbf{c}_9$, $\mathbf{c}_{10}$ and $\mathbf{c}_{11}$.

The first stage of the methodology is composed of two parallel activities: the transformation of the CTM into the corresponding BBN model, deriving the conditional probabilities (e.g., $\mathbf{P}(\mathbf{t}_1|\mathbf{c}_1, \mathbf{c}_4, \mathbf{c}_7, \mathbf{c}_9, \mathbf{c}_{10}, \mathbf{c}_{11})$ with reference to the example of Figure 2.4) and the identification of the subset of tests to be used initially.

This second element represents the initial test subset, initTS, that verifies all components; this first block of tests is used to collect enough information about the system under test, avoiding a situation where a component never gets verified. Chapter 3 discuss the problem of the identification of *optimal* initTSs: this set is identified by considering both test-priority and level of coverage; in general, since the level of coverage and the covered components are different from test to test, an optimal initTS constitutes a trade-off between the cost of tests included in the set and the global quality of the coverage.

Online operations Operations described in previous paragraph are executed *once*, using exclusively the information contained in the system model CTM.

In the second step, information collected after the initial test sequence is analyzed and further tests are executed if no tests have failed. In fact, if no test fails, no information can be used to identify a set of faulty candidates and it is necessary to improve the covering of the board adding further tests (one at time). Since the absence of possible candidates is the driver of this step, tests are chosen to increase the coverage factor associated with the initTS.

Once at least an executed test fails, the core of adaptive test selection starts. In this step, the effects of the execution of each remaining test are *simulated* ahead, one step at a time using the BBN engine. Tab. 2.5 depicts the following scenario: tests $\mathbf{t}_{02}$, $\mathbf{t}_{03}$, $\mathbf{t}_{04}$, $\mathbf{t}_{06}$, $\mathbf{t}_{08}$ and $\mathbf{t}_{09}$ have been executed, with partial syndrome -PFP-P-PP; $\mathbf{t}_{01}$, $\mathbf{t}_{05}$ and $\mathbf{t}_{07}$ have not been executed yet, and an evaluation of the information they would provided is computed in the matrix of simulation results.

The reported information includes the absolute probability for each component of being faulty with respect to each test $\mathbf{t}_i$ and its possible outcome (PASS- P or FAIL- F), completed by the probability of the test

to have such an outcome; all this values are computed by the BBN engine based on the availabele outcomes and the model. For example, the execution of test $\mathbf{t}_{01}$ with a PASS outcome, would indict component $\mathbf{c}_{10}$, with a 97% probability of being faulty (3% of being fault-free), $\mathbf{c}_{11}$, with a 93%, $\mathbf{c}_8$, with a 2%, etc. Furthermore, the probability of the test to PASS is 84%, 16% it FAILs, giving the available information on the already executed tests. Do note that P($\mathbf{t}_i$=PASS) + P($\mathbf{t}_i$=FAIL) = 100%, while all the other terms are not normalized.

All components appearing in one of the tests that have failed constitute the Faulty Candidate Components (FCC) set, that is used in conjunction with the derived matrix of simulation results to determine what to do in the next step. Chapter 4 discuss the cost functions that have been defined to compute a final value associated with each not-yet executed test, with the aim to reduce the cardinality of the FCC set, also preferring those tests that better differentiate the candidates' probability of being faulty.

A detailed description of the steps of AF2D are summarized as a pseudo-code in Figure 2.11, where the flow of the methodology is outlined.

Some considerations are worth discussing on the preliminary scalar function introduced in [ABS+09] to select the next test. First of all, the FCC set includes only components with a probability of being faulty greater than 0, a value used to rank them. The second consideration concerns the fact that the function has been defined to provide a ranking for each single test, aggregating information from both outcomes, PASS and FAIL. Another important issue concerns the distribution of the probabilities of being faulty among the components in the PFC set. More precisely, given two situations FCC_1 = $\{(\mathbf{c}_h : 43\%); (\mathbf{c}_i : 10\%); (\mathbf{c}_j : 30\%); (\mathbf{c}_k : 43\%); (\mathbf{c}_l : 5\%)\}$ and FCC_2 = $\{(\mathbf{c}_h : 10\%); (\mathbf{c}_i : 90\%); (\mathbf{c}_k : 25\%); (\mathbf{c}_l : 5\%)\}$, the latter is more interesting since it focuses the attention on a single component ($\mathbf{c}_i$) rather than two ($\mathbf{c}_h$ and $\mathbf{c}_k$). To model this preference, the standard deviation of the probability values is used, able to express the pursued situation of small cardinality of FCC sets, with differentiated probabilities. The last contribution to the scalar function is the probability of the outcome of each test, used to weight the results provided by each possible future syndrome.

Since the incremental approach aims at limiting the number of tests to be executed to make a diagnosis, the methodology needs to include a strategy to determine whether to proceed with the next test or not. Before executing the selected next test, the matrix of the simulation results is analyzed to evaluate the benefits of the potential future syndrome in computing the final ranking of the possible faulty candidates against the

```
Input: System Model CTM, Test Suite 𝒯
Output: FCC (Possible Faulty Candidates).
Procedure FindCandidates(CTM,T)
{
 1  initTS = Select_Initial_Tests(T)
 2  T = T – initTS
 3  PartialSyndrome = Execute_Tests(initTS)
 4  PFC = Compute_Possible_Faulty_Candidates(partialSyndrome)
 5  while (PFC = ∅))
 6     T0 = Select_Add_Initial_Tests(T)
 7     initTS = initTS + T0
 8     T = T – initTS
 9     PartialSyndrome = Execute_Tests(initTS)
10     PFC = Compute_Possible_Faulty_Candidates(PartialSyndrome)
11  end while
12  while ((T ≠ ∅) and (not STOP )))
13     Matrix = Simulate_Tests(T,PartialSyndrome)
14     Ti = Select_Next_Test(Matrix)
15     STOP = Analize_STOP_Condition(Matrix)
16     if(not STOP)
17        T = T – T0
18        PartialSyndrome = PartialSyndrome + Execute_Tests(Ti)
19        PFC = Compute_Possible_Faulty_Candidates(PartialSyndrome)
20     end if
21  end while
22  output PFC
}
```

Figure 2.11: The incremental diagnosis methodology flow.

confidence of the current results. For a preliminary assessment of the methodology, the *stop condition* in [ABS+09] has been associated with a situation where the additional tests would only modify the relative probability of the FCC elements while maintaining the same order.

2.4.2 Framework and implementation

A framework has been designed to implement the proposed approach, to cope with both algorithm complexity and computation efficiency. A multi-layered architecture has been adopted; this is based on a bottom-up partitioning into an inference engine, a network state manager and an intelligent explorer, as it is shown in Figure 2.12. Each layer targets a specific task implementing, in turn, a set of optimal strategies focusing on overall execution time reduction.

At the lowest level, the **Inference Engine** provides a uniform API exposing methods to receive a set of test outcomes, to compute the probability of all *component* nodes, as long as the probability of *non-executed tests* nodes. Furthermore, the API is designed to discriminate *real* executed test outcomes, ideally retrieved from an ATE running a diagnostic process of a real system, and *virtual* test outcomes, which are simply generated by the framework to explore different strategies to continue the test session; this distinction is important to improve the efficiency of the engine because the simulation of remaining tests is an extremely *regular* exploration, and the code can be suggested to take into account intermediate results and save computational time. The engine interface and inference core has been coded in *plain C language* for performance reasons; two cores have been implemented, an *exact inference* algorithm (based on *quickscore* described in Section 2.2) and a stochastic inference engine, designed to tackle the inference on 4-layer BBN (Section 2.3).

The intermediate layer is introduced for **BBN state management**, to decouple the superior adaptive test sequencer logic from the inference engine, for some reasons. On one hand, this is necessary to provide the transparency with respect to the specific algorithm used for probability evaluation, especially for what concerns the translation of the CTM model in a valid BBN structure; on the other hand, it is not strictly necessary to recompute the probabilities values for components and tests more than once, when a *syndrome* containing a specific set of test outcomes first appear; for later reference, the probability set can be preferably loaded from an *ad-hoc syndrome database*. For instance, for at any step of the diagnosis, the AF2D Tool explores all possible *next test* exe-

cution scenarios, the actual probabilities after a new test outcome insertion are already available within such dataset, and another computation would be unnecessary and inefficient; furthermore, database query are faster than inference engine execution, and this makes the AF2D Tool more suitable for real-time test sequencing. Same considerations apply for *optimal strategy* identification: given an unchanged BBN model, the best test selection is the same for any occurrence of a given syndrome. The intermediate layer is implemented in Python [Pyta] and it interacts with the lower inference engine API through a Python C/API [Pytb] encapsulating the C code. *Syndromes* and *strategies* databases implementation is currently based on SQLite [SQL], not requiring advanced DBMS features as user authentication or concurrent query execution, and they are accessed by the BBN state management through a specific Python library.

The highest layer, **AFD Engine**, is in charge of implementing AI logic, by functionality integration of Python modules. In this layer, the language is chosen to be used as a *glue* logic, coordinating several task-specific problems (e.g., the initial test set definition, the stop condition evaluation); while some simpler tasks are performed directly (stop condition, Chapter 4), others are integrated in C/C++ libraries, opportunely wrapped (as ILP optimization, in Section 3.2). Other support modules, while not directly involved in the adaptive methodology, are integrated at this level: for instance, this is the case of the approach for test-set suite diagnosis resolution evaluation (and extension) described in Section 5.3. Finally, the layer is also in charge of interaction with other external modules designed for the Tool users. In Figure 2.10 depicts an spreadsheet module (implemented in MS Excel, for compatibility with Agilent Fault Detective Tool [Agi04]) for the completion of new CTM modules (Figure 2.14), a telnet-like console for user interaction during the adaptive test sessions (Figure 2.13), and a Web-oriented interface, conceived for remote deployment (Figure 2.15).

Table 2.5: Next step matrix of the simulation results.

Next Test	Possible Future Syndromes	Components' Probability to be faulty											Outcome Prob. %
		$\mathbf{c}_1$	$\mathbf{c}_2$	$\mathbf{c}_3$	$\mathbf{c}_4$	$\mathbf{c}_5$	$\mathbf{c}_6$	$\mathbf{c}_7$	$\mathbf{c}_8$	$\mathbf{c}_9$	$\mathbf{c}_{10}$	$\mathbf{c}_{11}$	
t_{01} = P	**P**PFP-P-PP	0	0	0	0	1	0	0	2	1	97	3	84
t_{01} = F	**F**PFP-P-PP	0	0	1	0	1	0	0	98	0	4	14	16
t_{05} = P	-PFP**P**P-PP	0	0	0	0	1	0	0	16	1	84	34	97
t_{05} = F	-PFP**F**P-PP	0	0	8	0	1	0	4	69	20	14	14	3
t_{07} = P	-PFP-P**P**PP	0	0	0	0	1	0	0	16	1	84	34	97
t_{07} = F	-PFP-P**F**PP	0	0	5	0	1	0	2	69	20	15	10	3

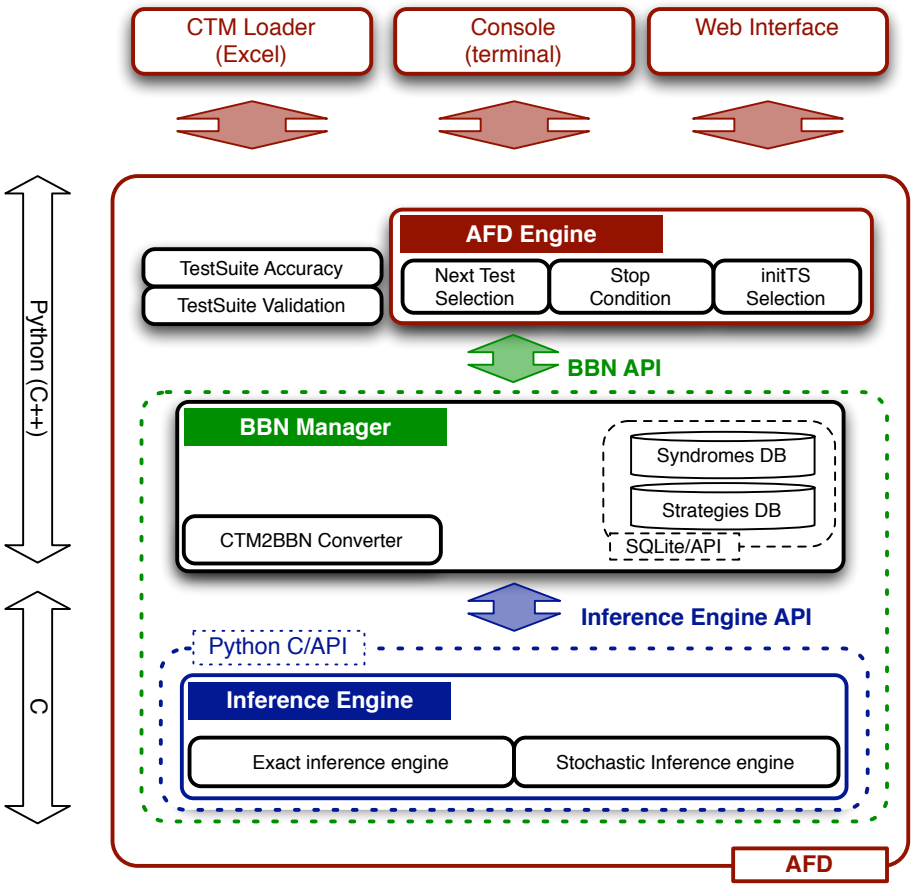


Figure 2.12: AF2D framework architecture.

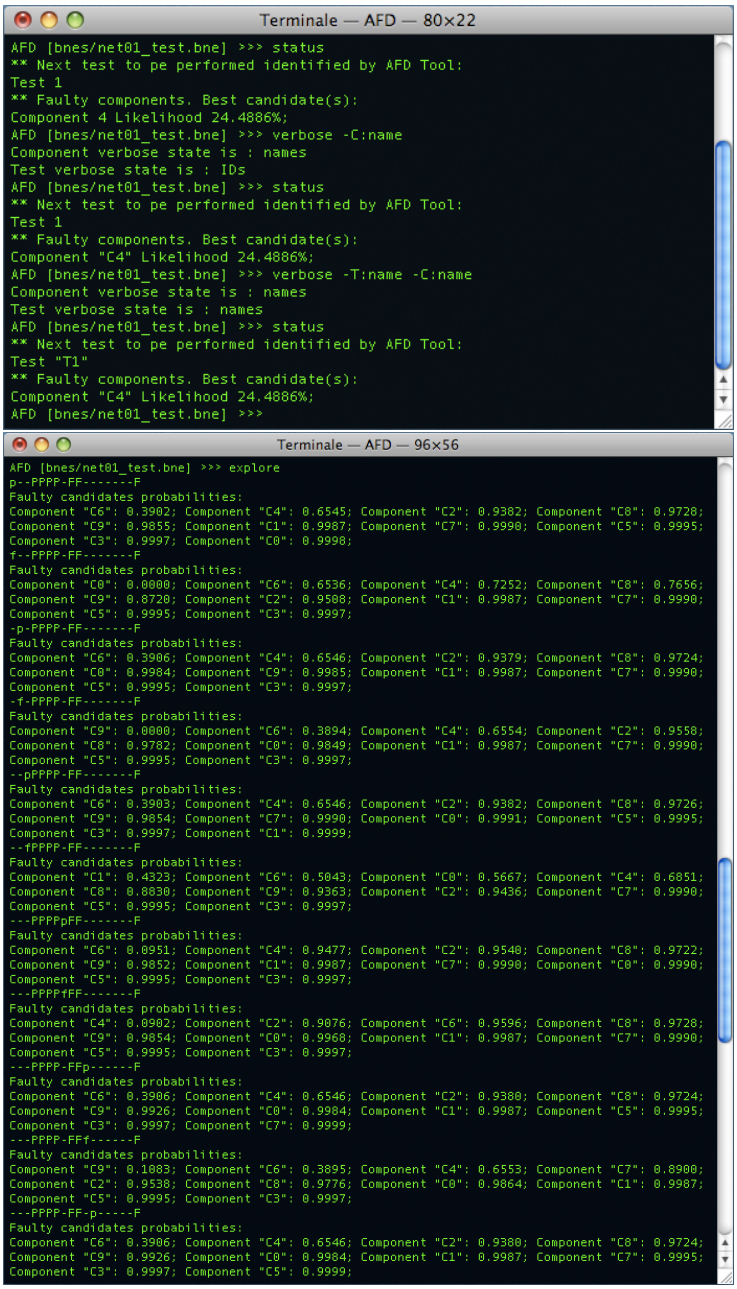


FIGURE 2.13: Console interface for AF2D Tool.

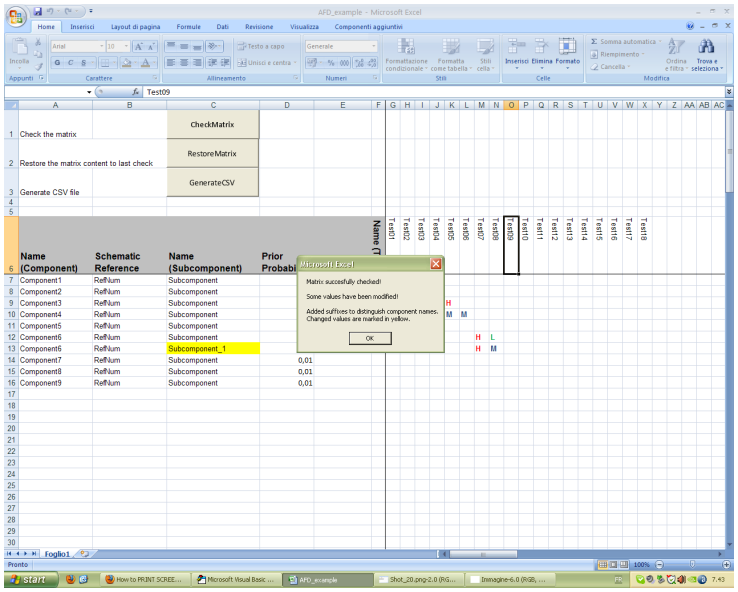


Figure 2.14: Spreadsheet interface for CTM editing.

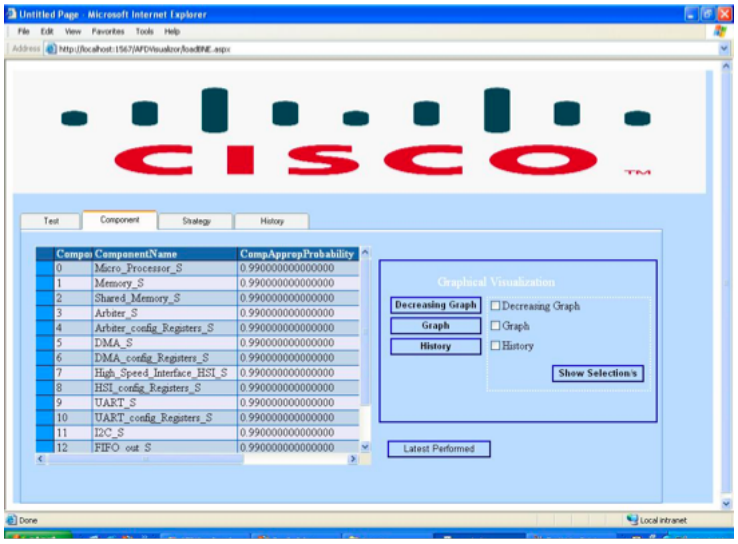


Figure 2.15: Web-based interface for AF2D Tool.

Introducing the process of system diagnosis in Section 1.2, we explained that one of the mot important goal of a diagnostic process is the *localization* of the cause of a misbehavior, failure, or fault, detected though its external manifestation (*syndrome*).

Policies taking place after the failure localization can be as different as the nature or the possible configurations of the system. In some scenarios, a system could be recovered from the last known *fault-free* state: this is the case when the localization of the misbehaving (faulty) component is precise enough to allow in-place heal by maintenance [BDJV+08b].

For other systems, instead, the nature of the device identified as the cause of the failure allows its substitution with another identical units [SS91a], in a process similar to system debug [ME02].

When the complexity of the system or its particular architecture prevent any kind of *fix* operations, the localization of the failure is important at least in order to implement a mitigation strategy through re-configuration (for instance, re-scheduling operations assigned to the faulty component to other units, or to spare ones). This last option, not strictly dependent on diagnosis, is to be supported by the system from design time.

In such contexts, diagnosis is conceived to take place off-line: monitoring the evolution of a system, an initial assumption of diagnosis is that at least one error has been recognized among the observable parameters, in order to start fault localization. For this purpose, specific

test suites should be designed aiming *explicitly* at diagnosis.

On one hand, at a *detection level*, a test is considered to be efficient if, in presence of a large number of potential failures, it shows output values which are different with respect to a fault-free response of the system. On the other hand, a *good diagnosis test* not only makes the presence of the fault observable, but it also supports a discrimination of the effects of a fault from those caused by others.

Clearly, test suites aiming at solving those two problems (respectively, *detection* versus *localization*) are inspired by different principles and they are consequently characterized by different properties. One of the most important properties is *test suite size*. In [PR10], author proposes a comparison of test suites sizes for combinational and sequential circuits of the ISCAS-89 benchmark suite: in general, diagnostic test suites are one order of magnitude larger for what concerns the number of involved test vectors. This is a problem both for storage space in BIST contexts [LC05], where the entire test suite needs to be archived within the chip under analysis, and for test selection in adaptive testing scenarios [Shu09], where the number of available options impacts on the complexity of the search algorithms. Furthermore, it has been observed that test suites designed for detection are capable to distinguish a large number of fault pairs, but larger suites are necessary to extend such capability to all faults [PR10].

Rather than fault coverage [ABF90], different metrics are used to evaluate the *quality* of a diagnostic test suite. In literature, several approaches have been proposed especially in the field of circuit testing, for several motivations. A large amount of work has been devoted to the detection problems in the last decades [SS94, Ise06, FMM05], fault models are precisely defined and understood, as well as tools [Too03]. In [KPFS92] authors adopt a metric they named *diagnostic resolution*, taking into account parameters as the number of *undistinguished pairs of faults* and the number of *completely distinguished faults*. This information consider only the size of the groups of faults producing similar (or different) output values.

From another perspective it is also possible to integrate other pieces of information about the structure of the system under inspection. For instance, in [Pom11], the author proposes to consider the *physical distance* on the chip of two undistinguishable pair of faults, in the case of a combinational circuit. This is motivated considering that an *ambiguity* of diagnosis among two or more points located in distinct areas on the

circuit are more likely to be resolved by other inspection techniques for failure analysis, than closer or overlapped areas. In another work [PR97] authors propose a mixed automatic approach for the definition of the test suite: while the fault coverage is considered for computing the elements of a detection suite, the final suite is defined as the tradeoff of the number of selected tests and the number of *inspection points*, which provide additional test information observing the value of a digital signal on a specific *line* of the circuit.

Analysis based on a detailed model of the system suffer of a drawback [ABF90] whose manifestation is twofold: on one hand, the fault model used to describe the behavior of the system in presence of a fault might be inaccurate; on the other hand, the fault models considered (as the *stuck-at* fault models) might not cover faults discovered into the system, and this limitation has to be overcome through the introduction of more complex fault models as for instance in [SBB+98]. In the case this complexity cannot be handled, it is still possible to rely on the empirical observation that test suites designed to account only for the presence of simple fault-models (single stuck-at model), are *statistically* proven to be able to *detect* the presence of other type of failures (*multiple* stuck-at faults, *bridge* faults) [Pom11]. In this scenario, the correct diagnosis of the failure has to be left for later investigations, when necessary. A close assumption about the fault detection capability of a test suite is done by authors in [LCLH98], where they further extend their analysis proposing an AI post-processing of the outcomes of diagnostic tests; in particular, they designed a *bayesian classifier* to learn the probability of occurrence of some non-modeled faults from a large dataset of faulty integrated circuits.

Finally, also in [ZWGC09], authors propose to consider simple fault models only (single stuck-at model) to build a diagnosis approach, rather than relying on a *blind* simplification, they propose to describe the presence of a fault within a portion of the system (subcomponent) as a combination of simple faults located on its external interface (output pins). With this approach, fault coverage rates are claimed to be a more reliable indicator of the real fault coverage provided by a specific test suite. In this Chapter, we adopt a similar approach: describing the system through the BBN model described in Chapter 2, we define a test suite using the information contained in the system description only; and in order to prove the effectiveness of the obtained test suite, we later correlate its expected coverage with respect to an indicator based on a more detailed fault model.

Because of the twofold aim of test vectors in the diagnosis test suite, an incremental approach is beneficial to reduce costs and efforts, thus applying only a subset of all tests to identify the faulty component (or device, depending on the abstraction level the diagnosis works at). In such a scenario, rather than executing all tests and using the obtained complete syndrome, a two-phase process can be adopted:

1. a preliminary *scouting* phase, to make the fault observable as soon as possible;
2. a main *steady* phase, aimed at discriminating between possible candidates causing the tests to FAIL.

In this perspective, in the first phase, the value of a FAIL outcome is very significant, whereas in the second one, both PASS and FAIL tests add relevant information. The approach proposed in this chapter aims at the identification of group of tests Initial Test Set (initTS), making at least a test FAIL, by applying focused on being as small as possible and offering the highest possible coverage. Therefore, in an ideal situation, tests belonging to the initTS should provide a good fault coverage in a reduced number of vectors, whereas other tests executed in the latter phase should offer a high diagnostic accuracy. The same two-phase process models a scenario (Figure 3.1) where a failure detection test suite is adopted for all systems at a later stage of a manufacturing line; whenever a FAIL is outcome is present for this suite, the faulty instance is taken out redirected for deeper diagnosis, possibly finalized at repair; otherwise, the instance is classified as fault-free and can leave the manufacturing line.

Several works have been devoted to the selection of minimum-size test sets for diagnosis purposes, as in [PR02] or [PRR02]. In [SA09], the problem is modeled as an *optimization* problem and it is solved using a general purpose Integer Linear Programming solver. Also in [DM03] and [HST+06], similar methodologies are proposed for combinational and sequential circuits, minimizing the number of test vectors of the diagnostic test-sets. The focus of such methods is on guaranteeing the highest possible diagnostic resolution of the obtained test-set. All these techniques are based on a low abstraction-level fault model, suitable only for digital circuits of limited size.

In general, diagnosis can be applied at higher abstraction levels, where a system consists of components of relevant complexity, using a less accurate fault model, and without the opportunity to easily apply suites of tests produced with Automatic Test Pattern Generator (ATPG) tools, due to the modules' complexity and the interconnections among them, that limits accessibility. Therefore, although an ATPG-generated

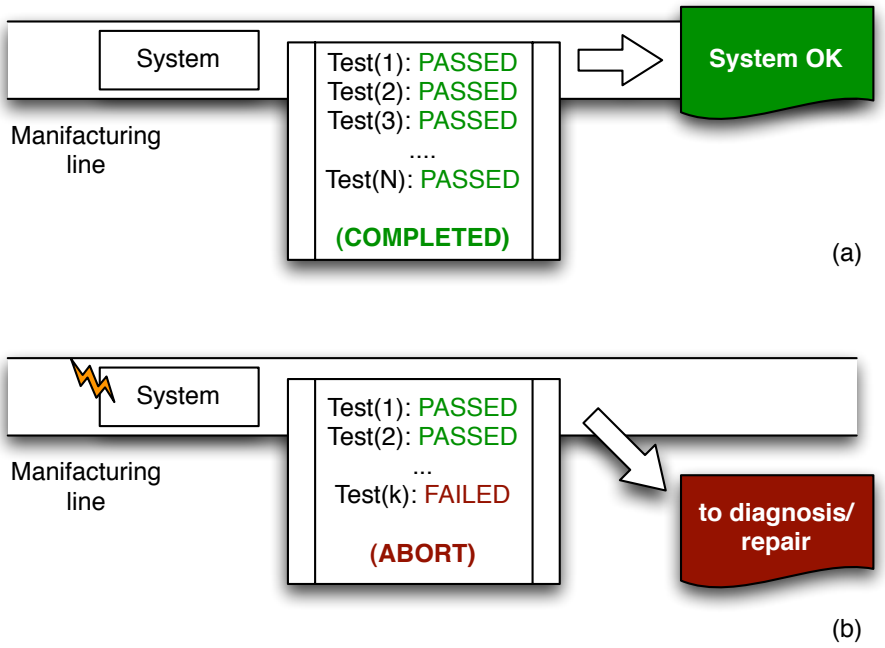


FIGURE 3.1: Manufacturing line test-suite and diagnosis line.

test suite could be used as the ideal model, it cannot be realistically applied and a different approach is necessary.

Considering the scenario depitected in Figure 3.1, another problem arises: not only the *selection* of an efficient group of tests to be part of the INITTS is critical for testing strategy, but also the *order of execution* is important for the strategy to be efficient. Not all sequences are equivalent in such context. On one hand, every test has to produce a PASS outcome before the instance of the system under test can leave the line (Figure 3.1(a)): being all tests associated with a *cost* (typically, *execution time*), the group of tests providing the *higher coverage* at the *lowest cost* are preferred as better choices for the INITTS.

The ordering or *sequencing* problem has been faced in the past in diagnosis literature, and the proposed algorithms are relatively different as the models used to describe both the system and the faults are not homogeneous. However, all problems are formulated as *optimization problems* and they are later solved relying on well known *optimization techniques*. For instance, in several works as [TP03, RTPPH04, KSC+08, BRdJ+09, BdJV+08a], the system under inspection is described using

the *faulty signatures* methodology (Section 1.3.2).

A problem similar to optimal sequencing of a test suite is described under the *regression test* problem paradigm [RH93], related to the software testing. In particular, this problem is usually formulated as the search for an *optimal* testing strategy of software modules, aimed at preventing local modifications on the module during development to be cause of future misbehaviors. In this contexts, tests are extracted from large and pre-defined test suites [RUCH01]; for this reason, *exahustive* testing or *random* testing are not suitable for both efficiency and coverage reasons. Adaptive techniques have been developed, for instance, in [RUCH01, WSKR06, LHH07, BMSS09, CPU07]; those techniques are defined taking into account the *coverage* of each item of the test suite with respect to the different portion of codes modified. In particular, authors in [RUCH01] introduce a particular metric (*Average Percentage of Faults Detected* (APFD)), to be used as a benefit function for comparing the efficiency of a given *order of execution* for a group of tests of the suite.

The contribution of the work proposed in this Chapter is twofold:

i) to tackle the problem of identifying an optimal initTS capable of providing the highest coverage of the system under diagnosis, while keeping the dimension of the test set small.
   The approach is applied using the methodology described in Section 2.3, in order to contain test costs by reducing the impact of the preliminary scouting phase, and to maintain most of the test effort on fault identification; it is worth noting that it is suitable for other diagnosis techniques at the functional level, where a generic fault model relating faults and test outcomes (syndromes) is adopted, information on the tests' fault coverage is not accurate, and probabilistic probabilistic relations between the presence of a fault in a system and the capability of each test of the test suites to detect it is available, as in [OMCE05].

i) to tackle the optimal execution sequence of tests contained in the initTS. For several reasons, i.e. an *a-priori* probability distribution leading to a bias towards specific components to be more likely *faulty candidates*, an execution order focusing on some target components first can have an expected *time to first* FAIL lower then another sequencing policy of the same test of initTS. All tests of the initTS scheduled *after* the first FAIL are not necessarily the best choice for diagnosis and fault isolation, and they do constitute an *potential overhead cost* if executed. Such saved testing time, even if representing a minor percentage of the overall cost associated with the optimal

INITTS, offers a significant impact of optimization in large volume lines.

The rest of Chapter is organized as follows: Section 3.1 introduces some definitions for establishing a methodology background. The main contribution is presented in Sections 3.2 and 3.3; in the former, the methodology for INITTS cardinality minimization is proposed, supported a numerical assessment of the consistency of the minimal solutions at a lower abstraction level, using an ATPG-based approach; in the latter, we cover the problem of the test-set ordering, describing an optimization approach based on an hill-climbing technique. At the end of each Section some experimental results and considerations are presented.

3.1 Definitions and problems formulation

We introduce few preliminary definitions to formalize the identification and sequencing of the INITTS in the form of *optimization* problems. In particular, it is necessary to formulate the concepts of system *coverage* and test set *cost* in order to provide quantitative metrics, to be used with general algorithms for *optimization*.

Definition 10. Let $\mathcal{S}$ be the *system under analysis* and $\mathcal{T}$ a suite of tests. Let us define $\mathcal{T}_y \subset \mathcal{T}$ as a *test-set*. We define $\mathcal{G}_{\text{COV}} : \mathcal{T}_y \rightarrow (0, 1)$ a coverage function representing the *portion* of faults, among all possible ones, covered by applying all tests contained in the test-set $\mathcal{T}_y$.

Definition 11. Let $\mathcal{S}$ be the *system under analysis* and $\mathcal{T}$ a suite of tests. Let us define $\mathcal{T}_y \subset \mathcal{T}$ as a *test-set*. We define $\mathcal{G}_{\text{cost}} : \mathcal{T}_y \rightarrow \mathcal{R}$ a real-valued cost function representing its *cost of execution* of the test-set $\mathcal{T}_y$.

Functions proposed in Definition 10 and 11 cannot be considered as *uncorrelated*. Intuition suggests that the larger the effort in testing, the lower is the likelihood that a failure might be undetected. For this reason, it is useful to represent both function using a unique representation, as proposed in Figure 3.2. In the plot, each possible group of test appears as a dot whose coordinates are computed through *coverage* $\mathcal{G}_{\text{COV}}$ and test-set *cost* $\mathcal{G}_{\text{cost}}$ functions.

Such functions can be defined as simple as linear relations (the *cost* of a test-set can be computed as its cardinality), or more complex non-functions, combining other parameters (for instance, execution time for the tests). However, independently from the choice of a specific relation,

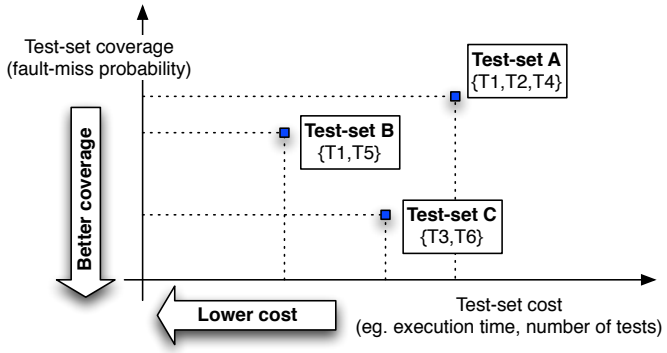


Figure 3.2: Test-set on cost-coverage plane.

coverage $\mathcal{G}_{\text{COV}}$ and test-set *cost* $\mathcal{G}_{\text{cost}}$ define a *multi-objective* optimization that can be constrained to formulate the problem of the identification of an initTS.

Problem 1. Let $\mathcal{S}$ be the *system under analysis* and $\widehat{\mathcal{G}_{\text{COV}}}$ the required *minimum coverage* parameter. We define the *minimum cost, coverage constraint* test-set optimization problem as:

$$\begin{aligned} &\text{minimize} && \mathcal{G}_{\text{cost}}(\mathcal{T}_y) \\ &\text{subject to} && \mathcal{G}_{\text{COV}}(\mathcal{T}_y) \geq \widehat{\mathcal{G}_{\text{COV}}} \end{aligned}$$

where optimal $\mathcal{T}_y$ depends on the choice of $\widehat{\mathcal{G}_{\text{COV}}}$.

The solution of Problem 1 is a minimum-cost test-set that is more likely to cover the largest part of the system, minimizing the probability not to obtain at least one FAIL from initTS. In particular, the *constraint* can be formulated to provide a minimum coverage not only to the system $\mathcal{S}$ as a whole, rather it can be defined to guarantee a minimum coverage for all its components. We propose an approach to identify a solution for this problem in Section 3.2.

Problem 2. For a system under diagnosis $\mathcal{S}$, let us define an allowed *maximum cost* parameter $\widehat{\mathcal{G}_{\text{cost}}}$. We define the *maximum coverage, cost constrained* test-set optimization problem as:

$$\begin{aligned} &\text{maximize} && \mathcal{G}_{\text{COV}}(\mathcal{T}_y) \\ &\text{subject to} && \mathcal{G}_{\text{cost}}(\mathcal{T}_y) \leq \widehat{\mathcal{G}_{\text{cost}}} \end{aligned}$$

whose the solution $\mathcal{T}_y$ depends on the choice of the parameter $\widehat{\mathcal{G}_{\text{cost}}}$.

The solution of Problem 2 is a maximum-coverage test-set, minimizing the probability not to obtain at least one FAIL from INITTS, while controlling the effort for the execution of the group of tests to be contained within an assigned budget (threshold $\widehat{\mathcal{G}_{\text{cost}}}$). We focus in upcoming Section 3.3 on a methodology to solve the *maximization* of the test-set coverage controlling, through the fitness function $\mathcal{G}_{\text{cov}}(\mathcal{T}_y)$, both the *selection* and the *order of execution* of tests.

3.1.1 Cost function

The execution each test is associated with a cost. The cost of a test-set is the sum of the contribution of all tests belonging to it. Without loss of generality, we could focus on the simplest form of cost function $\mathcal{G}_{\text{cost}}(\mathcal{T}_y)$, which is the *cardinality* of the test-set.

In general, the *cost* of a test should summarize the effort to produce the test outcome in a quantitative index. Without loss of generality, we could identify two main contributions:

(a) the *set-up time* of the test, taking into account the hardware/software operations of a test-operator to prepare the system for testing;
(b) the *test execution time*, which depends on the operations performed during the actual execution of the test.

A possibility to produce a valid estimation of the latter term is *scraping* the log data obtained of a test environment, which is usually configured to take trace of all events occurring during each test sessions. In particular, *time-stamps* relative to the beginning and conclusion of each test can be used to determine at least the coarse *statistics* of the test execution time, in particular for what concerns the *average* and *worst* cases of execution time.

It is worth noting that such statistics are consistent only when considering PASS test time, since they do involve a deterministic number of operations. Time to first FAIL does not have the same property, since for the sake of performance tests are usually *interrupted* whenever an error is detected. Such stop can occur at any point of the test execution, depending on the location of the fault in each particular system under analysis. However, considering only PASS time is not a significant limitation when dealing with the cost minimization from the test session begin until the detection of the *first* FAIL for a test. In this case, indeed, the error in the timing estimation resides only in the very last executed test, for which the average (or worst-case) PASS time is considered instead of the FAIL time.

Furthermore, it is difficult to obtain a precise evaluation of the test set-up time from log data, since time-stamps intervals depend in this case

```
Starting test: /callista5/fpga/dmaStats/test_asic_cpu_if  [Thu Jan 14 19:15:14 2010]
This command is currently supported only in standalone mode
*** PASSED *** /callista5/fpga/dmaStats/test_asic_cpu_if  [Thu Jan 14 19:15:14 2010]:
    skips=0 warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/fpga/dmaStats/test_dma_stats_engine

Starting test: /callista5/fpga/dmaStats/test_dma_stats_engine  [Thu Jan 14 19:15:14
    2010]
This command is currently supported only in standalone mode
*** PASSED *** /callista5/fpga/dmaStats/test_dma_stats_engine  [Thu Jan 14 19:15:14
    2010]: skips=0 warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/fpga/dmaStats/test_simultaneous_dma

Starting test: /callista5/fpga/dmaStats/test_simultaneous_dma  [Thu Jan 14 19:15:15
    2010]
This command is currently supported only in standalone mode
*** PASSED *** /callista5/fpga/dmaStats/test_simultaneous_dma  [Thu Jan 14 19:15:15
    2010]: skips=0 warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/fpga/mokaVsi/test_regs

Starting test: /callista5/fpga/mokaVsi/test_regs  [Thu Jan 14 19:15:15 2010]
*** PASSED *** /callista5/fpga/mokaVsi/test_regs  [Thu Jan 14 19:15:15 2010]: skips=0
    warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/fpga/test_access_fpgas

Starting test: /callista5/fpga/test_access_fpgas  [Thu Jan 14 19:15:16 2010]
*** PASSED *** /callista5/fpga/test_access_fpgas  [Thu Jan 14 19:15:16 2010]: skips=0
    warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/cpumodule/usb/test_usb

Starting test: /callista5/cpumodule/usb/test_usb -t 10  [Thu Jan 14 19:15:16 2010]
26670 packets successfully looped
*** PASSED *** /callista5/cpumodule/usb/test_usb -t 10  [Thu Jan 14 19:15:26 2010]:
    skips=0 warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/cpumodule/configprom/test_prom

Starting test: /callista5/cpumodule/configprom/test_prom -a 0 -length 1024 -save 1  [
    Thu Jan 14 19:15:27 2010]
*** PASSED *** /callista5/cpumodule/configprom/test_prom -a 0 -length 1024 -save 1  [
    Thu Jan 14 19:15:28 2010]: skips=0 warning=0 errors=0 loops=1
4510EUTP3(unlocked):/> /callista5/cpumodule/configprom/test_data_valid
```

Figure 3.3: Example of log (console output) of a test session.

on external factors (offline system configuration, manual intervention on the system, ...). For our analysis, we make the assumption that *set-up contributions* are non-recurrent and constant *overheads*, to be cumulated

to the overall test cost. Therefore, we consider in the following that all optimizations are based *recurrent* contributions, obtained from *test execution time*.

3.1.2 Coverage function

The definition of the *coverage* function for group of tests is not straightforward, because it depends on the fault model underlying the diagnosis methodology, and on the *interdependecies* of the different tests of the test-set on the system under analysis.

In this work we consider that system under diagnosis $\mathcal{S}$ is described through the CTM model (Section 2.3): for each component $\mathbf{c}_x$ of $\mathcal{S}$ and for each test $\mathbf{t}_y$ of the test suite $\mathcal{T}$ a *coverage coefficient* $\text{cov}(\mathbf{c}_x, \mathbf{t}_y)$ is defined.

In a bayesian scenario, $\text{cov}(\mathbf{c}_x, \mathbf{t}_y)$ represents the probability of obtaining a FAIL outcome when executing test $\mathbf{t}_y$ and component $\mathbf{c}_x$ *containing* a fault; it actually represents the probability of the fault in $\mathbf{c}_x$ to be *detected* when executing $\mathbf{t}_y$.

This probability can be evaluated as a *detection frequency*, similarly to the implemented methodology proposed in [ZWGC10b]. There are several alternatives to compute the *coverage* function $\mathcal{G}_{\text{cov}}(\mathcal{T}_y)$ for a test-set using those probability values, as presented in Example 7.

Example 7. Let us consider a simplified system containing three components $(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$, and test $\mathbf{t}_1$, with $\text{cov}(\mathbf{c}_1, \mathbf{t}_1) = \text{cov}(\mathbf{c}_2, \mathbf{t}_1) = 0.5$ and $\text{cov}(\mathbf{c}_3, \mathbf{t}_1) = 0.8$.

Using a frequency representation of probabilities, the coverage can be interpreted as the ratio between the number of faults detected by $\mathbf{t}_1$ and the total number of faults contained in $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$. Thus, the proportion of *detected faults* of the entire system is the *weighted average* of the coverage coefficients:

$$\mathcal{G}_{\text{cov}}(\mathbf{t}_1) = \frac{\sum_{x\in(1,2,3)} w_x \text{cov}(\mathbf{c}_x, \mathbf{t}_1)}{\sum_{x\in(1,2,3)} w_x} = 0.4$$

This computation would require the knowledge of the weight w_x, which is the number of faults contained in each component, an information not always available. A more conservative approach takes into account the *worst case coverage* to estimate the coverage of the entire system:

$$\mathcal{G}_{\text{cov}}(\mathbf{t}_1) = \min_{x\in(1,2,3)} \text{cov}(\mathbf{c}_x, \mathbf{t}_1)$$

◇

Do note that the coverage function proposed in Example 7 refers only to *singleton* test-sets. In general, at higher abstraction levels, we need to deal with test-sets composed of two or more tests.

For this reason, we need to extend the coverage function to *non-singleton* test-sets $\mathcal{T}_z = \{\mathbf{t}_{z1}, \mathbf{t}_{z2}, \ldots, \mathbf{t}_{zn}\}$ composed of n tests. In this case, for each component $\mathbf{c}_x$, several coverage coefficients ($\text{cov}(\mathbf{c}_x, \mathbf{t}_{z_1})$, ..., $\text{cov}(\mathbf{c}_x, \mathbf{t}_{z_n})$) are available. Again, a *worst-case assumption* would be the selection of the *maximum* coverage for each component, such as:

$$\mathcal{G}_{\text{cov}}(\mathcal{T}_z) = \min_{\mathbf{c}_x}(\max_{\mathbf{t}_z \in \mathcal{T}_z} \text{cov}(\mathbf{c}_x, \mathbf{t}_z)) \tag{3.1}$$

Example 8. Let us consider the example CTM of Figure 3.4, and in particular the group composed of tests $\mathbf{t}_1$ and $\mathbf{t}_2$. Using the *worst case* coverage function, we compute the coverage for all components as:

$$\begin{aligned}
\mathbf{c}_1 &: \max(\text{cov}(\mathbf{c}_1, \mathbf{t}_1), \text{cov}(\mathbf{c}_1, \mathbf{t}_2)) = \max(0.9, 0.0) = 0.9 \\
\mathbf{c}_2 &: \max(\text{cov}(\mathbf{c}_2, \mathbf{t}_1), \text{cov}(\mathbf{c}_2, \mathbf{t}_2)) = \max(0.5, 0.9) = 0.9 \\
&\ldots \\
\mathbf{c}_6 &: \max(\text{cov}(\mathbf{c}_6, \mathbf{t}_1), \text{cov}(\mathbf{c}_6, \mathbf{t}_2)) = \max(0.1, 0.0) = 0.1
\end{aligned}$$

Thus, the coverage for the entire system is: $\min(0.9, 0.9, \ldots, 0.1) = 0.1$

◇

This assumption does not take into account exhaustively the fact that coverage coefficients $\text{cov}(\mathbf{c}, \mathbf{t})$ represent also probability values of the BBN, and it misses important pieces of information obtainable from the *properties* of this model. This information can be exploited to obtain a better formulation of the coverage function. More precisely, the coverage value $\text{cov}(\mathbf{c}_x, \mathbf{t}_{z1})$ represents the probability to obtain a FAIL from test $\mathbf{t}_{z1}$ when component $\mathbf{c}_x$ contains a fault.

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$	$\mathbf{t}_6$	$\mathbf{t}_7$	$\mathbf{t}_8$
$\mathbf{c}_1$	0.9	0.0	0.0	0.0	0.0	0.9	0.0	0.0
$\mathbf{c}_2$	0.5	0.9	0.0	0.9	0.0	0.9	0.9	0.0
$\mathbf{c}_3$	0.1	0.5	0.9	0.9	0.0	0.5	0.0	0.0
$\mathbf{c}_4$	0.0	0.5	0.0	0.5	0.9	0.0	0.0	0.5
$\mathbf{c}_5$	0.5	0.0	0.0	0.0	0.0	0.0	0.5	0.9
$\mathbf{c}_6$	0.1	0.0	0.0	0.0	0.9	0.0	0.5	0.5

Figure 3.4: CTM for Examples 8 and 10.

Definition 12. Let us consider a faulty component $\mathbf{c}_x$. Let us consider also a test $\mathbf{t}_z$ stimulating $\mathbf{c}_x$, with a coverage $\text{cov}(\mathbf{c}_x, \mathbf{t}_z)$. We define *un-detection probability* of $\mathbf{t}_z$ the probability that the test result with a PASS outcome:

$$P(\mathbf{t}_z = \text{PASS} | \mathbf{c}_x = \text{faulty}) = 1 - \text{cov}(\mathbf{c}_x, \mathbf{t}_z) \tag{3.2}$$

Coverage of components *increases* when several tests are executed, and their outcome is made available. Therefore, for each component, it is necessary to evaluate the *un-detection* probability, combining the information coming from different tests.

Let us consider a simple scenario where two tests are executed. We can make two opposite hypothesis:

- **best-case:** tests stimulate portions of a target component which are completely non-overlapped; thus, the *un-detection* probability of a fault within the component of the tests pair is the sum of *un-detection* probabilities of the tests.
- **worst-case:** both tests stimulate portions of a target component which are completely overlapped ; thus, the *un-detection* probability of a fault within the component of the tests pair is equal to the *un-detection* probability of each test, taken singularly.

In general, we consider an intermediate scenario (**average-case**), with an equal proportion of overlapped and non-overlapping portions. This corresponds to using as *un-detection* probability information of the tests pair the *product* of *un-detection* probabilities of the tests. From a probabilistic point of view, this is equivalent to stating an hypothesis of *independency* for the probabilities of the tests of the entire test suite.

Example 9. Let us consider a single component and two tests from the test suite, where each test has 50% of coverage. The execution of these two tests could lead to two extreme cases:

- the final coverage is 100%; when tests analyze parts of the component sharing no logic (*best-case* hypothesis), and
- the final coverage is 50%; when tests analyze exactly the same parts of the component (*worst-case* hypothesis)

The more realistic situation (the scenario where tests analyze parts of the component which are only partially overlapped), leads to the reasonable hypothesis that the final coverage is 75%.

◇

From a strictly probabilistic point of view, this could be extended to form the hypothesis of probabilistic independency of *un-detection*:

$$P(\mathbf{t}_{z1}, \mathbf{t}_{z2} | \mathbf{c}_x) = P(\mathbf{t}_{z1} | \mathbf{c}_x) \cdot P(\mathbf{t}_{z2} | \mathbf{c}_x) \tag{3.3}$$

and the coverage function for the test pair $\{\mathbf{t}_{y1}, \mathbf{t}_{y2}\}$ is:

$$\mathcal{G}_{\text{cov}}(\{\mathbf{t}_{y1}, \mathbf{t}_{y2}\}) = 1 - P(\mathbf{t}_{y1}, \mathbf{t}_{y2}|\mathbf{c}_x) \tag{3.4}$$

Equation 3.4 can be generalized as follows:

Definition 13. Let $\mathcal{S}$ be the *system under analysis*, $\mathcal{T}_z = \{\mathbf{t}_{z1}, \mathbf{t}_{z2}, \ldots, \mathbf{t}_{zn}\}$ a generic test-set composed of n tests from the test suite, and $\text{cov}(\mathbf{c}_x, \mathbf{t}_z)$ the set of coverage coefficients for all components $\mathbf{c}_x$ in $\mathcal{S}$, for all tests $\mathbf{t}_z \in \mathcal{T}_z$ derived from the CTM. We define the *multi test* test-set coverage function $\mathcal{G}_{\text{cov}}(\mathcal{T}_z)$ as:

$$\mathcal{G}_{\text{cov}}(\mathcal{T}_z) = \min_{\mathbf{c}_x}(1 - \prod_{t_z \in \mathcal{T}_z} (1 - \text{cov}(\mathbf{c}_x, \mathbf{t}_z))) \tag{3.5}$$

Example 10. Consider the example CTM in Figure 3.4. We compute the *undetection* probability of the test-set $\{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$ for each component using Equation 3.3, as:

$$\begin{aligned}
\mathbf{c}_1 \quad &: \quad (1 - \text{cov}(\mathbf{c}_1, \mathbf{t}_1))(1 - \text{cov}(\mathbf{c}_1, \mathbf{t}_2))(1 - \text{cov}(\mathbf{c}_1, \mathbf{t}_3)) = \\
&\quad (1 - 0.9) \cdot (1 - 0) \cdot (1 - 0) = 0.1 \\
\mathbf{c}_2 \quad &: \quad (1 - \text{cov}(\mathbf{c}_2, \mathbf{t}_1))(1 - \text{cov}(\mathbf{c}_2, \mathbf{t}_2))(1 - \text{cov}(\mathbf{c}_2, \mathbf{t}_3)) = \\
&\quad (1 - 0.5) \cdot (1 - 0.9) \cdot (1 - 0) = 0.05 \\
&\ldots \\
\mathbf{c}_6 \quad &: \quad (1 - \text{cov}(\mathbf{c}_6, \mathbf{t}_1))(1 - \text{cov}(\mathbf{c}_6, \mathbf{t}_2))(1 - \text{cov}(\mathbf{c}_6, \mathbf{t}_3)) = \\
&\quad (1 - 0.1) \cdot (1 - 0) \cdot (1 - 0) = 0.9
\end{aligned}$$

Thus, the coverage for the system is: $\min(1-0.1, 1-0.05, \ldots, 1-0.9) = 0.1$

◇

3.1.3 Test-set system coverage and test sequencing

The test-set coverage function introduced in Definition 13 combines, for the selected test-set, coverages of components in an unique expression $\mathcal{G}_{\text{cov}}$, involving the system under analysis $\mathcal{S}$ as a whole.

While the approach is useful for what concerns the use of the coverage function as a *constraint* (Problem 1), this is not necessary true when it is adopted as an *evaluation* function, i.e., a fitness or cost function (Problem 2). In this latter case, two key pieces of information are not considered explicitly:

i) the weights, or more in general the composition function, used to combine *contributions* from single components;

ii) the impact of the order of execution of the tests of the test-sets on the *incremental* degree of coverage of the system, and relationship with test costs.

For the composition function i), a straightforward alternative to the *min* operations introduced in Definition 13 is a linear combination of the *un-detection* probabilities, evaluated independently for each component. This choice introduces as a side effect a weighting factor among all components, to be used for defining a *priority* for detecting faults contained in a group of components first, with respect to all others.

Definition 14. Let $\mathcal{S}$ be the *system under analysis*, $\mathcal{T}_z = \{\mathbf{t}_{z1}, \mathbf{t}_{z2}, \ldots, \mathbf{t}_{zn}\}$ a generic test-set composed of n tests from the test suite, and $\mathrm{cov}(\mathbf{c}_x, \mathbf{t}_z)$ the set of coverage coefficients for all components $\mathbf{c}_x$ in $\mathcal{S}$, for all tests $\mathbf{t}_z \in \mathcal{T}_z$ derived from the CTM. Let w_x a weight factor, for each $\mathbf{c}_x$, such that $\sum_{\mathbf{c}_x} w_x = 1$.

We define a linear *multi test* test-set coverage function $\mathcal{G}_{\mathrm{COV}}(\mathcal{T}_z)$ as:

$$\mathcal{G}_{\mathrm{COV}}(\mathcal{T}_z) = 1 - \sum_{\mathbf{c}_x} w_x (1 - \prod_{t_z \in \mathcal{T}_z} (1 - \mathrm{cov}(\mathbf{c}_x, \mathbf{t}_z))) \tag{3.6}$$

The simplest choice for the weight factor is $w_x = \bar{w} = \frac{1}{n}$, where n is the number of components contained in $\mathcal{S}$, and Equation 3.6 corresponds to the *average coverage* of the components of the system.

Note Equation 3.6 can be also interpreted from a pure BBN perspective: it represents the probability to observe an *all*-PASS configuration at all outcomes of tests the test-set. For this reason, it could be computed directly through inference: in particular the *quickscore* inference algorithm (Section 2.2) keeps this piece of information in a specific field to obtain this information. However, on one hand the computational overhead of the engine is larger than the direct evaluation of Equation 3.6, and on the other hand, is is not possible to tune components' *weights* in BBNs inference since they are constraint, in the quick-score, to be the *a-priori* probability values.

◇

Example 11. Consider the example CTM in Figure 3.4. Let us consider a test-set $\mathcal{T}_z = \{\mathbf{t}_{z1}, \mathbf{t}_{z4}\}$. The coverage for components is

$$\begin{aligned}
\mathbf{c}_1 &: (1 - \mathrm{cov}(\mathbf{c}_1, \mathbf{t}_1))(1 - \mathrm{cov}(\mathbf{c}_1, \mathbf{t}_4)) = (1 - 0.9) \cdot (1 - 0) = 0.1 \\
\mathbf{c}_2 &: (1 - \mathrm{cov}(\mathbf{c}_2, \mathbf{t}_1))(1 - \mathrm{cov}(\mathbf{c}_2, \mathbf{t}_4)) = (1 - 0.5) \cdot (1 - 0.9) = 0.05 \\
\mathbf{c}_3 &: (1 - \mathrm{cov}(\mathbf{c}_3, \mathbf{t}_1))(1 - \mathrm{cov}(\mathbf{c}_3, \mathbf{t}_4)) = (1 - 0.1) \cdot (1 - 0.9) = 0.09 \\
\mathbf{c}_4 &: (1 - \mathrm{cov}(\mathbf{c}_4, \mathbf{t}_1))(1 - \mathrm{cov}(\mathbf{c}_4, \mathbf{t}_4)) = (1 - 0.0) \cdot (1 - 0.5) = 0.5 \\
\mathbf{c}_5 &: (1 - \mathrm{cov}(\mathbf{c}_5, \mathbf{t}_1))(1 - \mathrm{cov}(\mathbf{c}_5, \mathbf{t}_4)) = (1 - 0.5) \cdot (1 - 0) = 0.5 \\
\mathbf{c}_6 &: (1 - \mathrm{cov}(\mathbf{c}_6, \mathbf{t}_1))(1 - \mathrm{cov}(\mathbf{c}_6, \mathbf{t}_4)) = (1 - 0.1) \cdot (1 - 0) = 0.1
\end{aligned}$$

Average coverage is obtained using $w_x = \bar{w} = \frac{1}{6}$, and it is:

$$\mathcal{G}_{\mathrm{cov}}(\mathcal{T}_z) = 1 - \frac{1}{6} \cdot (0.1 + 0.05 + 0.09 + 0.5 + 0.5 + 0.1) = \cdots = 0.776$$

Otherwise, we can consider a scenario where components $\mathbf{c}_1$ and $\mathbf{c}_2$ require higher priority for detection; we can attribute an (arbitrary) higher value for those components, for instance $w_1 = w_2 = 2/8$ and $w_3 = \cdots = w_6 = 1/8$.

Because of the fact that the *un-detection* probabilities of those components, due to the selected test-set, is relatively low, we obtain an increase of the system coverage:

$$\mathcal{G}_{\mathrm{cov}}(\mathcal{T}_z) = 1 - \frac{2}{8} \cdot (0.1 + 0.05) - \frac{1}{8} \cdot (0.09 + 0.5 + 0.5 + 0.1) = \cdots = 0.813$$

◇

System coverage and cost are correlated to the test execution order, as we can visualize on a curve dubbed *Cumulative Coverage* (CC) curve (for instance the one shown in Figure 3.5). This curve can be drawn for each possible test-set sequence $\mathcal{K}_{\mathcal{T}} = \{1 : \mathbf{t}_{k1}, 2 : \mathbf{t}_{k2}, 3 : \mathbf{t}_{k3}, \dots\}$, plotting on the coverage-cost space the list of points with coordinates $\langle \mathcal{G}_{\mathrm{cost}}(\{\mathbf{t}_{k1}\}), \mathcal{G}_{\mathrm{cov}}(\{\mathbf{t}_{k1}\}) \rangle$, $\langle \mathcal{G}_{\mathrm{cost}}(\{\mathbf{t}_{k1}, \mathbf{t}_{k2}\}), \mathcal{G}_{\mathrm{cov}}(\{\mathbf{t}_{k1}, \mathbf{t}_{k2}\}) \rangle$, ….

According to Equation 3.6, system coverage function $\mathcal{G}_{\mathrm{cov}}$ is equivalent to a *cumulative probability* distribution, and in particular to the cumulative probability of the *time to first* FAIL event. From fundamental probability theory, the *area* under the CC curve is *proportional* to the average *cost* of the test sequence $\mathcal{K}_{\mathcal{T}}$, then the minimization can be performed irrespectively with one or the other variable.

Example 12. Let us consider a test-set $\mathcal{T}$ composed of four tests: $\{\mathbf{t}_1$, $\mathbf{t}_2$,$\mathbf{t}_3$,$\mathbf{t}_4\}$. The excerpt of the CTM containing the coverages of $\mathcal{T}$ for

	$\mathbf{P}(\mathbf{c}$	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$
$\mathbf{c}_1$	0.25	0.9	-	0.5	-
$\mathbf{c}_2$	0.25	0.9	-	-	0.5
$\mathbf{c}_3$	0.25	-	-	0.5	0.9
$\mathbf{c}_4$	0.25	-	0.5	-	0.9

a 4-component system is shown in the following table; for the sake of simplicity, all components have an equal *a-priori* probability (0.25).

System coverage can be computed rather easily through Equation 3.6 for all combinations of tests: for instance, $\mathbf{t}_2$ has the minimum coverage $\mathcal{G}_{\text{cov}}(\{\mathbf{t}_2\}) = 0.125$ (only component $\mathbf{c}_2$ is involved, with a 50% coverage), while the test pair $\{\mathbf{t}_1, \mathbf{t}_4\}$ offers an high coverage $\mathcal{G}_{\text{cov}}(\{\mathbf{t}_1, \mathbf{t}_4\}) = 0.912$ (all components are covered at 90%). The maximum coverage is attained by $\mathcal{T}$ and $\mathcal{G}_{\text{cov}}(\mathcal{T}) = 0.95$.

Figure 3.6 compares the CC curve (a) for a *good* sequence $\{\mathbf{t}_4, \mathbf{t}_1, \mathbf{t}_3, \mathbf{t}_2\}$ and (b) for a *bad* sequence $\{\mathbf{t}_2, \mathbf{t}_3, \mathbf{t}_1, \mathbf{t}_4\}$ (b), under the hypothesis that all tests have unit cost.

Sequence $\{\mathbf{t}_4, \mathbf{t}_1, \mathbf{t}_3, \mathbf{t}_2\}$ is no longer optimal if we set $\mathbf{t}_4$ to have a cost which is 100 times bigger than other tests: Figure 3.7 compares this sequence with another sequence, $\{\mathbf{t}_1, \mathbf{t}_3, \mathbf{t}_2, \mathbf{t}_4\}$, putting the highest cost test in the last position.

◇

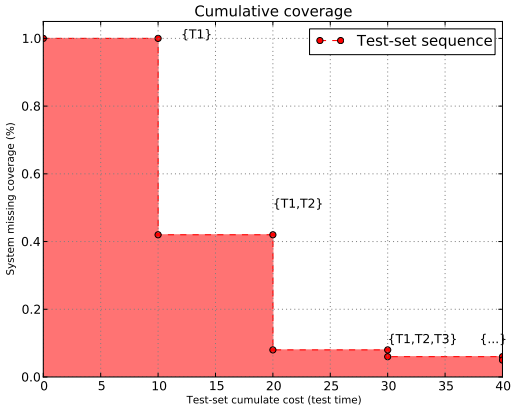


FIGURE 3.5: CC curve.

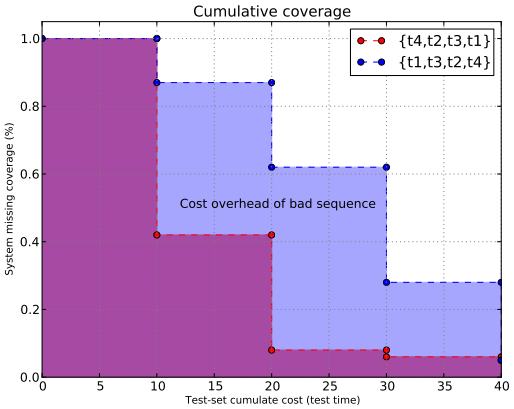


Figure 3.6: CC curve for Example 12, uniform test costs.

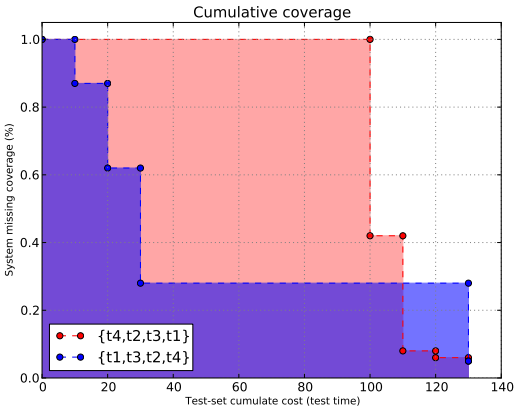


Figure 3.7: CC curve for Example 12, different test costs.

3.2 Minimum cost initTS

When the the *cardinality* is used as cost function $\mathcal{G}_{\text{cost}}(\mathcal{T}_y)$, the problem is also known in operations research literature as the *set covering problem* [Cor01].

We introduce two optimization algorithms for the solution of the defined test-set coverage problem, the former based on a greedy heuristic, the latter exploiting the Integer Linear Programming (ILP) paradigm.

Input: CTM (System Model), $\mathcal{T}$ (Test Suite), $\widehat{\mathcal{G}_{\text{cov}}}$ (Required cov.)
Output: INITTS.
Procedure GreedyInitTS(CTM, $\mathcal{T}$, $\widehat{\mathcal{G}_{\text{cov}}}$)
{
1 initTS $= \emptyset$
2 availableTS $= \mathcal{T}$
3 STOP $=$ False
4 while ((availableTS $\neq \emptyset$) and (not STOP))
5 $\quad \mathbf{t}_{\max} = \underset{\mathbf{t}_k \in \text{availableTS}}{\text{argmax}}\, \mathcal{G}_{\text{cov}}(\text{INITTS} + \{\mathbf{t}_k\})$
6 $\quad$ initTS $=$ initTS $+ \{\mathbf{t}_{\max}\}$
7 $\quad$ availableTS $=$ availableTS - $\{\mathbf{t}_{\max}\}$
8 $\quad$ STOP $= \mathcal{G}_{\text{cov}}(\text{INITTS}) > \widehat{\mathcal{G}_{\text{cov}}}$
9 end while
10 **output** INITTS
}

FIGURE 3.8: A greedy algorithm for the INITTS.

3.2.1 Greedy heuristic

An optimal INITTS could be obtained by using a heuristic greedy approach given by the algorithm in Figure 3.8. From an empty test-set (line 1) the algorithm adds new tests from the test suite, that maximize the system coverage (line 5). The loop is executed until the required coverage is reached or no other tests are available (line 4).

Algorithm in Figure 3.8 uses Equation 3.5 as both a stop criterium (line 4) and a selection criterium, to identify which test is to be added to the INITTS. The approach suffers of the typical drawback of greedy algorithms, tending to prefer tests with higher coverage coefficients first, a potential pitfall possibly leading to local-minimum solutions.

Example 13. Let us consider again the CTM presented in Figure 3.4. A minimum coverage $\widehat{\mathcal{G}_{\text{cost}}} = 0.75$ is required. Let $\{\mathbf{t}_1, \mathbf{t}_2\}$ be an initial solution. The minimum coverage (0.1) is for component $\mathbf{c}_6$. Among all possibilities, test $\mathbf{t}_5$ is selected because it produces an higher minimum coverage (0.5), on component $\mathbf{c}_3$. Then, tests $\mathbf{t}_3$ and $\mathbf{t}_8$ are selected, until all components are covered at the required minimum level. Final solution of the greedy algorithm results INITTS $= \{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_5, \mathbf{t}_3, \mathbf{t}_8\}$.

$\diamond$

3.2.2 ILP optimization

We propose another method to solve the formulated problem of finding the optimal test-set using Integer Linear Programming (ILP) [BV04]. ILP is an optimization technique where both the objective function and all constraints are *linear*, and solutions are integer-valued.

Formally, we define an ILP problem as:

$$\begin{aligned} &\text{minimize} && \mathbf{c}^T\mathbf{y} \\ &\text{subject to} && \mathbf{a}_x^T\mathbf{y} \leq b_x \quad \forall x \in [1, 2, \ldots, n_c] \end{aligned} \tag{3.7}$$

where $\mathbf{y}$ is the the solution of the problem; $\mathbf{c}$, $\mathbf{a}_1$, ..., $\mathbf{a}_{n_c}$ are real-valued vectors of the same size of $\mathbf{y}$, b_1, b_2, ..., b_{n_c} are real-valued scalars, and n_c is the number of constraints to be satisfied by solution $\mathbf{y}$.

In the given scenario, n_c corresponds to the number of components of the system, and $\mathbf{y} = (y_1, \ldots, y_{n_t})$ is a binary vector of size n_t (number of tests in $\mathcal{T}$) defining test-set $\mathcal{T}_y$, where each entry y_i is equal to 1 if and only if $\mathbf{t}_i \in \mathcal{T}_y$.

Since the objective function is the *cardinality* of the test-set $\mathcal{T}_y$, it is obtained in Equation 3.7 by setting $\mathbf{c} = \mathbf{1}$ (all 1's vector).

The constraint functions are expressed, by using Equation 3.5, as:

$$\prod_{t_z \in \mathcal{T}_z} (1 - \text{cov}(\mathbf{c}_x, \mathbf{t}_z)) \leq (1 - \widehat{\mathcal{G}_{\text{cov}}}) \tag{3.8}$$

which, by using a *logarithm* transformation, reduce to:

$$\sum_{t_z \in \mathcal{T}_z} \log(1 - \text{cov}(\mathbf{c}_x, \mathbf{t}_z)) \leq \log(1 - \widehat{\mathcal{G}_{\text{cov}}}) \tag{3.9}$$

Thus, the elements a_x^j (j-th component of vector $\mathbf{a}_x$) and the scalar b_x of the ILP problem of Equation 3.7 are defined as:

$$a_x^j = \log(1 - \text{cov}(\mathbf{c}_x, \mathbf{t}_j)) \tag{3.10}$$

$$b_x = \log(1 - \widehat{\mathcal{G}_{\text{cov}}}) \tag{3.11}$$

3.2.3 Analysis and experimental results

The solution obtained by ILP is optimal, exploiting the information contained in the CTM system model. Indeed, we want to validate the adoption of the ILP problem formulation against i) the adopted hypothesis for the coverage function in Equation 13, and ii) the fact that the CTM actually specifies high abstraction-level information on the coverage, rather than accurate fault coverage per test data. In fact, dealing

with the diagnosis of large systems, the test engineers' knowledge of the capability of each test to detect the presence of a fault is intrinsically limited. therefore, to handle the complexity of the problem, a high abstraction-level model is adopted and to simplify the definition of the system CTM, the coverage coefficients of the matrix are selected from a coarse, limited set.

However the coverage of the INITTS identified with the methodology reflects the *actual* coverage of the system under diagnosis if the coefficients of the CTM represent the *exact* probabilities in Equation 3.2. Only more precise information about the causal relationship between the presence of a fault and the PASS/FAIL outcomes of tests would provide better estimation of the coverage of the system provided by the INITTS.

Therefore, to validate the approach, we apply it at a lower abstraction level, where an accurate relation between faults and tests is available, and coverage actually represents the probability of a vector to make the fault observable. In particular, we use *digital combinational circuits* and the single stuck-at fault. In this scenario, with a classical ATPG [Ha94] we extracted the basic information to build the circuit CTM.

We give an example of the use of the ATPG starting from the $c17$ combinational circuit from ISCAS85 benchmarks. Figure 3.9 depicts the circuit under analysis, partitioned arbitrarily in two *logical* components $\mathbf{c}_1$ and $\mathbf{c}_2$: the first gathers gates G10 and G11 (plus input signals IN1, IN3, IN6), while the second aggregates gates G16, G19, G22 and G23 (plus input signals IN2 and IN7)[1]. Furthermore, each partition gathers one or more adjacent gates: this condition is not essential but it preserves the *physical* nature of the logical component.

On the partitioned circuit, it is possible to identify both the set of each possible stuck-at fault affecting each single logical component. For instance, the stuck-at-0 fault on signal IN6 is potentially affecting $\mathbf{c}_1$, while the stuck-at-1 fault on the output of gate G19 is potentially affecting $\mathbf{c}_2$. It is possible that a fault is *shared* between the components.

By using the ATPG, two pieces of information can be obtained: a set of test vectors and, for each of them, the set of faults that they can detected. All tests of the test suite ($\mathcal{T}_{\mathrm{c17}} = \{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3\}$) for the system diagnosis are defined as *groups of test vectors*. This is shown in Figure 3.10, where test vectors $tv_1 \cdots tv_7$ are grouped to form tests $\mathbf{t}_1$, $\mathbf{t}_2$ and $\mathbf{t}_3$ defined as $\mathbf{t}_1 = \{tv_1, tv_2\}$, $\mathbf{t}_2 = \{tv_3, tv_4, tv_5\}$ and $\mathbf{t}_3 = \{tv_6, tv_7\}$.

The outcome of such tests is PASS if all output values correspond to

[1]Note that other partitions could be obtained, and a single circuit is a potential source of a set of examples.

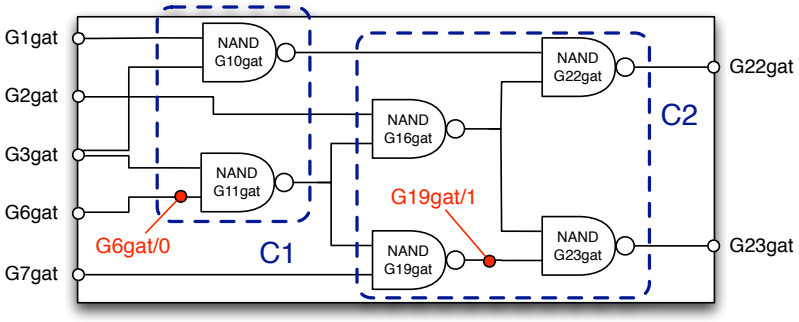


Figure 3.9: $c17$ circuit partition.

	$\mathbf{t}_1$		$\mathbf{t}_2$			$\mathbf{t}_3$	
	tv_1	tv_2	tv_3	tv_4	tv_5	tv_6	tv_7
$\mathbf{c}_1$ f1	x					x	
f2	x	x	x	x	x		
f3					x		
f4	x	x					
f5		x					
$\mathbf{c}_2$ f6			x			x	
f7			x			x	x
f8						x	

Figure 3.10: Example of fault-vector binary matrix.

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
$\mathbf{c}_1$	0.80	0.40	0.20
$\mathbf{c}_2$	0.00	0.66	1.00

Figure 3.11: Derived CTM.

the fault-free circuit response for all test vectors in the group. The use of groups of test vectors instead of single test vectors guarantees higher coverage for components. Thus, the coefficients of the obtained teCTM are similar to those hand-designed by test engineers.

From the table in Figure 3.10, it is possible to compute the *un-detection* probability of the tests referred to each component. For instance, test $\mathbf{t}_1$ in Figure 3.10 covers faults $\{f_1, f_2, f_4, f_5\}$ in $\mathbf{c}_1$, whereas

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
$\mathbf{c}_1$	0.90	0.50	0.10
$\mathbf{c}_2$	0.00	0.50	0.90

FIGURE 3.12: Corresponding teCTM.

$\mathbf{t}_3$ $\{f_6, f_7, f_8\}$ in $\mathbf{c}_2$. Thus the un-detection probabilities are $\frac{1}{5} = 0.2$ and $\frac{0}{3} = 0$ for $\mathbf{c}_1$ and $\mathbf{c}_2$, respectively. As a result, the corresponding CTM can be computed (Figure 3.11), which is then used to derive the one that uses the adopted coarse-grain set of values, teCTM in Figure 3.12.

In [ABSF10a], the AF2D methodology was used for the diagnosis of a real telecom board, provided by Cisco Photonics (CISCO1). This is a medium-size board, described with a CTM containing 55 components, and whose diagnostic test suite is composed of 86 tests. The INITTSs identified using the greedy algorithm and the ILP optimization are shown in Table 3.1. It presents three columns, for both approaches, and minimum required coverage $\mathcal{G}_{\widehat{\text{COV}}}$ (Problem 1) is fixed respectively to 90%, 95% and 99%.

Table 3.1: INITTS size for CISCO1 board.

	Greedy			ILP		
Circuit	90%	95%	99%	90%	95%	99%
CISCO1	15	15	17	10	11	12

As it can be observed, ILP method allows saving up to 5 tests, corresponding to an average 30% reduction of the fixed costs of AF2D due to INITTS size.

Experimental analyses have been executed then on ten combinational circuits with different characteristics, obtained from the ISCAS85 benchmark suite. For each circuit, 10 arbitrary partitions have been computed using the presented procedure. Each partition represents a different system under analysis; thus, in total 100 matrices have been extracted defining the teCTM, computed using the same coarse set of possible coverage coefficients introduced for manual creation of system models. Coefficients (H,M,L,-) have been mapped respectively to values (0.9, 0.5, 0.1 and 0.0). Tab. 3.2 reports the characteristics of the used circuits.

First, let us consider the results of INITTS optimization using the ILP approach. Tab. 3.3 reports, for each circuit, the average size (rounded

Table 3.2: Summary of circuits properties.

Circuit	**# Comp.**	**# Faults**	**#Tests** $\mid \mathcal{T} \mid$	**# Vectors**
c432	15	524	35	402
c499	15	758	36	358
c880	20	942	54	826
c1355	20	1574	52	703
c1908	50	1879	113	1512
c2670	50	2747	128	922
c3540	50	3428	103	2244
c5315	80	5350	204	1921
c6288	80	7744	120	340
c7552	80	7550	216	4217

Table 3.3: Optimal initTS size.

Circuit	90%	95%	99%
c432	8 (78)	8 (103)	9 (114)
c499	5 (80)	6 (87)	8 (96)
c880	8 (71)	10 (96)	12 (108)
c1355	6 (113)	7 (132)	9 (150)
c1908	11 (266)	16 (297)	21 (319)
c2670	12 (152)	14 (160)	17 (181)
c3540	14 (180)	18 (231)	19 (232)
c5315	7 (178)	11 (207)	13 (213)
c6288	6 (42)	8 (48)	10 (53)
c7552	11 (304)	15 (349)	18 (391)

at the nearest integer) of the initTS exploiting the system description teCTM; the size specifies the number of tests and, in parenthesis, the total number of test vectors constituting the tests. We computed the initTS aiming at achieving diffetent fault coverages (namely 90%, 95%, and 99%), to analyze the growth of such test sets.

For instance, for circuit $c432$, when aiming at a 90% fault coverage in the initial diagnostic phase, 8 tests are included in initTS for a total of 78 test vectors, while for a 95% coverage, tests are 9, consisting of 103 test vectors.

Table 3.4: Component coverage (from CTM and Stuck-at model).

	90%		95%		99%	
Circuit	CTM	Stuck-at	CTM	Stuck-at	CTM	Stuck-at
c432	98.58%	95.61%	99.17%	97.14%	99.58%	98.28%
c499	95.39%	81.79%	97.94%	89.18%	99.72%	95.91%
c880	95.93%	90.02%	98.46%	95.65%	99.74%	98.41%
c1355	95.55%	83.35%	97.92%	86.40%	99.56%	89.77%
c1908	93.81%	77.96%	97.35%	78.60%	99.52%	78.81%
c2670	93.67%	84.91%	97.42%	85.92%	99.67%	86.84%
c3540	94.44%	78.82%	96.83%	81.16%	99.41%	85.68%
c5315	94.93%	79.20%	97.36%	82.04%	99.51%	86.06%
c6288	94.90%	96.28%	98.86%	97.47%	99.74%	98.85%
c7552	94.31%	83.33%	96.99%	86.11%	99.23%	89.64%

Table 3.5: Comparison of ILP and ATPG solutions.

	ILP		ATPG	
Circuit	#TV	Fault Cov.	# TV	Fault Cov.
c432	114	98.28%	49	99.23%
c499	97	95.91%	54	98.94%
c880	109	98.41%	62	100.00%
c1355	150	89.77%	86	99.49%
c1908	320	78.81%	188	99.52%
c2670	182	86.84%	98	95.74%
c3540	232	85.68%	148	96.04%
c5315	213	86.06%	117	98.89%
c6288	54	98.85%	34	99.56%
c7552	392	89.64%	204	98.25%

Table 3.4 proposes a comparison: on the first column, for each class of circuits and for each required coverage, it presents the system coverage computed with high level coefficients. This coverage is estimated using all components of the system and not only the worst case (i.e. the component with the minimum coverage), which constraints the ILP optimization and it is closer to the required coverage threshold, but does not take into account the overtesting of some components. On the second column, the *actual* proportion of covered faults of the initTS identi-

fied is computed using the stuck-at fault model, from the information obtained from the ATPG.

Numerical results are interesting, because they indicate that even if the coverage of the solution is overestimated, results are *robust* with respect to the required minimum coverage parameter. We underline how the meaningfulness of such numerical results is to be considered with respect to the level of abstraction of the fault models used to compute them.

Finally, in Table 3.5 we compare the optimal initTS derived using the teCTM model against the results of the execution of the ATPG, in terms of the number of test vectors and fault coverage, respectively. As expected, the ATPG fault model better fits with the problem (since more details are available with respect to the functional level of CTM). However, the strong correlations between the obtained results highlight how the proposed methodology well exploits the available information and produces good results in terms of test-set cardinality minimization. The advantage of the ILP methodology applied on teCTM models resides on the fact that it can be applied to scenarios where low-level fault models and fault/test coverage relation are not available or are only partially accurate.

3.3 Maximum coverage ordered initTS

Method proposed in Section 3.2 is focused at guaranteeing a *minimum coverage* for *all components* of the system under analysis. Scenario depicted in Figure 3.1 would benefit more, in some circumstances, of the minimization of the expected *time to (first) fail*. In particular, an effort of covering few components first would increase the *throughput* of the instance of the system in the testing line, by reducing the amount of time spent for discriminating faulty instances to be redirected to diagnosis. Therefore, an optimization approach aims at *maximizing* the coverage provided by the initTS, within the time-slot allocated for each system instance on the manufacturing line (Problem 2).

3.3.1 Hill-climbing test selection

Hill-climbing is a metaheuristic optimization technique, often applied for solving combinatorial optimization problems [PFT+07]. This technique based on the analysis of the neighborhoods of a set of possible solutions. New solutions are explored progressively changing a single element, accepting only configurations improving a specific cost function.

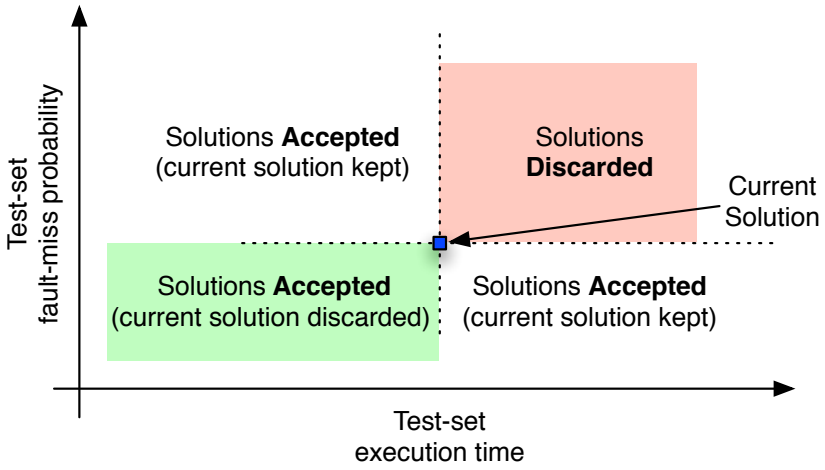


FIGURE 3.13: Accept-discard regions for hill-climbing search.

In our problem, we explore neighborhood of each test-set by randomly adding or removing a test; then, we compute both the cost and *un-detection* probability for the newly generated test-set. This is either *accepted* as a valid solution or refused according to the benefit criterium depicted in Figure 3.13. It is worth noting that newly created test-set lie in one of the three regions indicated. Therefore, a test-set that degrades both time and un-detection probability is discarded. If it improves both indicators, it discards the original solution. Otherwise, both original and new test-sets are kept as potentially optimal solutions.

From a group of randomly generated initial test-sets we apply a sequence of iterations of hill-climbing; only, taking care of distributing uniformly, among all possible test-sets sizes, the elements of the initial solution.

At the completion of the required number of iterations, candidate solutions are located on an *optimal curve*, over the *un-detection*-time plot. Figure 3.14 presents, for instance, all test-sets explored by the hill-climbing algorithm (red dots); also, it shows all test-sets which satisfy the optimality condition (blue curve). This curve is an approximation of the **CC!** curve introduce in the previous Section.

Even if test-sets located on the optimal curve provide an optimal *tradeoff* between test cost and benefit, such curve alone does not provide directly a methodology for extracting an optimal *sequence* of tests to be executed to minimize the cost of the scouting phase. Furthermore,

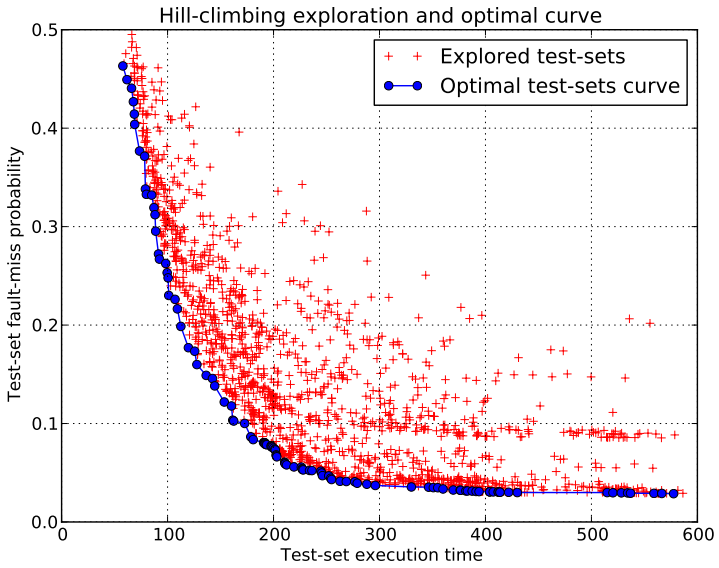


Figure 3.14: Test-sets analyzed during hill-climbing and optimal curve.

test-sets located on the curve could be composed of completely different tests, and this increases the complexity of the sorting problem.

To do this, we need to tackle some issues:

- the identification of a *sequence of test-sets*, all lying on the curve, so that all test-sets are *subsets* of successive ones. If this is condition does not hold, the execution of another test results in a test-set outside the optimal region.
- the *execution order* of the tests within a test; optimality of the curve takes into account the information obtained from executing all tests contained in the test set, but it does not specify if some order is more fit to scouting the first FAIL outcome.

It is worth underlining two properties:

- test-sets with a small number of tests are more likely to be located on the left side of the curve (low execution time), while larger test-sets on the right side;
- smaller test-sets are more likely to be included in larger test-sets (subset).

From an algebraic point of view, *subset-of* is a partial order relation; consequently, test-sets on the optimal curve can be represented using a

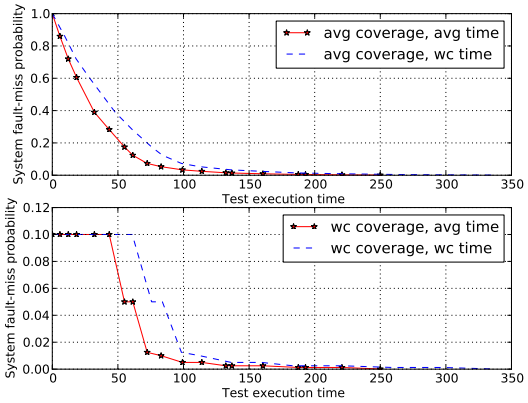


FIGURE 3.15: Optimal test order execution (exaustive search, SYS2.

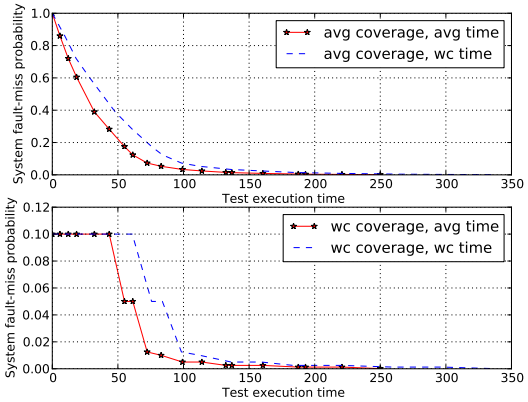


FIGURE 3.16: Optimal test order execution (Equation 3.5 used for system coverage).

Direct Acyclic Graph (DAG). Each node x of the graph represents a test-set, while an arc from node x to node y indicates that $x \subset y$.

Representing test-sets inclusion through a DAG allows the application of known algorithms from literature for the shortest-path identification. For instance, the Floyd-Warshall algorithm [Cor01] can be used to obtain, for each pair of nodes in the graph, the minimum cost path. In this case, we are interested in the *longest* path of the graph: this is more

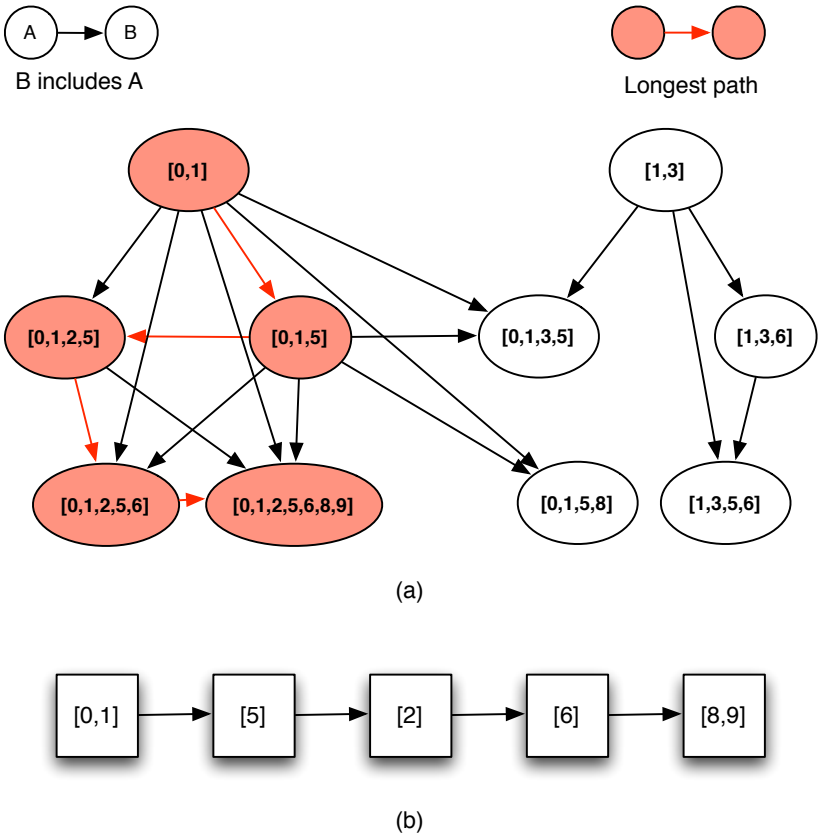


Figure 3.17: Inclusion graph (a) and optimal test sequence (b).

likely to be spread all-over the optimal curve and, because of this, it is more likely to provide the better execution order of tests minimizing the overall test-effort. We exploit the Floyd-Warshall algorithm setting a weight of -1 to each arc of the graph. Therefore, the minimum cost path corresponds to the longest path on the graph. Figure 3.17 (a) presents the inclusion relation among ten test-sets; also, it presents the minimum cost path on the graph (from [0,1] to [0,1,2,6,7,8,9]). Figure 3.17 (b) depicts the optimal test sequence extracted from the minimum cost path.

When this procedure is applied using as graph nodes all test-sets of the optimal curve, a sequence of test-sets is localized on the curve. Each blue point on Figure 3.18 represents a test-set (a node of a minimum cost path), and traversing the curve left to right is possible to increment the number of executed tests at minimum cost. It is worth noting that

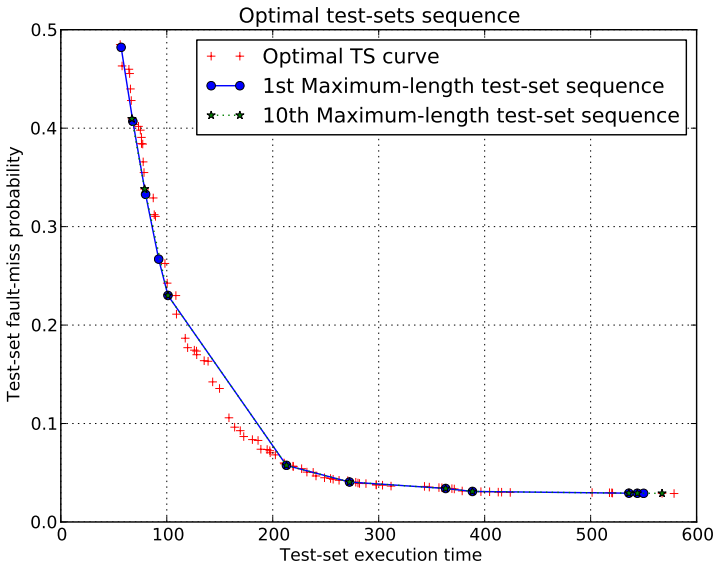


FIGURE 3.18: Optimal test-sets sequence on optimal test-sets curve.

if any other test is selected, the blue curve drifts outside the optimal region.

For sake of completeness, we need to define an execution order for tests subgroups, as $[0, 1]$ and $[8, 9]$ in Figure 3.17 (b). We sort tests within each group, executing first tests maximizing an *un-detection* probability *drop* (Figure 3.19). Given the reduced size of such test groups exhaustive exploration is sufficient to compute the optimal order in a reasonable time.

3.3.2 Analysis and experimental results

Experimental analyses have been executed on six systems, described through their CTM model.

SYS1 and SYS2 are two synthetic systems designed for research and analysis purposes. CTMs from CISCO2 to CISCO5 are models designed at Cisco for systems diagnosis, using the Automatic Fault Detective tool from Agilent [Agi04]. System CISCO4, even if claimed to be inaccurate for some components by test designer engineer, has been included in the set of system under diagnosis for comparison purposes. A database containing the information of all test sessions executed on each system

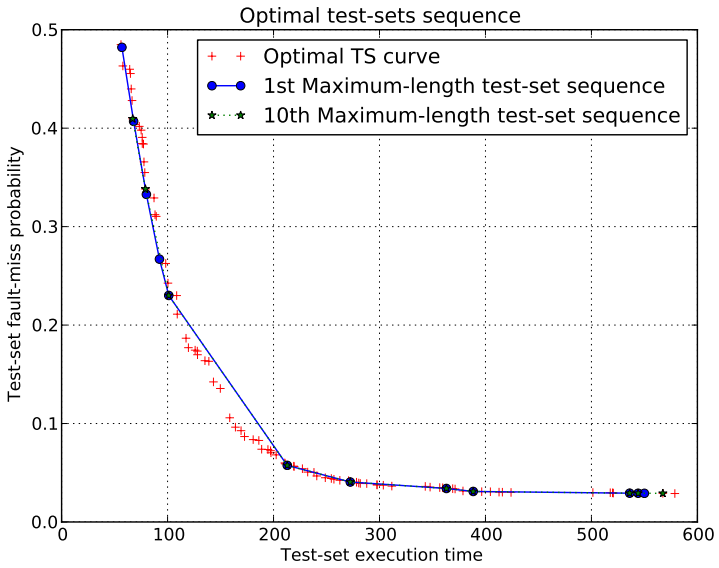


Figure 3.19: Fine sorting criterium of tests within a group.

Table 3.6: Summary of systems properties.

Sys.	**# Comp.**	**#Tests**	Test-suite time	# Sessions
Sys1	10	18	249.99	3000
Sys2	16	24	617.04	3000
Cisco2	32	50	3720.16	500
Cisco3	58	103	6011.93	1700
Cisco4	40	83	3162.14	780
Cisco5	55	112	7033.96	1500

is available. Each test session record contains all PASS, FAIL outcomes for all executed tests and tests execution; furthermore, it contains the diagnosed faulty component in the system.

Tab. 3.6 reports the characteristics of the models: the first two columns report the CTM size; column *Test-suite time* reports the execution time for all tests of the test suite (considering *average* execution time for PASS case only); column *# Sessions* reports the number of records of the test session database for each system under analysis.

First, let us consider the results of hill-climbing for the definition

Table 3.7: Hill-climbing report for optimal test-sets curve.

		100		150		200	
Sys.	**#InitTS**	#opt	#visit	#opt	#visit	#opt	#visit
SYS1	150	47	2228	50	3656	52	4830
SYS2	150	110	4540	121	8855	123	12466
CISCO2	400	177	10361	225	20087	242	29275
CISCO3	1000	189	15335	282	29863	300	46544
CISCO4	400	189	12861	212	18477	238	32821
CISCO5	1000	160	11383	193	22353	258	34216

of the optimal test-sets curve (Section 3.1). Tab. 3.7 reports, for each system, the number of test-sets randomly generated as initial solution. A larger number of test-sets has been chosen for systems with larger test suites, to minimize the probability of the algorithm to be stuck in local minimum. Also, the optimization has been repeated 10 times for each system, then a unique optimal curve has been extracted from the final results of each algorithm run.

Tab. 3.7 also reports the number of test-sets forming the optimal curve and the overall number of candidate solutions considered during optimization after 100, 150 and 200 iterations. Number of iterations but not time required for optimization is reported for this first implementation of the methodology; improvements will be part of future investigations. It is worth noting that the identification of the optimal test sequence is an offline process required to be performed only once when the CTM model of the system is defined.

In order to validate the metric selected to evaluate the fault coverage of the system in Section 3.1, used as a criterium for hill-climbing, we report in Tab. 3.8 a comparison between the value computed by our algorithm for each test-set and the respective coverage estimated by Agilent Fault Detective tool. Direct comparison can be done because Agilent tool takes as input a model compatible with CTM. In Tab. 3.8, each row reports the fault coverage of the test-sets composing the optimal curve, sorted in ascendent order. We observe that the coverage is relatively overestimated by our algorithm, especially for test-sets with low coverage.

Table 3.9 and Table 3.10 provide a performance evaluation of the proposed approach in terms of detection time for the first FAIL outcome,

Table 3.8: Current method and Agilent Tool coverage comparison.

	Cisco2		Cisco5	
Step	AF2D	Agilent	AF2D	Agilent
0	57.0%	45.1%	72.4	61.1%
1	64.1%	53.4%	75.9	68.2%
2	67.4%	59.8%	84.0	77.5%
3	83.3%	75.4%	85.7	80.8%
4	89.2%	79.3%	88.9	81.1%
		...		
(Last)	97.6%	86.9%	98.7	93.4%

for each system under analysis. Column 7 indicates the rate of success (*hit-rate*) of the initTS to detect (at least) one FAIL outcome; the test set was defined for all systems following the methodology proposed in Section 2.4, and requiring a 95% coverage for all components. The poor performance obtained on Cisco5 provided a confirmation of the remarks from test engineers, about the *low* quality of the system design. All other metrics are evaluated for successful sessions only.

In Column 2 we report the average detection time computed on all testing sessions. Same index is normalized over all systems in Column 5, where we report the *saved* test time with respect to the initTS, and in Column 6, with respect to the cost of the entire test suite.

It is worth noting that, with the single exception of Cisco4, the *average* improvement of a test sequence extraction over the entire execution of the initTS is appreciably in the range $[55\%, 75\%]$. Furthermore, the adoption of the *worst-case* test execution time is over-conservative, since some critical tests are not correctly taken into account during the identification of the optimal initTS: the performance gain drops, in average, of a significant 10% with respect to the adoption of the *average-case* for test execution time.

Columns 3 and 4 report, for both Tables 3.9 and 3.10, the 50− and 80− percentiles of the time to first FAIL distributions, since the average time is characterized by an important variance, where a majority of sessions has short detection time while remaining sessions reveal the first FAIL outcome almost at initTS completion.

Finally, Table 3.11 shows how a difference between *expected* and *actual a-priori* component probability distributions affects the performance of the test-set ordering algorithm. The 50- (Columns 2-4) and 80-

Table 3.9: First FAIL outcome detection time (using average-case test execution time).

		PERCENTILE		IMPROVEMENT		
Sys.	Avg.	50pc	80pc	Init	Suite	Hit-rate
SYS1	42.61	37.61	55.96	54.6%	85.0%	93.3%
SYS2	77.11	50.97	117.10	75.9%	87.5%	95.6%
CISCO2	167.10	144.15	284.59	55.1%	96.1%	89.8%
CISCO3	444.73	344.11	776.18	72.8%	94.3%	98.6%
CISCO4	171.42	152.26	289.85	47.5%	95.2%	72.9%
CISCO5	511.66	397.43	903.12	73.2%	94.3%	98.8%

Table 3.10: First FAIL outcome detection time (using worst-case test execution time).

		PERCENTILE		IMPROVEMENT		
Sys.	Avg.	50pc	80pc	Init	Suite	Hit-rate
SYS1	57.20	47.38	61.46	43.3%	81.0%	93,3%
SYS2	94.73	70.65	135.91	52.6%	88.5%	95.6%
CISCO2	209.65	187.42	311.20	43.6%	94.6%	89.8%
CISCO3	541.05	433.94	855.79	66.8%	92.7%	98.6%
CISCO4	212.24	195.79	324.27	35.0%	93.8%	72.9%
CISCO5	614.19	505.40	994.28	67.8%	92.8%	98.8%

percentiles Columns 3-5) are reported for two systems SYS2 and CISCO5. In the first row, all components are set to have a uniform probability distribution. Rows 2 to 6 report the behavior of such percentiles when a certain number of components (Column 1) is set to be 4 times more likely to be the *faulty component* than others. We observe a significant degradation of performance of the 80-percentile and an appreciable degradation for 50-percentile, especially for CISCO5, while the irregular behavior of the index on SYS2 can be explained by the reduced size of the system under test. The adoption of a correct *a-priori* is proven to be critical for the approach to produce an effective sequence of tests in order to increase the margin for a *cost* saving.

Table 3.11: First FAIL outcome detection time percentiles (non-uniform fault probability for components).

	Sys2		Cisco5	
# Comp.	50pc	80pc	50pc	80pc
0	50.97	117.10	397.43	903.12
1	52.84	120.90	402.19	1086.05
2	49.01	125.40	406.14	1112.06
3	47.75	129.51	445.46	1148.39
5	53.43	132.55	458.06	1156.74
8	54.95	144.83	470.56	1157.99

3.4 Chapter summary

In this Chapter, we have presented a method for the identification of an optimal initial test-set for diagnosis purposes, aimed at optimizing the probability to detected a fault in the initial phase of the process. We defined the optimality criteria for initial test-sets as minimum cardinality and maximum coverage. The problem has been solved as an Integer Linear Programming optimization; to validate the proposed method, a correlation was found with the properties of the solutions identified using a detailed fault-model and a modified ATPG tool. Experimental analysis has confirmed the validity of the results obtained with the method proposed.

Adaptive Test Selection 4

Once the presence of a failure in a system has been recognized, the reduction of the number of tests necessary for complete diagnosis is required in different scenarios. In general terms, the aim is at a *minimization* of the expected cost of test sessions involving each faulty instance of the system.

The entire suite of available tests, proposed by test engineers, is designed for covering the entire spectrum of expected failures, in order to obtain an accurate localization of any potential fault. Therefore, we propose to integrate an *adaptive* fault detective strategy, based on *selective execution* of a subset of available tests. The choice of such tests is made through a quantitative metric, aiming at the maximization of the diagnostic information inferred after the execution of each test.

In this chapter we describe the framework for an incremental exploration we proposed in [ABS+09], and we extend it with the analysis proposed in successive works; respectively, we aim at:

- the selection metric for the next test to be executed, with the objective to limit the number of executed tests by selecting only those actually adding non-redundant information for the identification of the faulty component [ABS10]; in other words, it is necessary to *anticipate* the execution of the *significant* tests, i.e. tests whose outcome is able to underline a better separation between potential faulty components in a system, and non-faulty ones.
- the definition of the stop criterion for identifying when to interrupt the diagnosis process, determining the faulty component on the basis of the collected information [ABSF10b]; the main idea is that,

at a certain point, the exploitation of additional test results would not contribute useful information to identify the faulty component, and thus the final diagnosis can be expressed without any further analysis.

The BBN model proposed in 2.2 is used to evaluate the actual benefit of each one of the not executed tests in reaching the correct diagnosis, in relation with the probability of their outcomes, to determine which test to run next. These elements are used to analyze how the BBN evolves when new test outcomes are available, after executing new tests, and if the outcome of further tests (not executed yet) would *substantially modify* the current Faulty Candidate Components (FCC) set (and the best faulty candidate identified in it).

The chapter is organized as follows: Section 4.1 introduces a context for the problem of the test session lenght reduction encountered in diagnosis methodology, presenting some significant related works. In Section 4.1 an analysis of the evolution of the BBN during an incremental diagnostic process is presented, together with the formalization of the problem in a vectorial space, leading to the introduction of the elements necessary to define the selection heuristics. Section 4.3 and Section 4.4 are respectively devoted to the description of alternative metrics for the *next test* selection and the *stop condition*. A set of experimental results is reported in Section 4.5 for the evaluation of the proposed approaches.

4.1 Background

The ordering or *sequencing* problem has been faced in the past in diagnosis literature, and the proposed algorithms are relatively different as the models used to describe both the system and the faults are not homogeneous.

For instance, in several works as [TP03,RTPPH04,KSC+08,BRdJ+09, BdJV+08a], the system under analysis is described using the *faulty signatures* model (Section 1.3.2). Kodali et al. in [KSC+08] propose to solve the problem formulating it as a dynamic *set covering* problem [Cor01], using a Lagrangian relaxation technique. Another technique is proposed in [RSP+99b], focusing on the minimization both of expected test time diagnostic ambiguity. It is formulated as an optimization problem and a solution through dynamic programming is found [RSP+99a]. Although those approaches propose *adaptive* testing policies, using the information of previously executed tests, their reliance on deterministic *faulty signatures* limits the presence of a failure to be always retrieved by the

execution of a specific test, and such constraint could limit the applicability of the methodology. This is true especially when the system description is available only at a high level of abstraction and deterministic relations between system faults and symptoms detected by testing cannot be established and maintained by test engineers.

Other works as [TP03, BdJV+08a] propose the transformation of the faulty signature representation graph structure (*AND/OR graph*). This structure is in a directed acyclic graph where the root represents the initial state of the diagnosis (where all components are potentially candidates for explaining the faults), while the leaves represents the diagnostic conclusions at the maximum resolution allowed by the test suite (diagnostic conclusions, Section 1.2). Intermediate nodes are used to represent either *tests* (associated with an execution cost) or *intermediate conclusions*. Arcs in the graph associate either the root and intermediate conclusion nodes to tests to be executed, or tests to the subsequent diagnostic conclusions depending on the test outcomes. A diagnosis strategy is a path traversing the graph and its cost is the sum of test nodes encountered. On this graph structure, Tu et al. in [TP03] propose an heuristic technique inspired on *dynamic programming* to identify the strategy *minimizing* the average cost to reach a diagnostic conclusion.

Their contribution has been further extended, taking into account hierarchical systems [BRdJ+09], to tackle problems where the relationship between components and tests cannot be described in a flat manner due to the complexity of the system under analisys; authors propose to solve the general problem as a composition of local optimizations, on portions of the system, where complexity can be handled. In this work, alternative heuristics to optimization are proposed, in order to face with the exponential complexity of the problem. Another extension is proposed in [RTPPH04], where authors modify the cost function of the test sessions in scenarios where the execution cost cannot be described as a sum (or another *linear* relation) of execution cost of single tests, but other contributions have to be taken into account (initial setup time, re-configuration time, or intermediate repair operations costs).

A key aspect for the incremental approach is the definition of a quantitative cost function to decide whether or not the execution of further tests is significant, i.e., whether the supplementary execution of a group of tests could modify the diagnosis outcome, leading to a different faulty candidate. We present such cost function and the associated *stop condition*, or criterion, to determine when the incremental diagnosis process

can be suspended, having identified the most probable faulty candidate. The bases for such a definition lie in the characteristics of the evolving BBN as new tests are executed and their outcomes are made available and introduced in the BBN. The next section introduces such aspects, together with the innovative cost function.

The proposed framework is based on the BBN model with the purpose of supporting the optimal next test selection, to minimize the overall test sequence length and, consequently, test execution costs. We first introduce here a few preliminary definitions and assumption, before introducing the formulation of the proposed metric.

We require few definitions to establish a mathematic framework, to be used for the formulation of the test-selection metric. We refer to a system model corresponding to a CTM (Section 2.2), composed of a set $\mathcal{C}$ with n_c components and with a test-suite $\mathcal{T}$ of n_t tests.

As long as new tests are executed, the *state* of the current instance of the system under analysis is described univocally through a *vector*, which contains the whole information obtained after test execution, summarized in test outcome, which is used to feed the BBN to perform system diagnosis.

Definition 15. A **partial syndrome** is a vector of n_t elements, $\mathbf{o}_i$, such that $\mathbf{o}_i \in \mathsf{O_E} = \{\mathsf{PASS}, \mathsf{FAIL}, \mathsf{UK}\}$, and $\exists k | \mathbf{o}_k = \mathsf{UK}$, that is test k has not been executed, the outcome UnKnown.

In this work, we do not consider explicitly the SKIP possible outcome; rather, we exclude from the vector representation the tests that cannot be executed, or we substitute it with their expected outcome (PASS or FAIL) when it can be uniquely inferred by other tests. When the outcomes of all tests are available, the instance status is described univocally by another vector.

Definition 16. A **syndrome** is a vector of n_t elements, $\mathbf{o}_i$, such that $\mathbf{o}_i \in \mathsf{O} = \{\mathsf{PASS}, \mathsf{FAIL}\}$.

Furthermore, an hypothesis is made to handle the complexity of the diagnosis, of the system, the *single fault* assumption.

Assumption 1. We assume a single component can fail at a time, and there is at least a test that fails.

In [ABF90], authors analyze the adoption of single fault hypothesis in digital circuits; there, while observing that this assumption appears to be *limitative*, they suggest that such hypothesis could be safely adopted for at least to important reasons. First, because it corresponds to a likely

real world scenario, at least from a *statistical viewpoint*. Second, because the multi-faults conditions are handled correctly by most of test suites designed to target single fault cases; this occurs because symptoms are observed from different probe points (e.g. circuit outputs), or failures are correlated and at least one of them appear in the final diagnosis.

We introduce this concept with the following Definition.

Definition 17. We define *singleton faulty distribution* a probability distribution of component nodes where a single component is a faulty candidate. The number of different possible *singleton faulty distributions* is equal to the number of components, n_c.

If Assumption 1 limits the valid probability distributions to respect Definition 17, a corollary follows: when a component is faulty all tests that do not cover it will PASS, whereas any one of the tests with a not null coverage may FAIL.

Corollary 1. Let $\mathbf{c}_x \in \mathcal{C}$ be the faulty component in system $\mathcal{S}$. Then

- $\forall \mathbf{t}_z$, if $\text{cov}(\mathbf{c}_x, \mathbf{t}_z) \in \{H, M, L\}$ then $\mathbf{t}_z$ may FAIL, and
- $\forall \mathbf{t}_{z*}$, if $\text{cov}(\mathbf{c}_x, \mathbf{t}_{z*}) \in \{-\}$ then $\mathbf{t}_{z*}$ must PASS.

We reinforce the capability of syndromes (Definition 16) to report correctly the presence of a failure introducing an additional assumption:

Assumption 2. Given a $\langle \mathbf{c}_x, \mathbf{t}_z \rangle$ pair such that $\text{cov}(\mathbf{c}_x, \mathbf{t}_z) = H$, the test is extremely unlikely to pass when the involved component is faulty.

Those assumptions, are required to avoid meaningless results from the diagnosis: such condition could arise if a *fault-free system* is analyzed with the methodology. In particular Assumptions 1 and 2 reinforce the characterization of some tests of the test suite to target at particular faulty components: this association is a leading factor during the definition of the system model CTM by the design and test engineering teams.

Furthermore, we observe also that, given the independence hypothesis between components' nodes implicitly provided by the NOR approach (see [Hen87]), there is no a-priori impossibility to manage a *multi-fault* diagnosis result, if a **syndrome** is *compatible* with two or several rows of the CTM. A consequence of this fact is that the *sum* of the probabilities of all components to be the faulty one, is not necessarily 1. Furthermore, a *normalization* of probability values does not produce a significant probability distribution, in particular when applied to **partial syndromes**.

4.1.1 Relations between syndromes and test selection

A deterministic relation exists between the information carried by a test outcome and the consequent update of the nodes probabilities resulting from probabilistic inference. This update depends only on the BBN topology, i.e., the coefficients of the CTM. Given a generic BBN, an exahustive *exploration* of all possible outcome combinations could be approached to determine, before test execution, the optimal sequence of leading to all possible diagnosis conclusions.

While deterministic, exhaustive explicit exploration is unaffordable because of the number of possible outcome combinations to be analyzed. Since the probability update depends only on the BBN topology, i.e., the coefficients of the CTM, we could resolve an implicit problem not considering the outcomes combinations, rather all bayesian inference equations in the form of a dynamic system. The optimization problem would be similar to the formulation proposed for test sequencing in [RSP+99a], while the stop condition would be formulated as a steady-state search. Unfortunately, inference equations [KP83] are extremely complex even for small 2-layers BBN, and the complexity grows exponentially with the number of nodes.

Since such partitions in the search space are defined through the conditional probabilities inference equations underlying the BBN, their properties should be derived analyzing directly those equations [Dar03]. Unfortunately, this analysis is not trivial because of the large number of equations involved in a usual BBN used for diagnosis, and, also, because of the number of terms involved in each equations. The problem is even harder if we consider that equations are dynamically changed in adaptive diagnosis, when outcomes of newly executed tests become available.

Alternative solutions have been proposed to overcome the difficulty of such analysis. A possible approach is based on the notion of *mutual information*, used for instance in [NJ98]. Such metric, derived from Information Theory and closely related to the concept of *entropy*, quantifies the impact of the insertion of new outcomes (newly executed tests) on a set of variables of the components' nodes in BBN.

This metric is a strong and well-known mathematical tool, especially for information compression; indeed, higher *entropy* values are associated with random sources, or in other words with sources where the amount of redundancy in the information is *low*. In a diagnostic scenario, lower entropy indicates redundancy in faulty component(s) identification, suggesting that further testing is little significant with respect to the quality of diagnosis. Authors in [TP03] propose to mimic Huffman's code prop-

erty to identify the minimum cost test outcomes set associated with each potential fault candidate. However, as it is explained in [RBM+05], *entropy* evaluation is costly for what concerns computational effort, and it results unaffordable for multi-valued coverage models as CTMs.

To overcome this limitation, a different approach is here proposed. It focuses on the identification of a non-decreasing function expressing the *plausibility* of a given component to be the faulty candidate. Such function is proposed to guide the refinement of the faulty candidates list at each step of the test sequence with a contained computational effort, while the non-decreasing property provides a basis for the definition of a *stop condition*.

4.2 Geometrical interpretation of the BBN state

In our BBN framework, new observations on nodes occur for test nodes only, corresponding to new test outcomes. An observation added on a component node has no meaning from the operational perspective; it would only be the final result of the diagnosis process. However, from a strict mathematical point of view, this operation can be performed, and, in particular, it is possible to insert observations describing a *singleton faulty distribution*. Because of the topology of the network the computation of the a-posteriori probabilities of test nodes is straightforward.

By selecting a target component $\mathbf{c}_x$, and setting its specific singleton distribution, the probability value of each test $\mathbf{t}_z$ to PASS, FAIL corresponds to the quantitative value of the coverage $\mathrm{cov}(\mathbf{c}_x, \mathbf{t}_z)$ defined in the CTM. Figure 4.1 depicts the a-posteriori inference of test probabilities for a sample BBN.

It is worth noting that this operation can be generalized by replacing the *singleton faulty distribution* with a *doubleton* or a generic *k faulty components* distribution. However, the number n_k of diagnosis to be considered increases (since each one has a different probability configuration); for a BBN with n_c components there are $n_k = \frac{n_c!}{(n_c-k)!k!}$ configurations for each k-faults acceptable diagnosis.

We propose a geometrical representation of the BBN status, providing a mathematical definition of the entities of our framework. We use as reference vector space $\mathcal{R}^{n_t}$, being n_t the number of test nodes contained in the BBN.

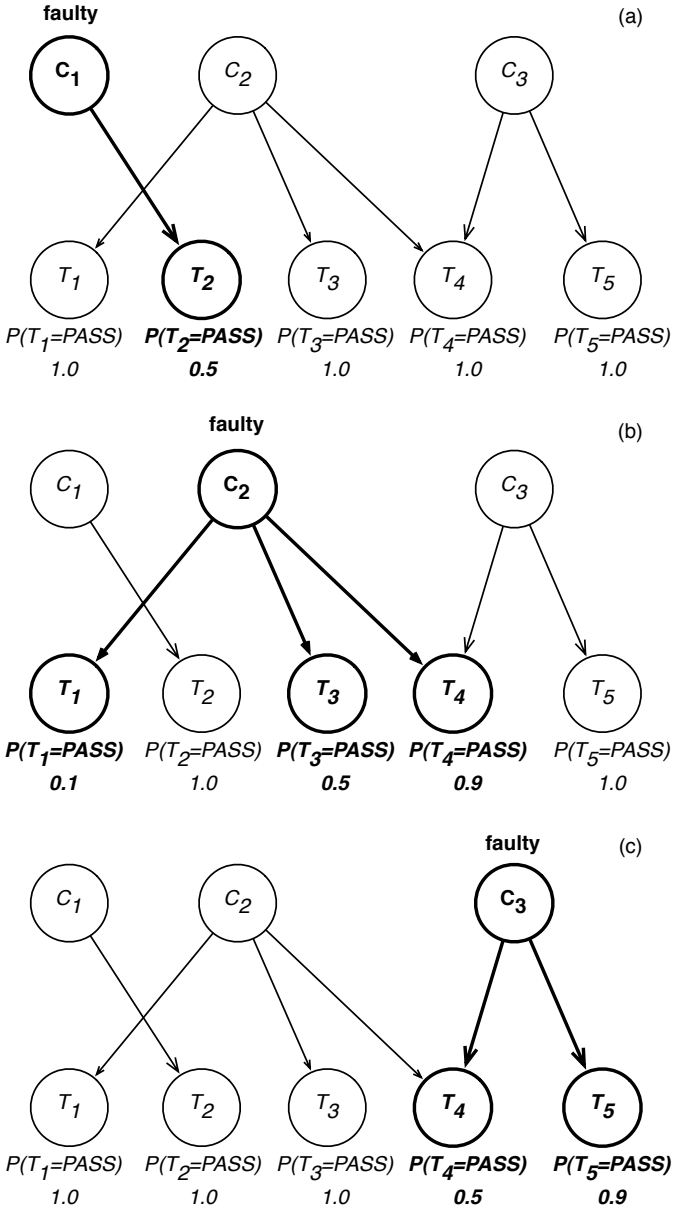


Figure 4.1: A-posteriori test probabilities inference (on singleton configuration $\mathbf{c}_1$ (a), $\mathbf{c}_2$ (b), $\mathbf{c}_1$ (c)), using sample BBN.

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$
$\mathbf{c}_1$	–	M	–	–	–
$\mathbf{c}_2$	H	–	M	L	–
$\mathbf{c}_3$	–	–	–	M	H

(a)

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$
$\mathbf{c}_1$	–	0.5	–	–	–
$\mathbf{c}_2$	0.9	–	0.5	0.1	–
$\mathbf{c}_3$	–	–	–	0.5	0.9

(b)

FIGURE 4.2: (a) Qualitative and (b) quantitative CTM for Example 14.

Definition 18. Let the **Complete Syndrome Vector ($\mathcal{Y}$)** be a $[s_1\ s_2\ \ldots\ s_{n_t}]$ vector of n_t elements, where $s_i \in \{\mathsf{I}, \mathsf{0}\}$, representing that the corresponding test outcome is PASS or FAIL, respectively.

Definition 19. Let the **Partial Syndrome Vector ($\mathcal{Y}_\mathcal{P}$)** be a $[\mathbf{o}_1\ \mathbf{o}_2\ \ldots\ \mathbf{o}_{n_t}]$ vector of n_t elements, where $\mathbf{o}_i \in \mathcal{R}$, such that $\mathbf{o}_i = \mathsf{I}$ when the test outcome is PASS, $\mathbf{o}_i = \mathsf{0}$ when the test outcome is FAIL, and $0 < \mathbf{o}_i < 1$ when the test has not been executed yet, respectively.

Given the relation between $\mathcal{Y}$, $\mathcal{Y}_\mathcal{P}$ and their corresponding **syndrome** and **partial syndrome**, it is possible to give also a mathematical definition of a test session.

Definition 20. Let $\mathcal{G} = \bigcup_{k\in[0,K]} g_k$ be a partition of the test set $\mathcal{T}$ of size n_t, where $\mathbf{t}_j \in g_k$ if and only if test T_j has been executed at step k. The value K depends on the partition $\mathcal{G}$ and it is always holds that $K \leq n_t$. Let $o_{j,s}$ be the outcome of test $\mathbf{t}_j$ at step s. It is always true that $o_{j,s}$ is UK for $s \in [0, k-1]$, while $o_{j,s}$ is PASS or FAIL for $s \in [k, K]$.

We can define a **Test eXecution session $\mathcal{TX}$** as the succession $\mathcal{Y}_{\mathcal{P},k}$, with $k \in [0, K]$. The beginning of the succession is the **Partial Syndrome Vector** $\mathcal{Y}_{\mathcal{P},0}$ corresponding to INITTS. The last element of the succession $\mathcal{Y}_{\mathcal{P},K}$ is equal to the **Complete Syndrome Vector ($\mathcal{Y}$)**.

Example 14. Consider the sample CTM of Figure 4.2, where $\mathcal{T} = \{\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \mathbf{t}_4, \mathbf{t}_5\}$, with $n_t = 5$. Let us consider a sample test set partition $\mathcal{G} = \{g_0, g_1, g_2\}$ with $g_0 = [\mathbf{t}_1, \mathbf{t}_5]$, $g_1 = [\mathbf{t}_3]$, $g_2 = [\mathbf{t}_2, \mathbf{t}_4]$, with $K = 2$. A $\mathcal{TX}$ is the succession $\mathcal{Y}_{\mathcal{P},k}$, with $k \in [0, 2]$, and a possible one is represented by
$\mathcal{Y}_{\mathcal{P},0} = \{\mathsf{FAIL}, \mathsf{UK}, \mathsf{UK}, \mathsf{UK}, \mathsf{FAIL}\}$,
$\mathcal{Y}_{\mathcal{P},1} = \{\mathsf{FAIL}, \mathsf{UK}, \mathsf{FAIL}, \mathsf{UK}, \mathsf{FAIL}\}$,
$\mathcal{Y}_{\mathcal{P},2} = \{\mathsf{FAIL}, \mathsf{PASS}, \mathsf{FAIL}, \mathsf{PASS}, \mathsf{FAIL}\}$.

Figure 4.3 presents the *evolution* of the test session with the succession $\mathcal{Y}_{\mathcal{P},0}$, $\mathcal{Y}_{\mathcal{P},1}$ and $\mathcal{Y}_{\mathcal{P},2}$. For clarity, the graphical representation considers only tests $[\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3]$.

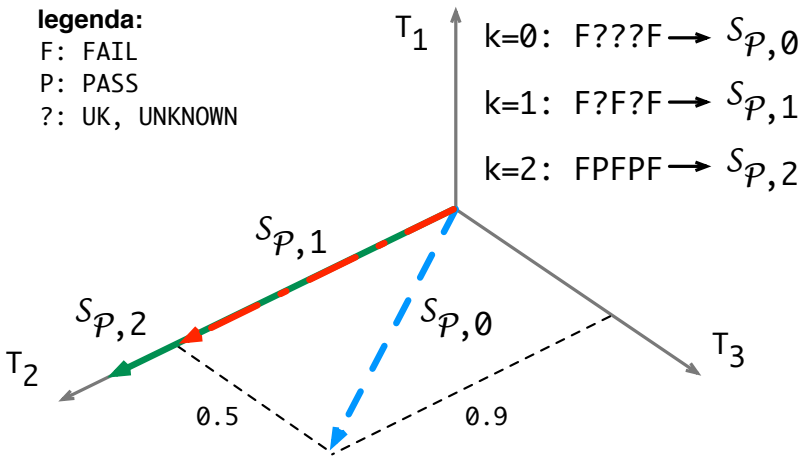


Figure 4.3: Test session evolution for Example 14.

◇

Definition 21. Let $\mathbf{v}$ be a $[v_1 v_2 \dots v_{n_t}]$ vector of n_t elements, where $v_j \in \mathcal{R}$. By recalling the notion of *distance in $\mathcal{L}^p$ spaces*, we can define the **distance** $d(\mathbf{v}^x, \mathbf{v}^y) \in \mathcal{R}$ between the vector pair $(\mathbf{v}^x, \mathbf{v}^y)$ as:

$$d(\mathbf{v}^x, \mathbf{v}^y) = (\sum_{j=1}^{n_t} (\|v_j^x - v_j^y\|)^p)^{1/p} \tag{4.1}$$

From this definition it is possible to derive the following weighted distance definition:

Definition 22. Let $\mathbf{v}$ be a $[v_1 v_2 \dots v_{n_t}]$ vector of n_t elements, where $v_j \in \mathcal{R}$. Let $\mathbf{w}$ be a weight vector $[w_1 w_2 \dots v_{n_t}]$ of n_t elements, where $w_j \in \mathcal{R}$. We can define the **weighted distance** $d_{\mathbf{w}}(\mathbf{v}^x, \mathbf{v}^y) \in \mathcal{R}$ between the vector pair $(\mathbf{v}^x, \mathbf{v}^y)$ with respect to the weigth vector $\mathbf{w}$ as:

$$d_{\mathbf{w}}(\mathbf{v}^x, \mathbf{v}^y) = (\sum_{j=1}^{n_t} w_j (\|v_j^x - v_j^y\|)^p)^{1/p}$$

Weighted distance has a general formulation but in our scenario we will define binary vectors for $\mathbf{w}$.

4.2.1 Attraction and Rejection

Partial syndrome vectors have a geometrical counterpart defined in the same vector space $\mathcal{R}^{n_t}$, defined in the following.

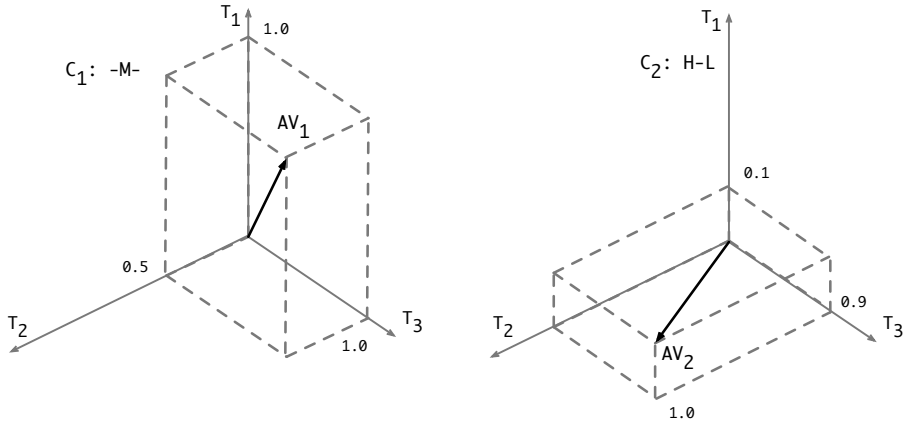


FIGURE 4.4: AVs for components $\mathbf{c}_1$ and $\mathbf{c}_2$ (for sample CTM in Figure 4.2).

Definition 23. We define Attraction Vector (AV) of component $\mathbf{c}_x$ ($\mathcal{AV}_x$) as vector $[s_1 s_2 \dots s_{n_t}]$ of n_t elements, where $s_z \in \mathcal{R}$, such that the value of s_z corresponds to the quantitative coverage of the CTM, i.e. $s_z = \text{cov}(\mathbf{c}_x, \mathbf{t}_z)$.

Given Assumption 2, we can define a dual concept for AV, dubbed Rejection Region (RR).

Definition 24. Let us consider coverage values $\text{cov}(\mathbf{c}_x, \mathbf{t}_z)$ of the CTM, relative to a specific component $\mathbf{c}_x \in \mathcal{C}$. Let $\mathcal{U}_y$ be a $[u_1 u_2 \dots u_{n_t}]$ vector of n_t elements, where $u_y \in \mathcal{R}$, such that the value of $u_y = 1$ if $\text{cov}(\mathbf{c}_x, \mathbf{t}_y) =$–, and $u_y = 0$ otherwise. We can define the Rejection Region (RR) of component $\mathbf{c}_x$ ($\mathcal{RR}_x$) the subspace of vectors $V = [v_1 v_2 \dots v_{n_t}]$ such that $V \cdot U_y = 0$ (operation $\cdot$ represents the standard internal product).

4.3 Next test selection heuristics

We propose four possible heuristics for the selection of the next test. Considering a generic test session $\mathcal{TX}$, the number of tests available for execution falls in range $[n_{its}, n_t]$, where n_{its} and n_t represent the size of the INITTS and the size of the test set $\mathcal{T}$. At each step k, we refer to $\mathcal{T}_r(k)$ as the set of remaining tests not executed yet; the size of such set is indicated with $n_{\mathcal{T}_r,k}$.

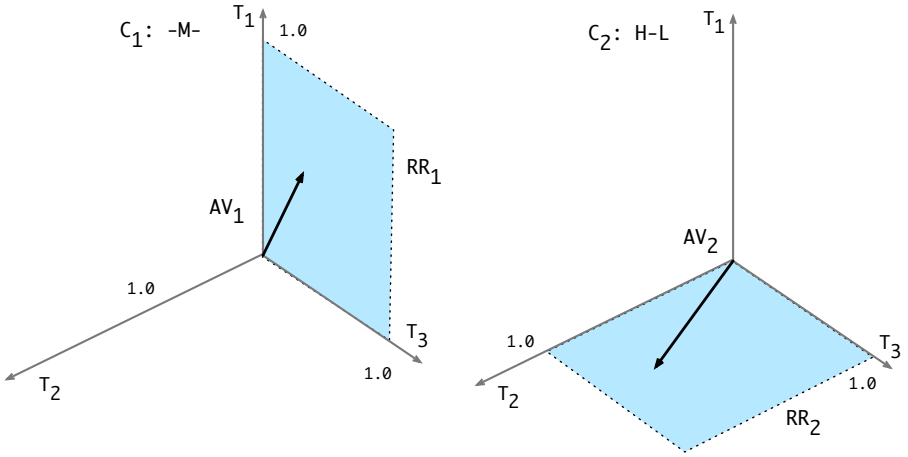


Figure 4.5: RRs for components $\mathbf{c}_1$ and $\mathbf{c}_2$ (for sample CTM in Figure 4.2).

4.3.1 Random Walk (RW)

As its name suggests, it is based on the *random* selection of the next test to be executed.

$$\mathcal{N}_{rand}(k) = rand(\mathcal{T}_r(k)) \tag{4.2}$$

This policy is not efficient because it discards completely the information contained in the outcomes of previously executed tests. Furthermore, it corresponds (in the geometrical interpretation) to an irregular path of Partial Syndrome Vector $\mathcal{Y}_\mathcal{P}$, and it does not guarantee any *regular* diagnosis in successive steps, and it limits the effectivness of the stop condition.

However, RW disposes of two interesting properties allowing us to consider it as a reference value for other heuristics. First of all, it represents an average case, so any other test selection policy should outperform the number of tests obtained using this approach. Second, it behaves the same in all possible CTM: this is important because if the test-component relationship is almost *one-to-one*, i.e. a fault on a component can be detected by a limited number of tests, and such tests cannot detect other faults in the system, information from previously executed tests is not significative, and any next test selection policy would behave exactly as the RW.

4.3.2 Variance (V)

The heuristic we proposed in work [ABS+09] is aimed at identification of the test whose execution will result in a probability distribution of the component nodes in BBN closer to a singleton distribution (Definition 17). The metric *compares* all possible $2 \cdot n_{\mathcal{T}_r,k}$ scenarios, obtained inserting one at the time all possible outcomes (PASS, FAIL) for each test in $\mathcal{T}_r(k)$.

Let us define vectors $\mathcal{C}_k(t_{\mathsf{PASS}})$ and $\mathcal{C}_k(t_{\mathsf{FAIL}})$, of size n_c (number of components), where item $\mathcal{C}_k(t_x)(c_i)$ contains the probability value of the component c_i obtained adding to the BBN respectively the PASS (or FAIL) outcome for test t_x. Different functions can be applied to vector $\mathcal{C}_k(t_x)$ to verify how similar it is to a singleton. We used the *sample variance*, since higher values correspond to a flat distribution with a small number of *spikes*, and as a consequence to a lower number of faulty candidates.

A priori, values of the metric evaluated for the same test t on $\mathcal{C}_k(t_{\mathsf{PASS}})$ and $\mathcal{C}_k(t_{\mathsf{FAIL}})$ can be different. Since it is impossible know the outcome of a test until its execution, it is necessary to associate the next test selection metric to the *expectation* $E[]$ of the variance.

$$\begin{aligned} \mathcal{N}_{var}(k) &= \underset{\mathbf{t}\in\mathcal{T}_r(k)}{\operatorname{argmin}} \quad \mathbf{E}[\mathcal{C}_k(\mathbf{t})] \qquad (4.3) \\ &= \underset{\mathbf{t}\in\mathcal{T}_r(k)}{\operatorname{argmin}} \quad \mathbf{E}[\mathbf{P}_k(\mathbf{t})var(\mathcal{C}_k(\mathbf{t}_{\mathsf{PASS}})) + (1-\mathbf{P}_k(\mathbf{t}))var(\mathcal{C}_k(\mathbf{t}_{\mathsf{FAIL}}))] \qquad (4.4) \end{aligned}$$

The approach requires the simulation of all possible outcomes for each test in $\mathcal{T}_r(k)$, and its complexity is linear with the number of remaining tests $n_{\mathcal{T}_r,k}$ ($(o(n_{\mathcal{T}_r,k}))$). The rationale behind this metric is to execute as soon as possible tests *moving* the BBN to a singleton, corresponding to a valid final diagnosis.

4.3.3 Failing Test First (FTF)

According to Assumption 2 and considerations set in Section 4.1, the *signature* of a faulty component is characterized uniquely by *failing tests*. Anticipated execution of such group of tests would shorten the overall test session length, because the faulty component would be identified sooner within the FCC set running all tests having coverage *High*.

Given $\mathbf{P}_k(\mathbf{t}_z)$ as the probability of the test $\mathbf{t}_z$ to pass at step k, we have:

$$\mathcal{N}_{fail}(k) = \underset{\mathbf{t}_z\in\mathcal{T}_r(k)}{\operatorname{argmin}} \quad \mathbf{P}_k(\mathbf{t}_z) \qquad (4.5)$$

The algorithm to compute this metric has a lower complexity ($o(1)$) than V, because it is based only on information obtained from the BBN with the update occurring after the insertion of each newly executed test outcome. However, it suffers from the limitation that component probabilities after the execution of the test are not taken into account, and this could lead the BBN to *fuzzy diagnosis*, increasing the number of tests required in successive steps for the fault detection. FTF heuristic is extremely efficient when the faulty candidate is clearly identified from the initTS or from the very first tests, especially if the number of tests covering this component is small. Otherwise, the execution of the selected test often results in a PASS outcome instead of a FAIL, producing a degradation of the efficiency of FTF.

4.3.4 Minimum Distance (MD)

Variance approach is based on the research of a singleton distribution on component probabilities, evaluated directly on an analysis of node values. As an alternative, this research can be done using the geometrical interpretation of BBNs, calculating the metric of test probabilities instead.

Let us indicate with $\mathcal{Y}_{\mathcal{P}k}(t_{\mathsf{PASS}})$ and $\mathcal{Y}_{\mathcal{P}k}(t_{\mathsf{FAIL}})$ PSV obtained inserting at step k into the BBN respectively the PASS (or FAIL) outcome for a test t in T_r. As for V, the algorithm needs to evaluate $2 \cdot n_{\mathcal{T}_r,k}$ different vectors.

Also, let us indicate with $\mathbf{dc}_{k,t}$ the most probable diagnosis conclusion identified after the insertion of the outcome of test $\mathbf{t}_z$, according to

$$\mathbf{dc}_{k,t} = \underset{\mathbf{c}_x \in \mathcal{C}}{\operatorname{argmin}} \quad \mathbf{P}_{k,\mathbf{t}_z}(\mathbf{c}_x) \tag{4.6}$$

This component is associated to an AV($\mathbf{dc}_{k,\mathbf{t}_z}$), according to Definition 23. The next test selection metric can be formulated as:

$$\mathcal{N}_{dist}(k) = \underset{\mathbf{t}_{z,\mathsf{PASS}},\mathbf{t}_{z,\mathsf{FAIL}} \in \mathcal{T}_r(k)}{\operatorname{argmin}} d(\mathcal{Y}_{\mathcal{P}k}(\mathbf{t}_z), AV(\mathbf{dc}_{k,\mathbf{t}_z})) \tag{4.7}$$

This heuristic is based on the minimization of the *expected* distance between the PSV and the AV of the most likely faulty component. The number of tests required to complete the diagnosis is reduced because at each step the BBN is moved towards a diagnosis having the higher probability to be the final one, being the distance between the PSV and the AV minimal.

A drawback of this heuristic is an increased computational effort. In fact, it requires both the simulation of all possible remaining tests (complexity is linear with $n_{\mathcal{T}_r,k}$, as for the *variance* approach) with the over-

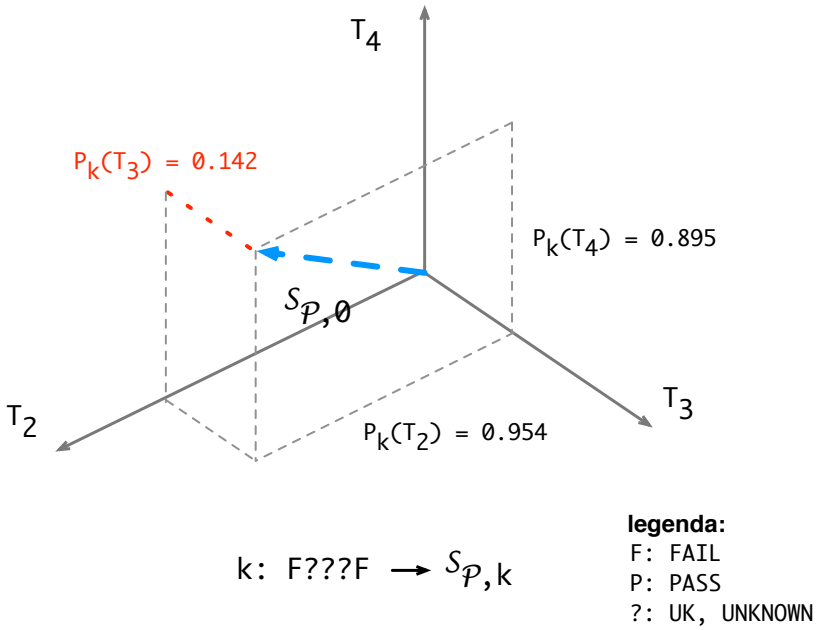


FIGURE 4.6: Geometric interpretation of heuristics FTF.

head of the estimation of the minimum distance $d(\mathcal{Y}_{\mathcal{P}k}(t), AV(fc_{k,t}))$, which is $o(n_c^2)$ on the number of components n_c.

Example 15. We here refer to SYS2, presented in the upcoming section, to compare the evolution of the same test session using different heuristics. After the execution of the INITTS ($k = 0$), we obtain the PASS outcomes for tests IDs **t**18,**t**5,**t**3,**t**2,**t**1 and **t**0. Also, tests IDs **t**17, **t**22, **t**23 failed, giving the component $\mathbf{c}_1$ (Micro_Processor_S) as best faulty candidate, with a probability 0.3208.

We compare in Table 4.1 the evolution of the four introduced heuristics, presenting the best *next test* identified by computing the metric at each step k. For each column, it is shown the ID of the selected test and the outcome resulting from its execution on the system. Also, the probability of the faulty candidate (Micro_Processor_S) is reported, evaluated by the BBN after the insertion of the new test outcome. The simulation is stopped at step $k = 9$, where the stop condition becomes true (for the *variance* heuristic).

◇

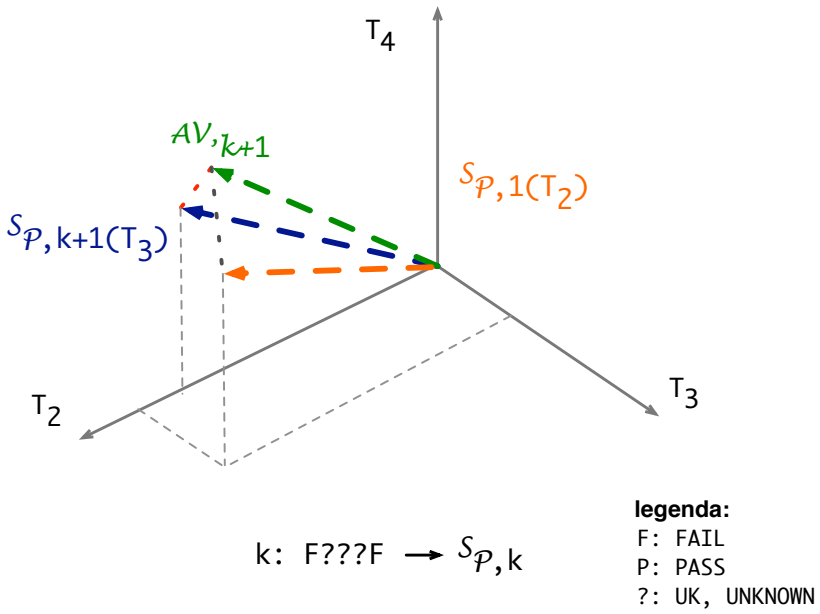


Figure 4.7: Geometric interpretation of heuristics MD.

Example 16. In Figures 4.7 and 4.6 we compare in the geometrical framework the different metrics minimized by FTF and MD. The selection is perfomed at step $k = 0$, using CTM presented in Example 15 to present BBN evolution. For clarity of representation, vectors are drawn in the test set subspace $[\mathbf{t}2, \mathbf{t}3, \mathbf{t}4]$. In this sample case, both metrics select $\mathbf{t}_3$ as optimal next test.

◇

4.4 Stop condition

The main goal of a stop condition metric is to identify the step $\hat{k}$ in a **Test eXecution session** $\mathcal{TX}$ such that the use of any further test outcome during step $k > \hat{k}$ can refine the component probabilities but *it does affect the identification of the best faulty candidate.* A secondary goal of such a metric is the definition of a non-decreasing function to quantify the level of *confidence* on the diagnosis result obtained with the identification of the faulty-candidate interrupting the execution of new tests at step $\hat{k}$. The former goal is simpler to achieve; the latter, instead, requires further refinements of the numerical properties of the metric function.

Definitions presented in the previous section will be used to introduce a metric function, to be applied at each step k of the Test eXecution session $\mathcal{TX}$ of the diagnosis process. The function at step k is computed by using the vector $\mathcal{Y}_{\mathcal{P},k}$, defined in $\mathcal{TX}$, all Attraction Vectors $\mathcal{AV}_x$ and Rejection Regions $\mathcal{RR}_x$.

The stop condition metric function $sc(k)$ is a scalar value ($sc(k) \in \mathcal{R}$) depending exclusively on step k. It can be evaluated for each component $\mathbf{c}_x$ through expression:

$$sc(k,i) = d(\mathcal{Y}_{\mathcal{P},k}, \mathcal{AV}_x) \tag{4.8}$$

The distance is evaluated, from the PSV at each step k of $\mathcal{TX}$ for all AVs. They act as *attraction entities* of the faulty candidate, and if Assumption 1 holds, PSV converges towards $\mathcal{AV}_x$ when the faulty component is exactly $\mathbf{c}_x$.

By discriminating executed tests from remaining ones, an alternative stop condition metric sc^* is presented in Equation 4.9. Two terms contribute:

$$sc^*(k,i) = sc_{\mathsf{UK}}(k,i) + sc_{\mathsf{PASS,FAIL}}(k,i) \tag{4.9}$$

where

$$sc_{\mathsf{UK}}(k,x) = d_{\mathbf{w}}(\mathcal{Y}_{\mathcal{P},k}, \mathcal{AV}_x) \tag{4.10}$$

$$sc_{\mathsf{PASS,FAIL}}(k,x) = \mathbf{W}(\mathcal{Y}_{\mathcal{P},k}, \mathcal{RR}_x) \tag{4.11}$$

The rationale behind Equation 4.9 is that *distance* is not independent of quantitative values associated with test outcomes (PASS, FAIL) and coverage labels (H,M,L). In particular, considering Corollary 2, the use of the FAIL outcome on a test $\mathbf{t}_j$ might be a penalization term if the (numerical) difference between FAIL and H is important. To overcome such a problem, in Equation 4.9 a *weight* mechanism is proposed.

Let us consider a boolean weight vector $\mathbf{w}$, as previously introduced, with $w_j = 1$ if test $\mathbf{t}_j$ has not been executed yet, and $w_j = 0$ otherwise.

Definition 25. Let $\mathbf{W}(\mathcal{Y}_{\mathcal{P},k}, \mathcal{RR}_x)$ be a penalty function proportional to the *number of* $\mathcal{Y}_{\mathcal{P},k}$ *components* intersecting hyperspace $\mathcal{RR}_x$.

The first contributions to the alternative sc^* metric (Equation 4.10) discards distance contributions for executed tests. The second term (Equation 4.11) introduces a penalization term for non-faulty components, that is suppressed from sc_{UK} due to the introduction of weight $\mathbf{w}$.

The metric is evaluated independently for each faulty candidate. It is worth noting that lower values for $sc^*(k,i)$ are associated with components $\mathbf{c}_x$ that are more likely to be among the best faulty candidates.

The stop condition can be formulated now as a logical predicate (TRUE, FALSE):

$$\mathcal{SC} : \min_x sc^*(k,x) < \delta \tag{4.12}$$

where the δ is a confidence interval, arbitrarily selected.

For the minimum step k verifying the stop condition $\mathcal{SC}_k$ = TRUE, the best diagnostic conclusion **dc** can be identified from the stop condition metric by using:

$$\mathcal{SC}_k \implies \mathbf{dc} = arg\min_x sc^*(k,x) \tag{4.13}$$

Example 17. We here refer to Sys2, presented in the upcoming section, to show the contribution of the distances on the defined metric. After the application of the initTS, four components constitute the FCC set, namely $\mathbf{c}_0$, $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_8$. For each one of them, at each step k of the diagnostic process we report in columns 3-6 the associated probability to be the faulty candidate ($\mathbf{P}(\mathbf{c}_x)$), and the distance metric between each component's AV (columns 7-10), where $D(\mathbf{c}_x)_k = d(\mathcal{Y}_{\mathcal{P},k}, \mathcal{AV}_x)$. At each step k, we also report the suggested next test to be executed (column 2), and the evaluated stop condition SC_k using $\delta = 2$ (column 11). We highlighted the highest probability values and the smallest distance values.

In the first three steps, probability values identify $\mathbf{c}_8$ as the best faulty candidate, however its estimated distance is not the minimal one, therefore the diagnosis is still uncertain. At step $k = 4$, $\mathbf{c}_0$ is identified, but again the distance does not support such a hypothesis. Then, the evidence introduced by $\mathbf{t}4$ leads to a convergence toward $\mathbf{c}_1$, not only from the probability point of view, but also on the distance metric, being it the component with the smallest estimated distance w.r.t. the partial syndrome. Nevertheless, the stop criterion is not met yet, and the overall distance distribution has a value higher than the selected δ, condition met at step $k = 9$, when the process is halted. $\mathbf{c}_2$ is listed as an example of components never considered as faulty candidates.

◇

	RW			FTF			V			MD		
Step	$\mathbf{t}_{ID}$	Out.	Prob.	$\mathbf{t}_{ID}$	Out.	Prob.	$\mathbf{t}_{ID}$	Out.	Prob.	$\mathbf{t}_{ID}$	Out.	Prob.
1	9	P	0.190	16	F	0.180	16	F	0.180	9	P	0.190
2	8	P	0.210	15	P	0.089	19	P	0.097	15	P	0.097
3	20	P	0.125	21	P	0.046	12	P	0.105	16	F	0.046
4	16	F	0.060	8	P	0.052	11	P	0.114	8	P	0.052
5	10	P	0.065	20	P	0.028	6	P	0.124	19	P	0.028
6	6	P	0.070	9	P	0.017	9	P	0.068	20	P	0.017
7	12	P	0.088	4	P	0.010	20	P	0.040	21	P	0.014
8	11	P	0.081	13	P	0.009	10	P	0.043	13	P	1.010
9	7	P	0.075	19	P	0.009	8	P	0.047	14	P	0.010

Table 4.1: Sample test session for Sys2.

Table 4.2: Sample test session.

k	Next Step	FCC Probability				Distance values				$\mathcal{SC}_k$
		$\mathbf{P}(\mathbf{c}_0)$	$\mathbf{P}(\mathbf{c}_1)$	$\mathbf{P}(\mathbf{c}_2)$	$\mathbf{P}(\mathbf{c}_8)$	$D(\mathbf{c}_0)$	$D(\mathbf{c}_1)$	$D(\mathbf{c}_2)$	$D(\mathbf{c}_8)$	
0	**t**3	0.007	0.001	0.001	0.010	7.951	3.604	2.785	6.268	FALSE
1	**t**17	0.068	0.055	0.001	0.857	7.966	3.566	2.795	6.278	FALSE
2	**t**23	0.037	0.030	0.001	0.934	7.984	3.606	2.798	6.293	FALSE
3	**t**14	0.550	0.440	0.001	0.062	5.934	3.584	16.570	19.642	FALSE
4	**t**4	0.125	0.878	0.001	0.027	4.957	3.506	20.515	23.699	FALSE
5	**t**22	0.019	0.985	0.001	0.020	14.243	3.566	34.318	37.701	FALSE
6	**t**1	0.019	0.985	0.001	0.020	14.630	3.430	34.700	37.702	FALSE
7	**t**18	0.007	0.997	0.001	0.019	14.706	2.662	34.700	37.799	FALSE
8	**t**2	0.007	0.997	0.001	0.019	14.804	2.293	34.700	37.901	FALSE
9	**t**8	0.005	0.999	0.001	0.019	14.901	1.632	34.700	37.900	TRUE
10	**t**21	0.005	0.999	0.001	0.019	14.944	1.628	34.700	37.900	TRUE
11	**t**6	0.004	0.999	0.000	0.019	14.998	1.626	34.700	37.900	TRUE

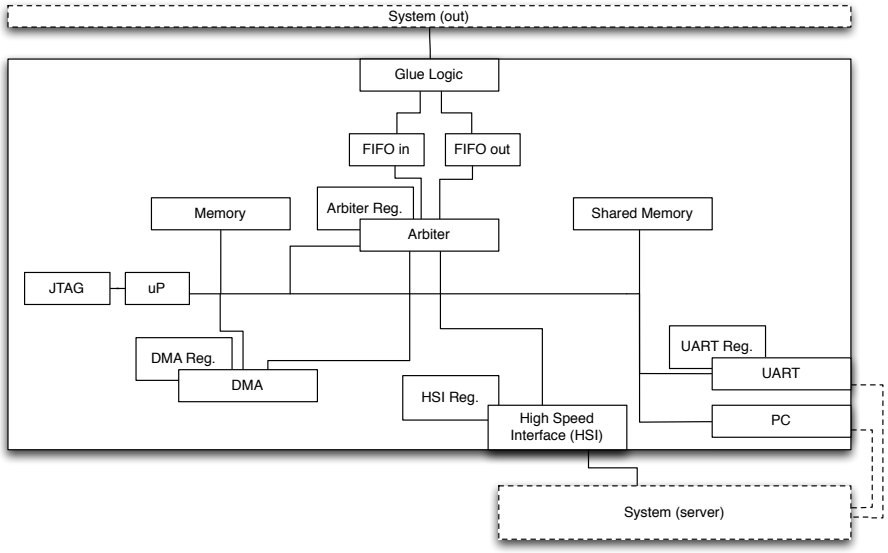


FIGURE 4.8: SYS2 block diagram.

4.5 Experimental results

4.5.1 Next test selection

Proposed heuristics have been coded as Python extension in the AF2D framework (Section 2.4), and they have been evaluated on four case studies. The properties of each system under analysis, named SYS1 to SYS4, are summarized in Table 4.3: number of components, number of available tests, number of tests of the INITTS, and the ratio between the size of INITTS and the number of available tests. SYS2 in particular is a sample system characterized by some hard-to-diagnose components, i.e. components which require the execution of almost all tests in order to obtain correct diagnosis.

System name	# Comp.	# Tests	# INITTS (%)
SYS1	10	18	6 (33%)
SYS2	18	24	9 (37%)
SYS3	27	80	9 (11%)
SYS4	49	100	17 (17%)

Table 4.3: System under analysis properties summary.

Comparison of the metrics in the framework have been executed computing the number of tests required to reach the stop condition. For each faulty component (of each system from Sys1 to Sys4), a list of Complete Syndromes has been obtained. Such syndromes have been used to feed our algorithm in a step-by-step fashion, simulating the incremental approach to analyze different test sequences.

All available Complete Syndromes have been tested against our three *next test selection* approaches (*Failing Test First (FTF)*, *Variance (V)*, *Minimum Distance (MD)*), together with a realization of the *Random Walk (RW)*.

As an example, we report in Table 4.4 a comparison among the average number of tests required to reach the stop condition, grouped by faulty components for a subset of components in Sys2. The number of available *Complete Syndromes* is reported (# CSyn), together with the average steps number for each heuristic, without considering the initTS (columns 2-5) and considering it (columns 6-9).

The reduction of the expected length for a test session is reported in Table 4.5. Two significative values are reported for each system under test: in columns 2 to 5 the ratio between the number of executed tests and available tests, considering only tests run *after* the initTS. Columns 6-9 report the same ratio, considering also the initTS cardinality in the number of executed tests.

Finally, we compare in Figures 4.9 and 4.10 the average test sequence length for Sys1 to Sys4, normalized to the length of RW, ignoring and considering the initTS respectively. It is worth noting that all approaches produce a significative reduction of the number of tests required for diagnosis, and that the impact of the sequence length reduction is greater when considering larger systems, with an wider choice of possible tests. *Minimum Distance* approach has better performance, on average, on Sys 1, 3, 4, at the price of an higher computational cost.

Results for Sys2 are characterized by a better performance for the *Variance* heuristic. This was expected because the test set for this system was designed to contain *hard-to-diagnose* components, requiring a larger number of test outcomes before correct final diagnosis. Since this hard diagnosis implies a short distance between the AV, the *Minimum Distance* approach suffers of such configuration.

4.5.2 Stop condition

The proposed metric has been evaluated on some case studies. Each network describes a system, containing 10, 16 and 28 components, respectively, with a test set of 18, 24 and 72 tests. Sys2 is a demo system,

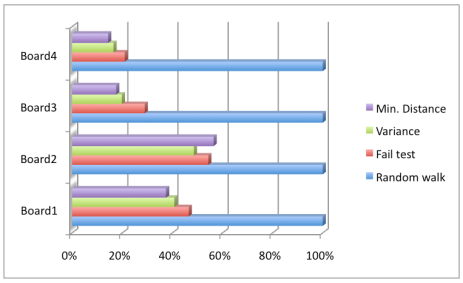


FIGURE 4.9: Test session length (normalized to RW, non considering INITTS).

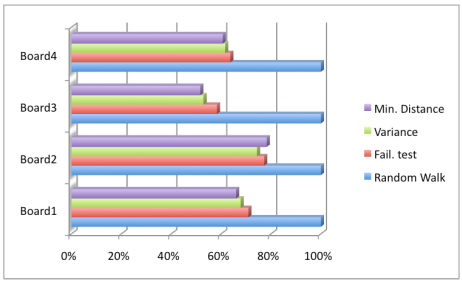


FIGURE 4.10: Test session length (normalized to RW, considering INITTS).

and its model is used to tune algorithm parameters. SYS2 is a sample microprocessor-based board, and CISCO1 is obtained from the analysis of a industrial network routing device by Cisco Photonics. The cardinalities of the Initial Test Set (INITTS) are 2, 7 and 10, respectively. The evaluation criteria of the metric is based on the comparison of the stop condition (during the incremental insertion of test outcomes) against the situation where the test set is integrally applied.

In particular, the evaluation has been done by comparing i) the final faulty candidate identified by the stop condition (and its plausibility) with respect to the real faulty component, and ii) the partial syndrome diagnosis result against the result derived from the complete syndrome.

In Table 4.6, 4.7 and 4.8 (showing data on SYS 1, 2 and CISCO1, respectively) the number of analyzed syndromes for each faulty component are reported, together with the average sequence length before the stop condition is reached, *evaluated with respect to the complete test set size*. Number of components magnitude is significative: it represents the number circuit locations (containing several elementary components) where a failure can be identified.

For instance, in Table 4.6, component $\mathbf{c}_0$ being the faulty one is associated with 256 possible syndromes (column *Seq.*) and the average number of test executed before the stop condition is reached is 10.42% (column *Avg.*). It is worth noting that the number of possible syndromes is proportional to the number of Medium labels in the row of the CTM of the faulty component (in this case, 5 M labels generate $2^5 = 32$ possible syndromes).

An unbiased average (column *Unbias Avg.*) is also reported, to compare the real test sequence length considering also the tests in the initTS (in the example, 2.94%).

The last two columns contain the information about the correctness of the final answer, evaluated at the stop condition and at the complete syndrome, for all simulated experiments.

Exhaustive analysis highlights that, by using the complete test set, the faulty candidate identification is correct in all cases for Board 1, 14 times out of 16 for Sys2 and in 25 out of 29 cases for Cisco1. The proposed stop condition fails in 2/15 situations for Sys2, and in 5/16 for Cisco1. For Board 2 (holding also for Cisco1), the evaluation failures (13.28% for $\mathbf{c}_1$, 25% for $\mathbf{c}_2$) are due to modeling problems; in particular, the current model presents two *attraction components* that dominate $\mathbf{c}_1$ and $\mathbf{c}_2$ in some particular conditions (limited *isolation power* of the CTM). This modeling problem is analyzed in Section 5.3.

		Without initTS				With initTS			
Faulty Comp.	# CSyn	RW	FTF	V	MD	RW	FTF	V	MD
Micro_Processor_S	148	12.07	4.98	4.68	11.08	21.07	13.98	13.68	20.08
Memory_S	149	12.48	4.49	4.67	4.76	21.48	13.49	13.67	13.76
Shared_Memory_S	150	3.33	2.51	2.00	2.47	12.33	11.51	11.00	11.47
Arbiter_S	133	12.80	11.52	9.82	10.26	21.80	20.52	18.82	19.26
Arbiter_config_Registers_S	150	1.00	1.00	1.00	0.95	10.00	10.00	10.00	9.95
DMA_S	125	12.58	10.14	8.78	9.14	21.58	19.14	17.78	18.14
DMA_config_Registers_S	150	8.55	1.00	1.00	1.01	17.55	10.00	10.00	10.01
High_Speed_Interface_HSI_S	135	12.30	8.74	4.73	8.10	21.30	17.74	13.73	17.10
HSI_config_Registers_S	150	6.69	1.00	1.00	0.97	15.69	10.00	10.00	9.97
UART_S	150	6.39	2.49	2.00	2.41	15.39	11.49	11.00	11.41
UART_config_Registers_S	150	5.22	1.00	1.00	0.92	14.22	10.00	10.00	9.92
I2C_S	150	8.36	2.35	1.89	2.34	17.36	11.35	10.89	11.34
FIFO_out_S	150	13.92	6.64	6.00	6.56	22.92	15.64	15.00	15.56
FIFO_in_S	150	8.71	2.00	3.00	1.91	17.71	11.00	12.00	10.91
Glue_Logic_S	150	16.00	16.00	16.00	16.00	25.00	25.00	25.00	25.00
JTAG_S	150	2.89	2.00	2.00	1.99	11.89	11.00	11.00	10.99
Average		8.96	4.87	4.35	5.05	17.96	13.87	13.35	14.05

Table 4.4: AF2D results (average test number to correct diagnosis) for Sys2.

	Without initTS				With initTS			
Board	RW	FTF	V	MD	RW	FTF	V	MD
Sys1	59.61%	27.73%	24.28%	22.33%	73.07%	51.82%	49.52%	48.22%
Sys2	59.71%	32.44%	28.99%	33.69%	74.82%	57.78%	55.62%	58.56%
Sys3	18.55%	5.37%	3.67%	3.25%	27.25%	15.88%	14.42%	14.06%
Sys4	17.44%	3.65%	2.87%	2.50%	31.47%	20.03%	19.39%	19.08%

Table 4.5: Test session length reduction (%).

Table 4.6: Experimental results for SYS1.

Comp	Seq.	Avg.	Unbias Avg.	Stop Cond.	Complete
$\mathbf{c}_0$	256	10.42%	2.94%	100%	100%
$\mathbf{c}_1$	8	28.12%	27.94%	100%	100%
$\mathbf{c}_2$	8	12.50%	5.88%	100%	100%
$\mathbf{c}_3$	8	31.25%	32.35%	100%	100%
$\mathbf{c}_4$	8	37.50%	41.18%	100%	100%
$\mathbf{c}_5$	8	39.58%	44.12%	100%	100%
$\mathbf{c}_6$	8	33.34%	35.29%	100%	100%
$\mathbf{c}_7$	8	12.50%	58.82%	100%	100%
$\mathbf{c}_8$	8	22.91%	20.59%	100%	100%
$\mathbf{c}_9$	256	12.50%	5.88%	100%	100%
	576	32.08%	23.59%		

Table 4.7: Experimental results for SYS2.

Comp	Seq.	Avg.	Unbias Avg.	Stop Cond.	Complete
$\mathbf{c}_0$	2	39.58%	14.71%	100.00%	100.00%
$\mathbf{c}_1$	256	56.97%	39.25%	86.72%	100.00%
$\mathbf{c}_2$	4	32.29%	4.41%	75.00%	100.00%
$\mathbf{c}_3$	64	66.54%	52.76%	92.19%	92.19%
$\mathbf{c}_4$	1	29.17%	0.00%	100.00%	100.00%
$\mathbf{c}_5$	256	72.05%	60.55%	82.03%	82.03%
$\mathbf{c}_6$	1	41.67%	17.65%	100.00%	100.00%
$\mathbf{c}_7$	8	52.08%	32.35%	87.50%	87.50%
$\mathbf{c}_8$	1	29.17%	0.00%	100.00%	100.00%
$\mathbf{c}_9$	4	37.50%	11.76%	100.00%	100.00%
$\mathbf{c}_{10}$	1	29.17%	0.00%	100.00%	100.00%
$\mathbf{c}_{11}$	2	41.67%	17.65%	100.00%	100.00%
$\mathbf{c}_{12}$	2	85.42%	79.41%	100.00%	100.00%
$\mathbf{c}_{13}$	2	33.33%	5.88%	100.00%	100.00%
$\mathbf{c}_{14}$	4	72.92%	61.76%	100.00%	100.00%
$\mathbf{c}_{15}$	1	33.33%	5.88%	100.00%	100.00%
	609	47.05%	25.25%		

Table 4.8: Experimental results for Cisco1.

Comp	Seq.	Avg.	Unbias Avg.	Stop Cond.	Complete
$\mathbf{c}_0$	2	27.77%	16.12%	100.00%	100.00%
$\mathbf{c}_1$	256	52.77%	45.16%	89.06%	89.06%
$\mathbf{c}_2$	4	20.83%	8.06%	100.00%	100.00%
$\mathbf{c}_5$	256	51.38%	43.54%	94.92%	100.00%
$\mathbf{c}_8$	128	26.38%	14.52%	86.72%	86.72%
$\mathbf{c}_9$	4	26.38%	14.52%	100.00%	100.00%
$\mathbf{c}_{10}$	4	37.5%	27.42%	100.00%	100.00%
$\mathbf{c}_{11}$	2	40.27%	30.65%	100.00%	100.00%
$\mathbf{c}_{12}$	32	55.55%	48.38%	87.50%	100.00%
$\mathbf{c}_{13}$	2	48.61%	40.32%	100.00%	100.00%
$\mathbf{c}_{14}$	4	52.77%	45.16%	100.00%	100.00%
$\mathbf{c}_{15}$	2	50%	41.93%	100.00%	100.00%
$\mathbf{c}_{18}$	4	40.278%	30.64%	100.00%	100.00%
$\mathbf{c}_{19}$	8	55.56%	48.39%	87.50%	100.00%
$\mathbf{c}_{20}$	256	55.56%	48.39%	91.80%	91.80%
$\mathbf{c}_{21}$	32	15.28%	1.61%	96.87%	100.00%
	1151	39.89%	30.20%		

4.6 Chapter summary

In this Chapter, we have tackled a two-fold problem, the identification of a quantitative function for the choice of the test to be executed for the characterization of a stop condition for the AF2D diagnosis methodology.

Because of the complexity of the analysis of the evolution of the underlying BBN, a geometrical interpretation of the model is proposed, aiming at identifying and defining the convergence points, dubbed Attraction Vectors, characterizing the evolution of the syndromes in a vector space; therefore, introducing a distance-based metric allows to quantify the benefit of not-yet executed tests, at each step of a diagnosis process. Three different heuristics are proposed and compared with random selection to resolve the next test selection problem. A supplementary function is then introduced to characterize the triggering of the *condition* where it is more significant to stop the adaptive investigation.

Analysis on some synthetic networks and industrial test cases has been carried out, to gather results on the validity of the proposed functions, with promising outcomes.

All approaches targeting automatic fault diagnosis require a relevant amount of knowledge about the relationship between faults, their symptoms and produced syndromes, irrespectively if they expressed in terms of rules [Abr05], models [HCdK92], or cases [Der00]. This knowledge is capital to identify and localize the occurrence of faults through the application of test patterns.

Furthermore, difficulties arise from imperfections of both the model of the device and the human experts' knowledge about the model, leading to incorrect and incomplete models, leading to inadequate diagnostic performance.

From this perspective, Bayesian Belief Network (BBN) provide a good resource to model the causes (components) and effects (test results) and their relationship, also considering the accuracy of the information that design and test engineers can provide, explaining how a failure in a component affects the performed tests.

In fact, BBNs mitigate the impact of the *imperfection* of the model, at least to a certain extent. Nevertheless, we deem it important to evaluate the *sensitiveness* of the diagnostic process and success, to the elements at the basis of the adopted model, that are:

i) the model *consistency*, derived from the understanding of the system behavior by the test engineer and his ability to model the causal relations from component failures to test outcomes,
ii) the *robustness* with respect to occasional, non-systematic mistakes in the model definition, and
iii) the intrinsic *diagnostic resolution* of the adopted approach, i.e. the

ability of the strategy to identify correctly the faulty component.

The aim of the contribution proposed in this Chapter is to present an in-depth analysis of *consistency* and *robustness* of the causal model at the basis of the diagnostic strategy based on BBNs.

Indeed, the correct specification of the relation between faults and test coverage and test outcome is of capital importance to make the causal model consistent with the syndromes collected during diagnosis. This is so because of the deductive nature of the approach: diagnosis can be interpreted as a search problem where the solution space is an a-priori, defined, partition of all possible diagnoses. However, as we observed during the development of an adaptive test selection policy in Section 2.2, the relevant number of equations beneath the a BBN model used for diagnosis makes such analysis not trivial; also, its adaptive nature of an evolving BBN model poses an even harder problem.

We will analyze how deviations from the correct, *golden* causal model affect the diagnosis, using a scenario of combinatorial circuits affected by the classical stuck-at fault, and exploiting a classical ATPG to derive the golden model. However, it is worth noting that we will abstract the key-concepts of the diagnostic process such that both aspects could be easily be generalized and extended to

- different causal models at the basis of a diagnostic approach exploiting BBNs (e.g., the one used in [Agi04]),
- different test suites, or even
- different fault models.

We target a sensitivity analysis of CTM model parameters, and the robustness of diagnostic results, under the assumption that a *non-systematic* mistake in coverage labels selection is committed by the test engineer. This entails that we need a solid approach to obtain the *real* values of probabilities of BBN (a *golden model* for BBN), and a method to generate *varied* causal models representing impreciseness and local mistakes. The golden model will then be used as a comparison against with the varied CTM models, and with the one that designed by a test engineer (Figure 5.1).

Concerning the *intrinsic* diagnostic resolution of a test suite, it can be quantified as function of the relative *coverage* that a test can provide with respect to each component. The simplification toward a test PASS, FAIL outcome synthesis is commonly adopted to simplify the diagnostic task, and it makes the diagnostic process harder for several reasons. In fact, as discussed in [LCLH98], the faulty model must be flexibile enough to embed all information extracted from the execution of the tests, with respect to the failure occurring in the system under anaysis. Furthermore, as in the case of PASS, FAIL faulty-dictionary methodologies

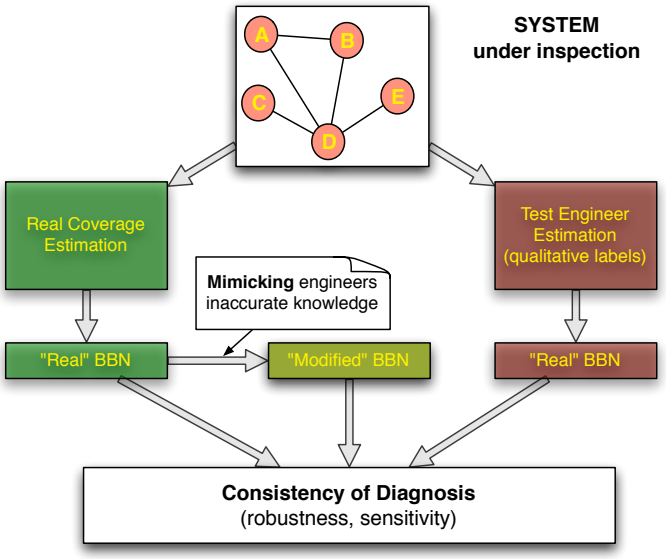


FIGURE 5.1: Quality assessment of CTM models.

[RB85], causal approaches may suffer of poor accuracy in identifying the faulty component, due to the extreme compression of the information contained in the test outcome, reducing the description of the behavior of the system to a boolean variable.

In particular, algorithms have been investigated to determine single-fault set diagnosability with respect to the models the authors refer to (e.g., [SR93]), which highlight the necessity to have a good test set in order to achieve a significant accuracy in the diagnostic process.

Indeed, the increase of the resolution power of a test set has been addressed in the past, in particular in the field of circuit diagnosis. In this scenario, the compaction of a generalized test set is to be performed by conserving the discrimination capability of the selected test vector [SA09]. Adaptive diagnosis policies have been investigate in the past, as in [HN84]. Also, *adaptive diagnostic test generation* has been investigated [AFT08], where the generation of new test patterns increases the diagnostic resolution of the test set. However, the size of the test set plays a key-role in a fault diagnosis methodology, because an optimal trade-off between the search for the *most significant* test (requiring a larger number of available tests) and the *complexity* of the search itself (requiring a compact test set) is needed. This is especially true for *incremental* approaches, as AF2D or as a different method proposed by Tang

et al. [TCXAS09]), where only a subset of tests is executed to identify the *faulty candidate* component.

In a typical (industrial) scenario, the test suite is identified by the test analyst group with the contribution of system designer groups; one of their goals is to identify, by applying a *best effort* policy, a reasonable trade-off between the *cardinality* and the *diagnostic efficiency* of a test set. In this perspective, they could benefit from a systematic analysis to improve the selected test set, through a methodology able to highlight deficiencies and shortcomings and to identify/evaluate/suggest tests to be added or removed from the set. In this contribution we focus on the evaluation of the resolution diagnostic power of a specific set of tests; it proposes i) a metric for estimating the relative resolution capability of tests in a test set, and ii) an algorithm for the construction of a minimal *extended* test set, capable of removing ambiguities between components pairs contained in the original test set. Applied to the AF2D methodology, it could be adopted in similar scenarios to improve test set diagnostic resolution capabilities.

This Chapter is organized as follows. Section 5.1 introduces the preliminaries of the diagnostic strategy, and the reference model being investigated. The main contribution is presented in Section 5.2, where the strategy to analyze parameters sensitivity is outlined. An approach for deriving the *golden* model, based on ATPG is introduced, and experimental results supporting the discussion are presented. In a second part, Section 5.3 presents the test suites *accuracy* problem: a definition of the metric for the evaluation of the resolution power of a test set is given in Section 5.4, together with a proposal of an incremental algorithm used to build a minimal extended test set increasing the diagnostic accuracy, and experimental results to validate the approach.

5.1 Golden model for CTM

5.1.1 Definitions

Let us consider a general system under analysis $\mathcal{S}$. An instance of $\mathcal{S}$ can be affected by a fault with probability $P_{\mathcal{S}}$. From a statistical perspective, we can state that this probability is the ratio between the number of instances of faulty $\mathcal{S}$ identified during diagnosis and the number of systems considered.

In system $\mathcal{S}$, a generic fault can occur in certain set of locations. We consider the fault occurring at each location as a permanent fault.

Definition 26. Let $\mathcal{F}_\mathcal{S}$ be the set $\{f_1, f_2, \ldots, f_{|\mathcal{F}_\mathcal{S}|}\}$ of size $|\mathcal{F}_\mathcal{S}|$. Each item f_i represents a location of the system $\mathcal{S}$ where a (permanent) fault can occur.

Definition 26 is very general, since it does not entail any specific fault model. Because of this, we can consider that all locations can be affected by a fault with equal probability:

Assumption 3. The probability distribution of faults occurring at each location $f_i \in \mathcal{F}_\mathcal{S}$ is the uniform distribution.

Usually, the resolution at which faults can be localized is much higher than the resolution at which the system can be described in terms of components (Section 2.2). Without lack of generality, we can consider that each component of system $\mathcal{S}$, regardless its implementation details, is associated univocally with a set of possible fault locations.

Definition 27. Let $\mathcal{FC}_\mathcal{S}$ be the set $\{fc_1, fc_2, \ldots, fc_{|\mathcal{FC}_\mathcal{S}|}\}$ of size $|\mathcal{FC}_\mathcal{S}|$. The size of this set corresponds to the number of components (n_c) contained in $\mathcal{S}$. Each item fc_x is a proper subset of size $|fc_x|$ containing the faults in $\mathcal{F}_\mathcal{S}$ producing a misbehavior in component $\mathbf{c}_x$: $fc_x \in \mathcal{F}_\mathcal{S}$.

From Definition 27 follows $\mathcal{F}_\mathcal{S} = \bigcup_x (fc_x)$. The diagnosis of system $\mathcal{S}$ can be performed using a certain number of tests, contained in test suite $\mathcal{T}$. Each test $\mathbf{t}_y \in \mathcal{T}$ provides a different *stimulus* to the system to detect a certain number of faults. Such detection results in a FAIL outcome of test $\mathbf{t}_y$.

Definition 28. We dub $\mathcal{FT}_\mathcal{S}$ the set $\{ft_1, ft_2, \ldots, ft_{|\mathcal{FT}_\mathcal{S}|}\}$ of size $|\mathcal{FT}_\mathcal{S}|$. The size of this set corresponds to the number of tests (n_t) available for the diagnosis of $\mathcal{S}$. Each item t_y is a proper subset of size $|ft_y|$ containing the faults in $\mathcal{F}_\mathcal{S}$ producing a FAIL outcome for test $\mathbf{t}_y$: $t_y \in \mathcal{F}_\mathcal{S}$.

We can use the above-presented definitions to specify the *golden model* of CTM used to apply the AF2D methodology to system $\mathcal{S}$. We need to derive the elements of the BBN model.

5.1.2 A priori probability

From the uniform distribution hypothesis (Assumption 3), it follows that the probability to find a fault in component $\mathbf{c}_x$ in a generic instance of system $\mathcal{S}$ is:

$$P^{ap}_{\mathcal{F}(\mathbf{c}_x)} = P_\mathcal{S} \cdot \frac{|f\mathbf{c}_x|}{|\mathcal{F}_\mathcal{S}|} \tag{5.1}$$

Equation 5.1 uses only the information on the partitioning of the system into components. Thus, $P^{ap}_{\mathcal{F}(\mathbf{c}_x)}$ represents the a-priori probability of component $\mathbf{c}_x$ to contain a fault and is the value that should be used in the BBN (of AF2D) to initialize the probability value at the corresponding node.

5.1.3 Conditional probability

Let us focus on a component $\mathbf{c}_x$. Let use assume that, in a specific instance of $\mathcal{S}$, fault $\mathbf{f}$ occurred and its location is part of $\mathbf{c}_x$, i.e. $\mathbf{f} \in fc_x$. Given the uniform distribution hypothesis (Assumption 3), $\mathbf{f}$ can be any $f_i \in fc_x$, with equal probability. However, being the exact location f_i unknown, its detection by a generic test $\mathbf{t}_y$ can be only associated with a probabilistic value $P^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)} = \mathbf{P}(\mathbf{t}_y|\mathbf{c}_x)$. This is a conditional probability because of the (temporary) hypothesis that $\mathbf{f}$ belongs to $\mathbf{c}_x$. We obtain:

$$\mathbf{P}^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)} = \mathbf{P}(\mathbf{t}_y|\mathbf{c}_x) = \frac{|ft_y \bigcap fc_x|}{|fc_x|} \tag{5.2}$$

In system $\mathcal{S}$, this value can be computed for all test-component pairs $(\mathbf{t}_y, \mathbf{c}_x)$. Such conditional probabilities represent the quantitative coverage coefficients to be used in the BBN in order to establish the a-posteriori components probabilities once the test outcomes become available.

5.2 Generation of *varied* diagnosis CTMs

Given a complex system, it is not possible to estimate a-priori and conditional probabilities with an accuracy similar to that of the golden model CTM (Figure 5.2). The lack of accuracy is due to the limited available knowledge of the set of potential faults $\mathcal{F}_\mathcal{S}$. This is true especially if the expertise of the test engineers team depends on an high-abstraction level in the description of the system. Also, the presence of a coarse label set introduced to simplify the matrix building process limits the correlation of CTM coefficients with numerical estimation of probabilities according to Equation 5.1 and 5.2.

Therefore, since we expect the CTM to be, in general, manually generated, we now want to estimate the impact of different values (with respect to the golden model) on the final diagnostic result. In particular, we propose three different transformations of the BBN golden coverage values, to obtain alternative models that can be used for diagnosis. Such transformations are conceived

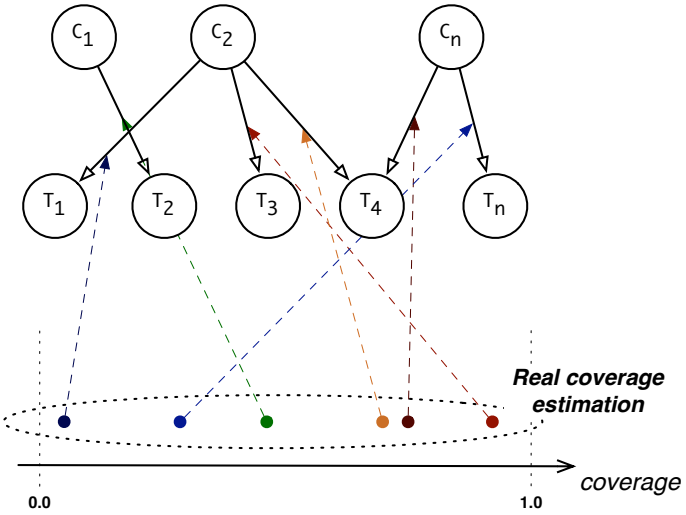


FIGURE 5.2: CTM (BBN) model containing real values for coverages.

- to describe an incremental modification of BBN to assess sensitivity to coefficients selection;
- to compare the results of the diagnosis based on a BBN which is different from the golden model.

By the results of this analysis, we assess the feasibility and limitations of diagnosis even with approximated CTM model estimated from test expertise.

We do not consider $\mathbf{P}^{ap}_{\mathcal{F}(\mathbf{c}_x)}$ values explicitly in transformations because of two reasons: on the one hand, they are associated with intrinsic fault rates of the component used to implement system $\mathcal{S}$; on the other hand, they are almost irrelevant for final probability assessment from BBN (mainly dependent on the conditional probabilities).

We first introduce a few concepts to define the transformations of CTM.

For the building of a modified CTM, we consider that a finite set of quantitative values is available.

Definition 29. Let us define $\mathcal{L}$ as the set $\{l_1, l_2, \ldots, l_{|\mathcal{L}|}\}$ of size $|\mathcal{L}|$. Each element $l_i \in \mathcal{L}$ represents a valid quantitative value to be associated in a CTM with the coverage of all test-component pairs.

Assumption 4. Coverage label –must be used only for test-component pairs $\langle \mathbf{c}_x, \mathbf{t}_y \rangle$ having $P^{cov}_{\mathcal{F}(\mathbf{c}_x, \mathbf{t}_y)} = 0$.

It is worth noting that this assumption implies that if a fault $\mathbf{f}$ occurs in $\mathbf{c}_x$ ($\mathbf{f} \in fc_x$) but is not detected by T_y ($\mathbf{f} \notin ft_y$), the only possible outcome of test $\mathbf{t}_y$ is PASS. Any other choice would make the BBN inconsistent with respect to the diagnosis of $\mathcal{S}$. Let us introduce now the CTM transformations.

5.2.1 BBN local transformation – LT

All non-zero coverage values from golden model BBN lie within the $(0, 1]$ range.

Let us consider a specific component $\mathbf{c}_x$. We can define a mapping $m_{\mathbf{c}_x}$ associating, for all tests $\mathbf{t}_y$, each pair $\langle \mathbf{c}_x, \mathbf{t}_y \rangle$ to a given value in the set $\mathcal{L}$. The formal statement is that this mapping is an application $m_{\mathbf{c}_x} : \mathcal{T} \to \mathcal{L}$.

According to this definition, the transformation of the CTM is formulated as a $k-$means clustering [HTFF05], with $k = |\mathcal{L}|$. The goals of the clustering are both the classification of each probability $\mathbf{P}^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)}$ of the golden model BBN in a class in levels' set $\mathcal{L}$, and the selection of optimal values for all items $(l_1, l_2, \ldots, l_{|\mathcal{L}|})$.

A generic Maximization-Expectation (ME) algorithm can be exploited to compute the optimal classification. We define a cost function for a quantitative evaluation of the quality of the classification of probabilities and the selection of values for items in $\mathcal{L}$. The solution of the ME is the mapping $\hat{m_{\mathbf{c}_x}}$ and set $\hat{\mathcal{L}}$, that minimizes for each component $\mathbf{c}_x$ function J_x:

$$\underset{\hat{m_{\mathbf{c}_x}},\hat{\mathcal{L}}}{\operatorname{argmin}} \quad J_x = \sum_{\mathbf{t}_y \in \mathcal{T}} (\mathbf{P}^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)} - l_{m_{\mathbf{c}_x}(\mathbf{t}_y)})^2 \tag{5.3}$$

If we minimize independently all J_x functions (i.e., we independently modify each row of the CTM), the distortion introduced on the conditional probabilities is minimum.

5.2.2 BBN global transformation – GT

Let us relax the previous transformation assumption and define a unique mapping $m_{\mathcal{S}}$. Such mapping is an application $m_{\mathcal{S}} : (\mathcal{C} \times \mathcal{T}) \to \mathcal{L}$. Because of this, we need to combine all J_x functions, computed for each component $\mathbf{c}_x$, in a unique cost function $J_{\mathcal{S}}$:

$$\underset{\hat{m_{\mathcal{S}}},\hat{\mathcal{L}}}{\operatorname{argmin}} \quad J_{\mathcal{S}} = \sum_{\mathcal{T},\mathcal{C}} (P^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)} - l_{m_{\mathcal{S}}(\mathbf{c}_x,\mathbf{t}_y)})^2 \tag{5.4}$$

In this case, we accept larger distortion to be introduced on the conditional probabilities. The possibility of mis-diagnosis is higher in this

case, because the difference between the real conditional probabilities $\mathbf{P}^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)}$ and the quantitative coverage coefficient l_i is in general higher. The global $J_{\mathcal{S}}$ cost function, while optimizing the coefficient classification in $\mathcal{L}$, produces less precise class values in solution $\hat{\mathcal{L}}$

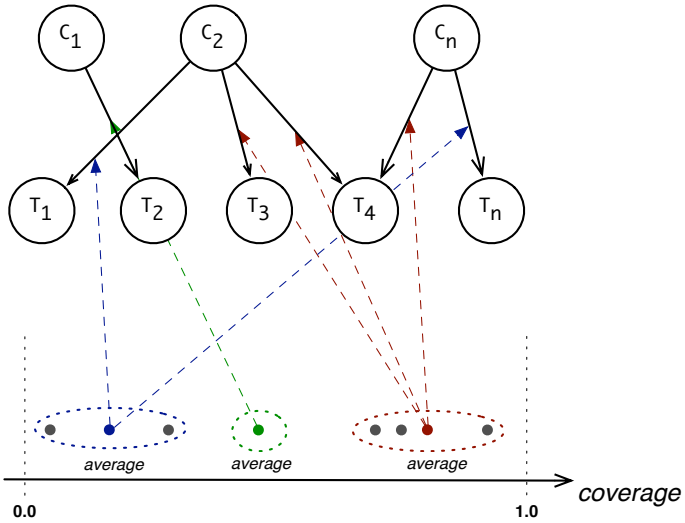


FIGURE 5.3: CTM (BBN) model after global transformation.

5.2.3 BBN global fixed transformation – FT

Let us consider again a global mapping $m_{\mathcal{S}}$. In this case, we remove the selection of values in $\mathcal{L}$, considering them as fixed parameters. For instance, we can consider that $\mathcal{L} = \bar{\mathcal{L}} = (\bar{l}_1, \bar{l}_2, \ldots, \bar{l}_{|\mathcal{L}|})$ (Figure 5.4). This scenario is exactly equivalent to the definition of the methodology adopted in [ABSF10a]. The $J_{\mathcal{S}}$ cost function, in this case, is minimized only through the mapping $m_{\mathcal{S}}$:

$$\underset{\hat{m_{\mathcal{S}}}}{\text{argmin}} \quad J_{\mathcal{S}} = \sum_{\mathcal{T},\mathcal{C}} (P^{cov}_{\mathcal{F}(\mathbf{c}_x,\mathbf{t}_y)} - \bar{l}_{m_{\mathcal{S}}(\mathbf{c}_x,\mathbf{t}_y)})^2 \tag{5.5}$$

We aim at using these transformation of the CTM to evaluate the different results produced by the diagnostic procedure, compared against the the results achieved when using the golden CTM model. The AF2D methodology will be applied to the four different models and the difference in the results will be used to estimate the sensitivity of the diagnosis to the casual model.

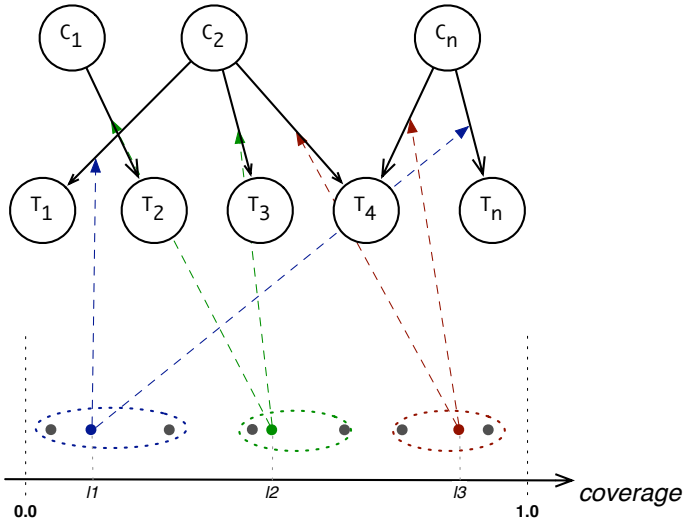


Figure 5.4: CTM (BBN) model after global fixed transformation.

Example 18. Let us consider the matrix in Figure 5.5 as a golden model for a system diagnosis. For the sake of simplicity, the label set $\mathcal{L}$ is composed of four symbols $\{l_0, l_1, l_2, l_3\}$. According to Assumption 4, all 0's values are kept unmodified in all trasformations, and they are mapped to $_0$ (corresponding to –). All other values are mapped to l_1, l_2 or l_3.

Figure 5.6 and 5.7 show the *local* (LT) and *global* (GT) transformations from the CTM in Figure 5.5, respectively. The 3-values scale holds for LT within each row (e.g., for $\mathbf{c}_3$ $\mathcal{L} = \{0, 0.351, 0.736, 1.000\}$), and for GT for all rows with mapping $\mathcal{L} = \{0, 0.273, 0.601, 0.974\}$. Finally, the *fixed* transformed CTM (FT) is presented in Figure 5.8, using the constant mapping $\mathcal{L} = \{0, 0.1, 0.5, 0.9\}$.

◇

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$	$\mathbf{t}_6$	...
$\mathbf{c}_1$	0.957	0.348	0.435	0.348	0.000	0.349	...
$\mathbf{c}_2$	0.000	1.000	0.000	0.000	0.000	0.370	...
$\mathbf{c}_3$	0.000	0.334	1.000	0.333	0.167	0.661	...
...	...	...	...	...	...	...	...

Figure 5.5: Partial golden model CTM for Example 18.

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$	$\mathbf{t}_6$	...
$\mathbf{c}_1$	0.961	0.393	0.393	0.393	0.000	0.393	...
$\mathbf{c}_2$	0.000	0.933	0.000	0.000	0.000	0.382	...
$\mathbf{c}_3$	0.000	0.351	1.000	0.351	0.351	0.736	...
...	...	...	...	...	...	...	...

FIGURE 5.6: Partial CTM after local transformation (LT) of CTM in Figure 5.5.

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$	$\mathbf{t}_6$	...
$\mathbf{c}_1$	0.974	0.273	0.273	0.273	0.000	0.273	...
$\mathbf{c}_2$	0.000	0.974	0.000	0.000	0.000	0.273	...
$\mathbf{c}_3$	0.000	0.273	0.974	0.273	0.273	0.601	...
...	...	...	...	...	...	...	...

FIGURE 5.7: Partial CTM after global transformation (GT) of CTM in Figure 5.5.

	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$	$\mathbf{t}_6$	...
$\mathbf{c}_1$	0.900	0.100	0.500	0.100	0.000	0.100	...
$\mathbf{c}_2$	0.000	0.900	0.000	0.000	0.000	0.100	...
$\mathbf{c}_3$	0.000	0.100	0.900	0.100	0.100	0.500	...
...	...	...	...	...	...	...	...

FIGURE 5.8: Partial CTM after fixed transformation (FT) of CTM in Figure 5.5.

5.2.4 Sensitivity analysis and results

We will compare the diagnosis results of the golden model BBN, with those achieved using the modified versions of the same BBN, obtained by applying the introduced systematic transformations. By using the vocabulary of pattern recognition, we can define the concepts of *true and false positives* (components considered to be faulty by diagnosis, with and without actual failures, respectively) and *true and false negatives* (components considered to be fault-free by diagnosis, without and with actual failures, respectively).

Diagnostic classification performance is evaluated according to the well-known metrics of *accuracy* AC, *precision* PR and *recall* RC (

[HTFF05]), computed as:

$$AC = \frac{t_P + t_N}{t_P + f_P + t_N + f_N} \tag{5.6}$$

$$PR = \frac{t_P}{t_P + f_P} \qquad RC = \frac{t_P}{t_P + f_N} \tag{5.7}$$

The metrics combine the number of true and false positives (t_P, f_P) and true and false negatives (t_N, f_N).

A golden model consists of any model extracted from a circuit representation allowing for a quantitative and *accurate* description of the relation between components, tests and test outcomes. In particular, such quantitative value represents the probability of a test to sensitize a fault in a component and make the effects observable at the output. To derive such a golden model, we exploited the use of an Automatic Test Pattern Generator (ATPG) [Ha94], in an unconventional way, to extract the information. Here we focus on combinational logic circuits and refer to the single permanent stuck-at fault model. The choice is motivated by the need to derive a golden model which to apply the transformation, and the selected scenario allows for the adoption of an immediately developed strategy; however, the same approach can be extended to sequential and complex circuits, and a broader fault model can be adopted.

We introduce the use of an ATPG to support the derivation of the CTM golden model for diagnosis. The role of ATPG in our scenario is not fault detection, but rather a tool to analyze circuit properties and, in particular, to support the exact evaluation of Equation 5.2 (used for AF2D diagnosis).

From this perspective, the focus on combinational circuits cannot be considered a limiting or excessively conservative assumption, as it would be true for a fault detection application, or if we aimed at implementing the AF2D methodology to diagnose circuits at gate level. In such a case, other more efficient approaches have been already proposed in literature, as [PR00] or [SA09].

A fault can occur on each gate input/output. Therefore, according to Definition 26, $\mathcal{F}_\mathcal{S}$ contains all possible gate inputs/outputs of the circuit, that can be affected with a stuck-at fault. In other words, $\mathcal{F}_\mathcal{S} = \{sa_1^0, sa_1^1, \ldots, sa_{n_p}^0, sa_{n_p}^1\}$, where n_p is the number of ports of the circuit. However, we do not consider each logic gate as a stand-alone component of the system $\mathcal{S}$ for two reasons. First, there would be very little interest in knowing exactly what gate is actually broken. Also, it would be impossible to implement such a BBN engine, because of the exponential complexity of computations involved. Furthermore, degen-

erating $\mathcal{FC}_{\mathcal{S}}$ to singleton sets, Equation 5.2 would become meaningless, since probabilities would be replaced with *binary relations* (0%, 100%).

Therefore, we need to define the concept of *component* on a network of logic gates, such that the number of components is meaningful and manageable.

5.2.5 Components

We consider a *cluster of gates* of the circuit to be a *component*, and the entire system to be the set of defined clusters. Consequently, the presence of a faulty gate within the cluster corresponds to the presence of a fault on the components related to the cluster itself.

The partition of gates in clusters is an arbitrary choice, and random allocation is a valid one. However, to make the components partition similar to the one we usually witness in real systems (for instance, in board level diagnosis), it is natural to consider only *convex gates clusters*; in other words, we consider a cluster to be a *valid* if no path between each pair of gates of the cluster contains a gate not belonging to the cluster itself (see Figure 5.9 for an example).

It is worth noting that, whatever the selection of a clustering schema, the partition of the circuit defines also the partition of faults $\mathcal{FC}_{\mathcal{S}}$ set (Definition 27). The set of fault locations ($\mathcal{F}_{\mathcal{S}}$) includes all gates' inputs and outputs, and no fault collapsing is considered.

5.2.6 Tests

The ATPG extracts a non-compressed test set of test vectors $TV = \{tv_1, tv_2, \ldots, tv_{|TV|}\}$, since we interested in the diagnostic capabilities of the TV set, and not in the smallest one. We do not compress the test vector set obtained from the ATPG, thus all possible test vectors are kept. Here we need not identify the smallest set of vectors detecting all possible faults, rather we are interested in the diagnostic capabilities of the test vector set. Therefore, the test set includes all 2^{n_i} configurations, being n_i the number of input signals. Therefore, TV includes all 2^{n_i} configurations, being n_i the number of input signals. By exploiting the ATPG fault simulation feature, it is possible to determine the subset of faults each test vector detects, information used to derive $\mathcal{FT}_{\mathcal{S}}$ (Definition 28).

From an AF2D perspective, we consider a test $\mathbf{t}_y$ as the *application of a group of test vectors* ($\mathbf{t}_y = \{tv_{z1}, tv_{z2}, \ldots, tv_{zy}\}$). Since T_y should stimulate the system $\mathcal{S}$ producing a binary outcome, it is said to have a FAIL outome if there is at least a difference on the outputs for a tv with

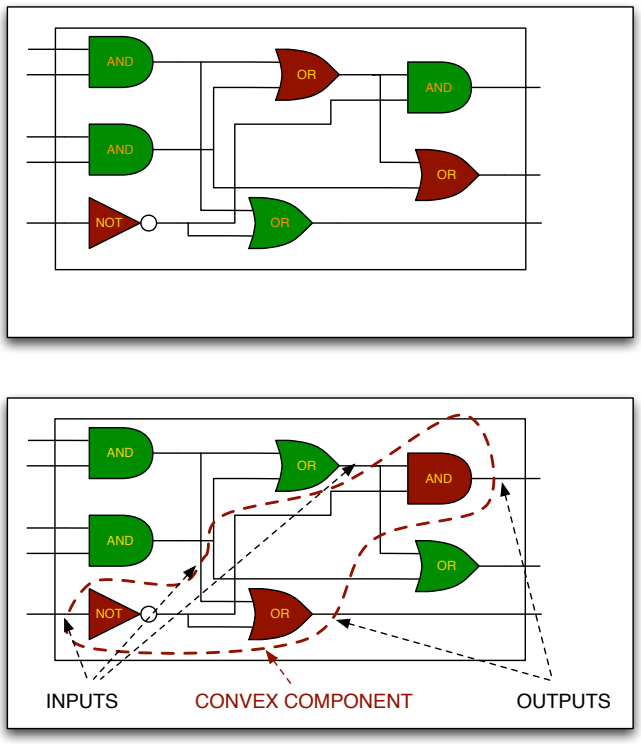


Figure 5.9: Random and convex partitions of gates.

respect to the expected fault-free values (PASS otherwise). The choice to form *groups* of test vectors is adopted because $\mathcal{FT}_{\mathcal{S}}$ sets defined for single test vectors, even if not singleton sets, do not cover a large number of faults per component. Providing CTM obtained from such probabilities a reduced discrimination among components, they do not reflect CTM we witness in AF2D analysis.

This notion of test is straightforward, but it needs a reformulation: considering a one-to-one mapping from test vectors to diagnosis tests ($\forall y, \mathbf{t}_y = \{tv_y\}$), we create a implicit correspondency between the faults detected by tv_y and the faults detected by $\mathbf{t}_y$ ($ft_y \in \mathcal{FT}_{\mathcal{S}}$), i.e., test $\mathbf{t}_y$ would detect all stuck-at faults associated with test vector tv_y. Because of the component partition, faults occurring in different components $\mathbf{c}_1, \ldots, \mathbf{c}_{n_c}$ could be detected by $\mathbf{t}_y$; however, the sets $fc_1, \ldots, fc_{n_c} \in \mathcal{FC}_{\mathcal{S}}$ and the set ft_y are uncorrelated (due to components boundaries) and, consequently, the portion of faults detected by $\mathbf{t}_y$ in each component $\mathbf{c}_x$ is lower with respect to the total number of

faults contained in it.

Thus, we propose an alternative definition of tests $\mathbf{t}_y \in \mathbf{T}$ to increase such portion.

5.2.7 AND-grouped test vectors

Let us define a group of test vectors $\{tv_{z1},tv_{z2},\ldots,tv_{zn}\}$ forming a proper subset of the set of all test vectors TV. If we apply each test vector to the circuit and we collect the output values, we obtain a list of n binary outcomes o_i: $\{o_{z1},o_{z2},\ldots,o_{zn}\}$. The outcome of this group of tests is FAIL if there is at least a FAIL among the test vector outcomes $\{o_{z1},o_{z2},\ldots,o_{zn}\}$.

Definition 30. Let each $\mathbf{t}_y \in \mathbf{T}$ be a group of test vectors $\{tv_{z1},tv_{z2},\ldots,tv_{zn}\}$ obtained from the ATPG. We define the outcome of $\mathbf{t}_y$ as the logic AND of the outcomes of each $\{tv_{z1},tv_{z2},\ldots,tv_{zn}\}$ applied independently, where the outcomes PASS and FAIL are respectively the logic values 1 and 0.

The grouping of test vectors introduces a fuzziness with respect to the real fault causing the test to fail. The portion of covered faults per component each test offers can be increased with the size of number of tv_zs, and coverage depends on which test vectors are selected. Random grouping would potentially result in useless tests, because the lack of coverage per component is caused by the uncorrelation of components boundaries. Thus, a systematic test vector grouping strategy is proposed, aiming at maximizing a component $\mathbf{c}_x$ coverage by $\mathbf{t}_y$, while minimizing the number of test vectors contained in T_y itself.

As in Section 3.2, we adopt Integer Linear Programming (ILP) as a mathematical optimization technique for linear objective functions, subject to linear (both equality and inequality) constraints.

Let us consider a test $\mathbf{t}_y$. For each vector $tv_j \in TV$, we introduce a binary variable x_j assuming value 1 only when $tv_j \in \mathbf{t}_y$. Let us consider also component $\mathbf{c}_x$: from fault simulation, we can determine a binary relation M between each stuck-at fault $f_i \in fc_x$ and each test vector tv_j. In a matrix form, $m(f_i, tv_j) = 1$ if fault f_i is detected by test vector tv_j. Focusing on a certain subset of faults (constraint faults, $\mathbf{fc*}_x \subseteq fc_x$) belonging to $\mathbf{c}_x$. The size of this set of faults can range from the empty set (0% coverage of $\mathbf{c}_x$) to the fc_x set itself (100% coverage of $\mathbf{c}_x$).

For each test vector tv_j, let us consider a cost variable w_j. By imposing w_j $(\forall j)$, we can use w_j to define a naive cost function of the group of test vectors (number of tvs). Alternatively, if we aim at maximizing the focus of $\mathbf{t}_y$ on component $\mathbf{c}_x$, we can define the cost term as the number of faults covered by $\mathbf{t}_y$ not included in fc_x, i.e., $w(\mathbf{c}_x)_j = |\mathcal{F}_{tv_j}/fc_x|$.

		tv_1	tv_2	tv_3	tv_4	tv_5	tv_6	
(cost)		(2)	(1)	(2)	(2)	(1)	(0)	
$\mathbf{c}_1$	f_1	1	1	1	1	1	0	$\mathbf{fc*}_1$
	f_2	1	0	0	1	1	0	$\mathbf{fc*}_2$
	f_3	0	1	1	0	1	1	
	f_4	1	1	1	0	0	1	$\mathbf{fc*}_3$
$\mathbf{c}_2$	f_5	1	0	0	0	1	0	
	f_6	0	1	1	1	0	0	
$\mathbf{c}_3$	f_7	1	0	1	1	0	0	

Figure 5.10: Faults and test vector coverage for ILP optimization (Example 19).

The minimization problem can be formulated as in Equation 5.8:

$$\text{minimize} \quad \sum_j w(\mathbf{c}_x)_j \cdot x_j \tag{5.8}$$

$$\text{subject to} \quad \forall \mathbf{fc*}_i \quad \sum_j m(f_i, tv_j) \cdot x_j \geq 1 \tag{5.9}$$

$$\forall j \quad x_j \quad \text{binary} \tag{5.10}$$

Several tests can be generated by executing the optimization, by changing the target component $\mathbf{c}_x$ and the number of constraints faults defined for $\mathbf{c}_x$.

Example 19. Let us consider the coverage described in Figure 5.10. The group of test vectors to form T_1 is defined by constraining the coverage of faults (f_1, f_2, f_4): the request is a coverage of $\mathbf{c}_1$ higher or equal to 75%. The cost function w represents the number of faults not belonging to $\mathbf{c}_1$ covered by each test vector.

The minimization of Equation 5.8 leads to $\mathbf{t}_1 = \{tv_5, tv_6\}$, with a 100% coverage of $\mathbf{c}_1$.

◇

It is worth noting that the simple fault model proposed (stuck-at faults) and the focus on combinatorial circuits only should not be considered as limiting or excessively conservative assumptions. In general, this is true in a fault detection application or in an AF2D implementation for gate level fault diagnosis of digital systems. We have already pointed out that, for this purpose, efficient approaches have been already proposed in literature, as [PR00] or [SA09]. The same can be said to

Table 5.1: Summary of circuits properties.

Circuit group	# circuits	# faults (avg)	**#Comp.**	**#Tests**
Group1	10	245	8	32
Group2	10	2602	40	110
Group3	5	980	10	44
Group4	5	3648	35	125

ATPG use, which is proposed as a support tool for quantitative coverage analysis and fault simulation [HKO+09].

The experimental analysis is executed on a group of combinatorial circuits with different characteristics. For each circuit, a partition has been defined, determining the components as cluster of gates and exploiting the ATPG to extract the non-compressed set of test vectors, used to define the diagnostic tests. Table 5.1 reports the characteristics of the classes of used circuits, 30 in total.

Given the definitions of components and tests with respect to the set of all faults identified through the ATPG, we computed the coefficients for golden model CTM for each circuit under analysis, using Equation 5.2. For each circuit, the derived golden CTM model has been manipulated according to the transformations defined in Section 2.2, generating three additional CTM matrices for the circuit. Then, each matrix has been used as input in an independent session of the AF2D methodology. Syndromes have been computed by injecting random single faults in the circuit, and by applying the diagnostic tests.

To avoid bias introduced by *model overfitting*, a *cross-validation* strategy has been adopted. The syndromes for each circuit have been partitioned in about 10 subsets, and the metrics have been evaluated on each subset independently. Finally, metrics results have been averaged from all subsets.

Let us recall that the golden model of the CTM and its transformed versions contains directly quantitative values. Since the manually created CTM contains only three possible values (L, M, and H), the CTM computed with the proposed transformations use a three-values scale as well. They correspond to experiments labeled *LT* and *GT*, representing the application of the *local* and *global* transformations, respectively. We also analyzed the global transformation using a 5-values scale, mimicking a CTM with possible values Low, Low/Medium, Medium, Medium/High and High, having intermediate values. This experiment is labeled *NT*.

Table 5.2: Accuracy: average value for each class of experiments.

Circuit group	Golden	LT	GT	NT	FT
Group1	0.997	0.995	0.995	0.995	0.993
Group2	0.971	0.937	0.946	0.958	0.923
Group3	0.996	0.962	0.969	0.980	0.956
Group4	0.931	0.901	0.913	0.926	0.895

Finally, experiment *FT* corresponds to the transformed CTM using the quantitative scale 0.1, 0.5 and 0.9 (as the one presented in Section 2.4).

The average *accuracy* of diagnosis is presented in Table 5.2. For all families of circuits under analysis, we obtain a minimal number of erroneous diagnosis (*faulty* components indicated as fault-free or vice-versa). This indicator supports the claim about the *correctness of modeling* the system with coarser labeled CTM. Furthermore, since transformations modify the coefficient value, the occasional insertion of a *wrong* coverage label, i.e., a coverage which is different from the one in golden model CTM, does not introduce a significant error in diagnosis.

From results, we observe that the number erroneous diagnosis increases with the size of the circuits (Groups 2 and 4), from a 7% for the golden model to an 11% for the *fixed* transformed CTM. This behavior can be explained by the greater complexity of obtaining a CTM with high diagnostic resolution in such systems.

The metrics *precision* and *recall* for the diagnosis quality evaluation are reported in Table 5.3 and Table 5.4, respectively. The classification of diagnosis results in *true and false positives* (and *negatives*) is done by analyzing the probability of each component to be faulty when the fault is actually injected in that component, using the following thresholds. A diagnosis is considered as correct (*positive*) when the probability of a component to be faulty is lower than 20% and the fault has been injected elsewhere; on the other hand, *false negatives* are the diagnosis for an actual faulty component when its probability to be faulty is lower than 95%. These assumptions about diagnosis correctness are more conservative with respect to the AF2D criteria in [ABSF10a], but they are here used to enforce the claim about coefficient sensitivity of the model.

We can confirm from results that the quality of diagnosis is not significantly affected by the adoption of coarse coefficients instead of the real coverage in the specification of the CTM. The AF2D methodology proves to be not sensible to the specific adopted coverage label

Table 5.3: Precision: average value for each class of experiments.

Circuit group	Golden	LT	GT	NT	FT
Group1	0.972	0.979	0.940	0.986	0.940
Group2	0.992	0.996	0.996	0.992	0.984
Group3	0.989	0.895	0.900	0.953	0.874
Group4	0.962	0.916	0.913	0.963	0.905

Table 5.4: Recall: average value for each class of experiments.

Circuit group	Golden	LT	GT	NT	FT
Group1	0.993	0.981	0.983	0.990	0.979
Group2	0.984	0.964	0.960	0.968	0.957
Group3	0.975	0.909	0.911	0.937	0.889
Group4	0.968	0.901	0.922	0.937	0.881

system. From this, we support the claim on confidence about the robustness of the model with respect to diagnosis quality in presence of a non-systematic mislabeling of some component-test coverage pairs.

As for the differences in the results, the analysis shows that the dimension of the circuit has an impact on the diagnosis. In particular, the rate of mis-located faults is higher in circuits of a smaller size (lower *precision*). This is motivated by the fact that, in small circuits, a test often covers faults belonging to several different components and the corresponding CTM has a low localization (diagnostic) power. On the other hand, when the size of the circuit increases, there are more faults that are difficult to cover, and therefore the number of *false negatives* increases, leading to a lower value for the *recall* indicator.

5.3 Evaluating Diagnostic Accuracy

This section introduces the main concepts at the basis of the proposed strategy to evaluate test effectiveness and to improve the test suite for an increased accuracy in fault diagnosis.

The BBN model, and in particular the *a-posteriori* test probability values computed for *singleton fault distribution* (Definition 17) offers an insight of the *capability* of the test-suite to *discriminate* different faulty components while analyzing syndromes. We focus on Complete

Syndrome only, disregarding Partial Syndromes (Definitions 15 and 16), because we are interested in evaluating the quality of the final diagnosis produced by the BBN when the outcomes set of the entire test-suite is available.

Let us consider a BBN, describing a system $\mathcal{S}$ containing n_c components and n_t tests. Furthermore, let us consider a scenario where we know that the location of a fault $\mathbf{f}$ in the $\mathcal{S}$ is component $\mathbf{c}_x$, and let $\mathbf{f}$ be the only fault observed. It is was noting that i) from BBN independence properties (Section 2.2), and ii) from the impossibility for some tests $\mathbf{t}_{z1}, \mathbf{t}_{z2}, \ldots$ to FAIL if they do not involve the faulty component (Assumption 2),it is possible to prune the set $\{\mathsf{PASS}, \mathsf{FAIL}\}^{n_t}$ of all possible syndromes, excluding those syndromes containing a FAIL for those latter tests.

Each syndrome is associated with a *conditional* probability computed composing the *conditional probabilities* defined for all component-test $\langle \mathbf{c}_x, \mathbf{t}_z \rangle$ pairs. All *pruned* syndromes are incompatible with the hypothesis of the fault $\mathbf{f}$ to be contained in $\mathbf{c}_x$, and they have a *null* probability to occur. All remaining syndromes, on the other hand, can be ranked according to their *non-null* probability from *more* to less *likely*.

Example 20. Figure 5.11 depicts a sample BBN. The observed faulty component is $\mathbf{c}_2$, Because of BBN independence, the probability of a generic syndrome $[\mathbf{t}_1 = \mathbf{o}_1, \mathbf{t}_2 = \mathbf{o}_2, \ldots, \mathbf{t}_5 = \mathbf{o}_5]$, where $\mathbf{o}_z$ represents the outcome of obtained after the execution of test $\mathbf{t}_z$, can be computed as

$$\mathbf{P}(\mathbf{t}_1 = \mathbf{o}_1, \mathbf{t}_2 = \mathbf{o}_2, \ldots, \mathbf{t}_5 = \mathbf{o}_5 | \mathbf{c}_2 = \mathbf{F}) = \prod \mathbf{P}(\mathbf{t}_x = \mathbf{o}_1 | \mathbf{c}_2 = \mathbf{F})$$

High, *Medium*, *Low* coverage labels are respectively to quantitative values 0.9, 0.5, 0.1. Therefore, next to each syndrome the *conditional* probability is reported, given that the *observed faulty component* is $\mathbf{c}_2$. For instance, the probability of the first syndrome where only the first test $\mathbf{o}_1$ =FAIL, while $\mathbf{o}_2$ =...= $\mathbf{o}_5$ =PASS, is equal to

$$\begin{aligned} \mathbf{P}(\mathbf{o}_1 = \mathsf{FAIL}, \mathbf{o}_2, \ldots, \mathbf{o}_5 = \mathsf{PASS} | \mathbf{c}_2) &= \mathbf{P}(\mathbf{o}_1 = \mathsf{FAIL}, \mathbf{o}_3 = \mathbf{o}_4 = \mathsf{PASS} | \mathbf{c}_2) \\ &= 0.9 \cdot 0.5 \cdot 0.9 = 0.405 \end{aligned}$$

◇

The *a-posteriori* probability of each syndrome $\mathbf{s}_y$ is computed multiplying its *conditional* probability $\mathbf{P}(\mathbf{s}_y | \mathbf{c}_x)$ with the *a-priori* probability of the *faulty* component $\mathbf{c}_x$ generating it. Given that the same syndrome $\mathbf{s}_y$ can be generated by several components of the BBN, it is possible to obtain from *marginalization*:

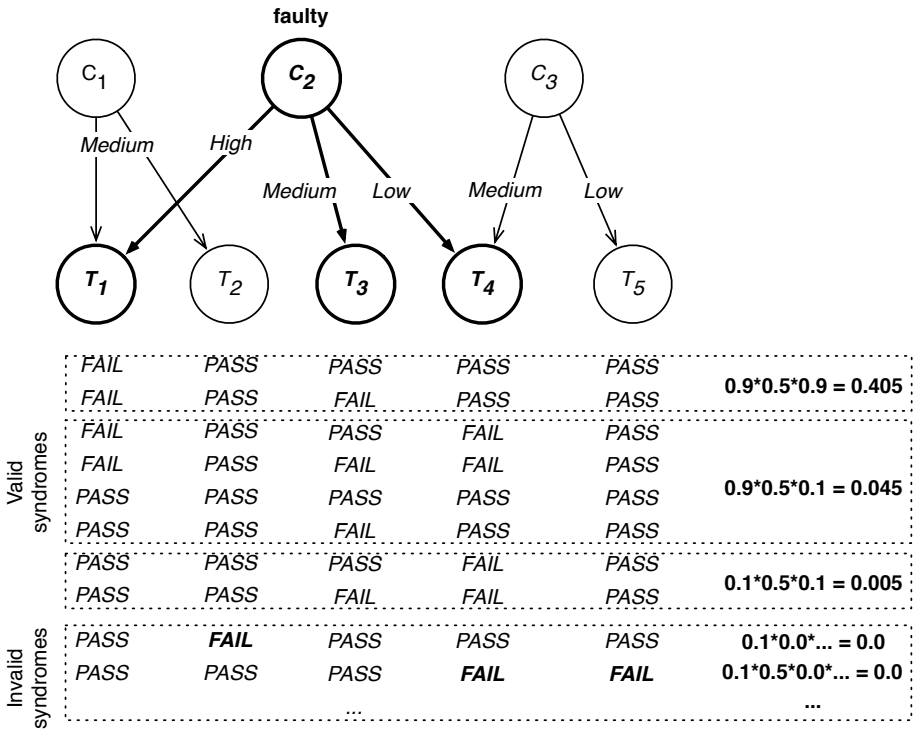


FIGURE 5.11: Syndromes ranking (decreasing *conditional* probabilities).

$$\mathbf{P}(\mathbf{s}_y) = \sum_{\mathbf{c}_x \in \mathcal{C}} \mathbf{P}(\mathbf{s}_y|\mathbf{c}_x) \cdot \mathbf{P}(\mathbf{c}_x) \tag{5.11}$$

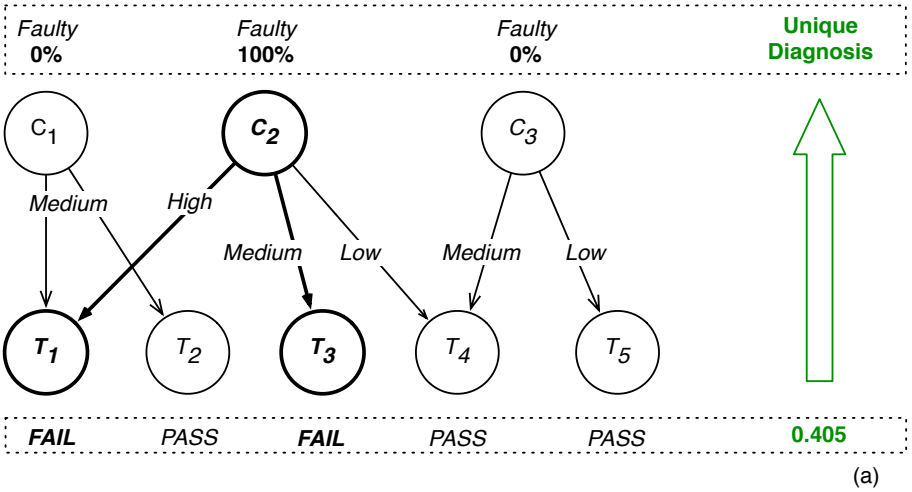
We can observe in Example 20 that only two syndromes alone, **FPPPP** and **FPFPP**, represent alone the 81% of the possible valid outcomes configurations. However, it is necessary to consider 6 of 8 syndromes to consider the 99% of valid configurations. This information is important for evaluating the accuracy of test-suite $\mathcal{T}$ described in a BBN.

In fact, $\mathcal{T}$ if as much efficient in *discriminating* faulty components as much the BBN diagnosis conclusion for a syndrome $\mathbf{s}_x$, produced by a faulty component $\mathbf{c}_x$, excludes all other components from the FCC set.

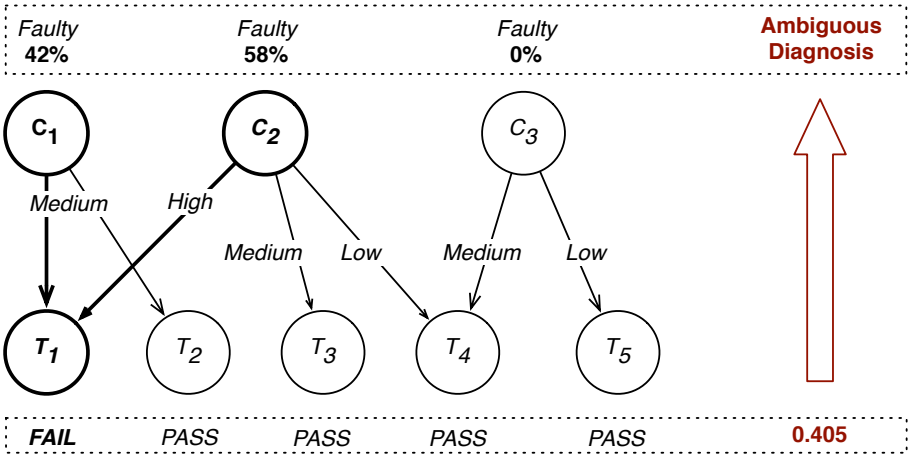
Example 21. Figure 5.12 depicts the diagnosis conclusion of the BBN for the two most likely syndromes generated by component $\mathbf{c}_2$ in Example 20. In (a), diagnosis for **FPFPP** is unique, and it is correctly $\mathbf{c}_2$. In (b), diagnosis for **FPPPP** is labeled as ambiguous, since the FCC set include, with $\mathbf{c}_2$, also $\mathbf{c}_1$. In fact, the same syndrome is one of the four pos-

sible syndromes, namely $\{\mathbf{FFPPP}, \mathbf{FPPPP}, \mathbf{PFPPP}, \mathbf{PPPPP}\}$, generated by component $\mathbf{c}_1$.

◇



(a)



(b)

Figure 5.12: Unique (a) and Ambiguous (b) diagnosis for syndromes of $\mathbf{c}_2$.

A possible framework for the solution of the *diagnostic resolution* evaluation of the test-suite $\mathcal{T}$ associated with a BBN can be formulated as the following. Let us consider a syndrome $\mathbf{s}_y$, generated by a

component $\mathbf{c}_x$. Equation 5.11 can be used to compute the joint probability $\mathbf{P}(\mathbf{s}_y, \mathbf{c}_x)$ of observing the syndrome-faulty component pair $\langle \mathbf{s}_y, \mathbf{c}_x$. BBN inference (Section 2.2) computes the probability $\mathbf{P}(\mathbf{dc}_x|\mathbf{s}_y, \mathbf{c}_x)$ of obtaining a diagnostic conclusion $\mathbf{dc}_x$ ($\mathbf{c}_x \in$ FCC).

$$\mathbf{P}(\mathbf{dc}_x, \mathbf{s}_y, \mathbf{c}_x) = \sum_{\mathbf{s}_y, \mathbf{c}_x} \mathbf{P}(\mathbf{dc}_x|\mathbf{s}_y, \mathbf{c}_x) \cdot \mathbf{P}(\mathbf{s}_y, \mathbf{c}_x) \tag{5.12}$$

Therefore, Equation 5.12 is a JPD evaluating the *diagnostic resolution* of the test suite $\mathcal{T}$, which is as higher as the probability of pairs $\mathbf{dc}_x \neq \mathbf{c}_x$ is low. Unfortunately, this JPD cannot be exploited directly because of two problems:

- the direct valuation if computationally *unaffordable*, because the number of possible syndrome-component $\langle \mathbf{s}_y, \mathbf{c}_x \rangle$ pairs grows exponentially with the size of the network;
- any modification at the BBN structure would require a complete re-evaluation of the JPD, and it can be exploited to compare different test suites $\mathcal{T}_{\mathcal{S},1}, \mathcal{T}_{\mathcal{S},2}, \ldots$ for the diagnosis of system $\mathcal{S}$.

5.3.1 Distance metric

Because of the drawbacks exposed, we need to exploit differently the *a-posteriori* probability information of syndromes. We consider again the *geometrical interpretation* of the BBN model adopted in Section 4.2.

Recalling the notion of distance in $\mathcal{L}^p$ spaces:

Definition 31. Let $\mathbf{v}$ be a $[v_1 v_2 \ldots v_{n_t}]$ vector of n_t elements, where $v_j \in \mathcal{R}$. By recalling the notion of *distance in* $\mathbf{L}^p$ *spaces*, we can define the **distance** $d(\mathbf{v}^x, \mathbf{v}^y) \in \mathcal{R}$ between the vector pair $(\mathbf{v}^x, \mathbf{v}^y)$ as:

$$d(\mathbf{v}^x, \mathbf{v}^y) = (\sum_{j=1}^{n_t} (\|v_j^x - v_j^y\|)^p)^{1/p} \tag{5.13}$$

In our scenario we are interested in computing the distance between a syndrome and an Attraction Vector, or between each pair of Attraction Vectors of the components. This is done to evaluate the efficiency of the available/adopted test set in discriminating the faulty candidates, with respect to the specified system model.

Since every element of the vector is defined in $[0, 1]$, by increasing p the importance of the terms that are very different from each other increases. Furthermore, because the values belong to the finite set $\mathbb{C}_{\mathbb{V}} = \{0.1, 0.5, 0.9, 1.0\}$, the significant pairs contributing to the distance for high values of p are $\{0.1, 1.0\}$ and $\{0.1, 0.9\}$, corresponding to the coverage label pairs $\{H,–\}$ and $\{H, L\}$, respectively.

5.3.2 Overlapping and non-overlapping distances

Even if, from a theoretical point of view, Equation 5.13 might seem the most appropriate one because of its generality, in the specific case of the CTM analysis this is not true because of the requirement expressed in Corollary 1. Therefore, we can proceed with the analysis of the CTM model.

Let $\mathbf{v^x}$ and $\mathbf{v^y}$ be two AVs, obtained from the CTM for components $\mathbf{c}_x$ and $\mathbf{c}_y$, respectively. It is possible to evaluate the distance $d(\mathbf{v^x}, \mathbf{v^y})$ between the vectors by Equation 5.14 (n_t is the number of tests).

$$d(\mathbf{v^x}, \mathbf{v^y}) = (\sum_{\mathbf{t}_z \in \mathcal{T}} (\|\text{cov}(\mathbf{c}_x, \mathbf{t}_z) - \text{cov}(\mathbf{c}_y, \mathbf{t}_z)\|)^p)^{1/p} \tag{5.14}$$

However, it is interesting to identify two contributions to such distance, to highlight the situations where only one of the two component is covered by a test, which we call **non-overlapping** distance terms. Do note that when both components are not covered by a test, the term does not offer any contribution to the overall distance. Consequently, we split the distance value into two contributions:

- $d_{nov}(\mathbf{v_x}, \mathbf{v_y})$ for non-overlapping terms $(\text{cov}(\mathbf{c}_x, \mathbf{t}_z), \text{cov}(\mathbf{c}_y, \mathbf{t}_z), |\text{cov}(\mathbf{c}_x, \mathbf{t}_z) = - \wedge \text{cov}(\mathbf{c}_y, \mathbf{t}_z) \in \{H, M, L\}$ and viceversa);
- $d_{ov}(\mathbf{v_x}, \mathbf{v_y})$ for all other terms $(\text{cov}(\mathbf{c}_x, \mathbf{t}_z), \text{cov}(\mathbf{c}_y, \mathbf{t}_z) | \text{cov}(\mathbf{c}_x, \mathbf{t}_z) \in \{H, M, L\} \wedge \text{cov}(\mathbf{c}_y, \mathbf{t}_z) \in \{H, M, L\})$.

Therefore, the equivalence with the previous defined distance in the $\mathcal{L}^p$ space is stated by the following equation:

$$d(\mathbf{v^x}, \mathbf{v^y}) = (d_{nov}(\mathbf{v^x}, \mathbf{v^y}) + d_{ov}(\mathbf{v^x}, \mathbf{v^y})) \tag{5.15}$$

When analyzing the contribution of each term, it is possible to classify them using the property of Corollary 1.

$\mathbf{d_{nov}}$: this term is the most significant one for the diagnostic resolution of the CTM. In particular, by considering a generic pair of components $\langle \mathbf{c}_x, \mathbf{c}_y \rangle$ the larger their distance the larger is the number of options in selecting a test able to explain if the component is (or is not) part of the possible faulty candidate set $\mathcal{FCS}$, as defined in Assumption 2. For instance, let $\mathbf{c}_x$ be the covered component. The optimal case for the isolation on a fault on $\mathbf{c}_x$ is that the non-overlapping distance value with respect to any component $\mathbf{c}_y$ in the system under analysis.

A side-effect of such an isolation, from the incremental diagnosis perspective, is the constraint *for each testing sequence to contain* this discriminating test, imposing a lower bound to the reduction of the test

sequence length. In fact, if this is not true, the diagnosis would depend on tests having a *weaker discrimination*, and in general this would imply in increased sensitivity of the model to the definition of $\mathbb{C}_{\mathbb{V}}$.

$\mathbf{d_{ov}}$: this term is less significant with respect to the isolation property. In principle, every distribution of PASS and FAIL outcomes is possible in the subset of tests in the overlapping coverage for $\mathbf{c}_x$ and $\mathbf{c}_y$. However, the diagnosis of the the BBN would depend on the a-priori probability of each *pattern* of outcomes, defined by the coverage values. Thus, the faulty component would be the one *maximizing* such probability. Clearly, the sensitivity to the values in $\mathbb{C}_{\mathbb{V}}$ is even stronger than the d_{nov} case, especially when there are few different labels between $\mathbf{c}_x$ and $\mathbf{c}_y$. Furthermore, the *removal* of faulty candidate from the $\mathcal{FC}_{\mathcal{S}}$ is *weaker* when inferred from a PASS outcome occurs in an overlapping test.

Indeed, the distance equation could be rewritten. The contributions of the single distance terms appearing in the definition are grouped into class values; for instance, $\alpha_{(H,M)} = \|High - Medium\|^p$ is the class of coverage level pair $\{H, M\}$. We use class cardinalities (the number of occurrences of each pair to weight class values); for instance, $n_{(H,M)}$ is the cardinality of $\{H, M\}$. Equation 5.15 can be reformulated as shown in Equation 5.16:

$$d_{nov}(\mathbf{v^x}, \mathbf{v^y}) = (n_{(H,-)} \cdot \alpha_{(H,-)} + n_{(M,-)} \cdot \alpha_{(M,-)} + \dots)^{1/p} \tag{5.16}$$

Equations can be further simplified looking at the first not null $n\_$ term. For instance, when there is at least one $\{H, -\}$ pair such term is $n_{(H,-)} \geq 1$ and Equation 5.16 becomes:

$$d_{nov}(\mathbf{v^x}, \mathbf{v^y}) \approx (n_{(H,-)} \cdot \alpha_{(H,-)})^{1/p} \tag{5.17}$$

If no $\{H, -\}$ pair is present but there is at least one $\{M, -\}$ pair, we obtain:

$$d_{nov}(\mathbf{v^x}, \mathbf{v^y}) \approx (n_{(M,-)} \cdot \alpha_{(M,-)})^{1/p} \tag{5.18}$$

Two considerations are worth making. First, the isolation property evaluated in terms of the distance is a necessary condition for the separability of the faulty candidate among each pair of components. This must occur *at least* when the outcomes of the entire test sequence are available and are used for fault diagnosis. *Partial* syndromes, obtained looking at test outcomes during intermediate steps of the test sequence, might not respect this property, since it depends on the distribution coverage levels in both overlapping and non-overlapping terms.

Second, the p term is a *sensitivity parameter* that can be used to stress the relative weight of the most significative coverage pair $\{H, -\}$

with respect to any other pair ($\{M,–\}$, $\{H,L\}$, …). Let us consider for instance Equation 5.17 and 5.18. Given $\mathbb{C}_{\mathbb{V}}$, all classes ($\alpha_{(H,–)}$, $\alpha_{(M,–)}$, …) assume specific values. Let us consider also the case where $\alpha_{(H,–)} = 1$ (Equation 5.17). In order to obtain the same distance d_{nov} in Equation 5.18, the value of $n_{(M,–)}$ depends on the choice of p. For instance, when $p = 1$ the distance equivalence occurs for $n_{(M,–)} \geq 2$, while when $p = 4$, it must be $n_{(M,–)} \geq 10$. This can be extended by considering the overlapping part of the coverage (i.e., $\{H,L\}$ pairs with respect to $\{H,M\}$ or $\{M,L\}$ pairs), recalling Assumption 2.

Example 22. By referring to the elements of Example 14, it is possible to evaluate the distance between all pairs of AVs. When $p = 4$, we obtain:

$$\begin{aligned} d(\mathbf{v^1},\mathbf{v^2}) &= 1.106 \\ d(\mathbf{v^1},\mathbf{v^3}) &= 0.402 \text{ and} \\ d(\mathbf{v^2},\mathbf{v^3}) &= 1.166. \end{aligned}$$

Considering independently overlapping and non-overlapping terms, we have for pairs $\langle \mathbf{c}_1,\mathbf{c}_2 \rangle$ and $\langle \mathbf{c}_2,\mathbf{c}_3 \rangle$:

$$\begin{aligned} d_{nov}(\mathbf{v^1},\mathbf{v^2}) &= d(\mathbf{v^1},\mathbf{v^2}) = 1.106, \\ d_{nov}(\mathbf{v^2},\mathbf{v^3}) &= d(\mathbf{v^2},\mathbf{v^3}) = 1.166, \end{aligned}$$

but for pair $\langle \mathbf{c}_1,\mathbf{c}_3 \rangle$:

$$\begin{aligned} d_{nov}(\mathbf{v^1},\mathbf{v^3}) &= 0.001 \text{ and} \\ d_{ov}(\mathbf{v^1},\mathbf{v^3}) &= 0.400 \end{aligned}$$

Looking for the minimum of the distance function, component pair $\langle \mathbf{c}_1,\mathbf{c}_3 \rangle$ appears to be *critical*.
On one hand, the reduced distance between vectors $\mathcal{AV}_1$ and $\mathcal{AV}_3$ produces a potential ambiguity in the diagnosis. Such ambiguity is evident when considering, for instance, the complete syndrome [10110].

On the other hand, almost whole distance contribution comes from the overlapping terms $\langle \mathbf{t}_2,\mathbf{t}_5 \rangle$, and this introduces the sensitivity problem on the selection of the numerical value in $\mathbb{C}_{\mathbb{V}}$ for coverage label *Low* ($\mathbf{t}_4$). ◇

5.4 Improving Test Suites' Accuracy

In a testing framework, it is possible to classify the entire set of available tests according to the definitions that follow.

Definition 32. The **Available Test Set (ATS)** is the set of all available tests for the system under analysis.

Tests can be both *adapted* from previous or similar systems or *designed ad-hoc* for the current system. The ATS can be seen as a general test data-base, used by the analyst to define a possible testing strategy.

Definition 33. The **Used Test Set (UTS)** is the set of tests adopted by the test designer to analyze the system.

The UTS usually contains all tests that are executed for generic system verification (fault *detection* and a group of tests considered sufficient to localize the component where a possible fault occurred in. This set is defined to reduce the complexity of the search and selection of the tests to be executed. It is a subset of the ATS.

With respect to the modeling framework proposed in the present chapter, the CTM models the current testing policy thus it refers to the UTS. The proposed algorithm is based on the *incremental modification* of the UTS by adding new tests (eventually taken from the ATS) to obtain a suitable set of tests leading to a CTM characterized by a good isolation property (and a good discrimination capability). The process is based on the evaluation of distance between each pair of AVs.

We propose a *greedy* and *iterative* algorithm (summarized in Figure 5.14), that attacks the ambiguity between each pair of components, resolving the critical situations first.

5.4.1 Algorithm input

The entry point of the algorithm is constituted by the ATS and the CTM, which is used to initialize UTS. The algorithm iteratively transforms the CTM by increasing the size of the UTS. During the process, the original CTM is preserved and no test is removed. This is required because the diagnosis properties of the matrix (required by test designer) depend on those tests, in terms of both *initial test set* minimality (*detection*) and *faulty component isolation* capability (*diagnosis*).

5.4.2 New test selection

A distance table (DT) is computed (line 3) containing the distance between each pair of AVs. It is used to verify if there is any ambiguity between components (line 4). If this is the case, the most critical pair is found (line 5) and an update of UTS is required. The design of a completely new test, not included in any previous set cannot be *inferred* from existing tests; this task would be up to the test or system designer.

Nevertheless, two alternative transformations of the UTS can be implemented at such a high level, and they are described in the following.

Excluded ATS test insertion. The selection of a test contained in the (ATS–UTS) difference set (i.e., an existing test not included in the CTM) is possible (lines 6-9). The search of an optimal test is based exclusively on the coverage of the ambiguous components pair. Such optimal test aims at *maximizing* the distance (Equation 5.14) between the components of the critical pair, should the test be included in the UTS. Such selection induces a ranking, preferring non-overlapping coverages (e.g., $\{H, –\}$) to overlapping ones (e.g., $\{H, L\}$). It is possible that several tests are necessary to solve ambiguity problems; they are added in successive steps of the algorithm.

UTS test modification. Whenever there is no test in the ATS containing the desired coverage combination, the modification of a component coverage in an existing test of the UTS is proposed (lines 10-12). It is worth noting that the modification represents a proposal to both test and system designers, since it requires a modification of components observability with respect to the selected test. In particular, the modification should affect the selected components pair only, and the original test must not be removed from the UTS. There is no constraint to prevent a modification of the coverages of other components in the system to obtain the required observability. From the methodology perspective, the focus on the components pair only provides the designer with a larger degree of freedom for the selection of the best implementation of such a new test. As an example, consider the situation in Figure 5.13. In a pipelined architecture, a test stimulates all components, leading to an *overlapping syndrome* in the CTM. By modifying the observation chain (loopback), a component ($\mathbf{c}_3$) is not covered by test ($\mathbf{t}_1^{*(1)}$), and the coverage of the last component ($\mathbf{c}_2$) of the chain can be improved; otherwise, if direct probe on the intermediate component ($\mathbf{c}_2$) can be inserted ($\mathbf{t}_1^{*(2)}$), the ambiguity can be resolved with respect to the head of the chain ($\mathbf{c}_1$) (see Table 5.5).

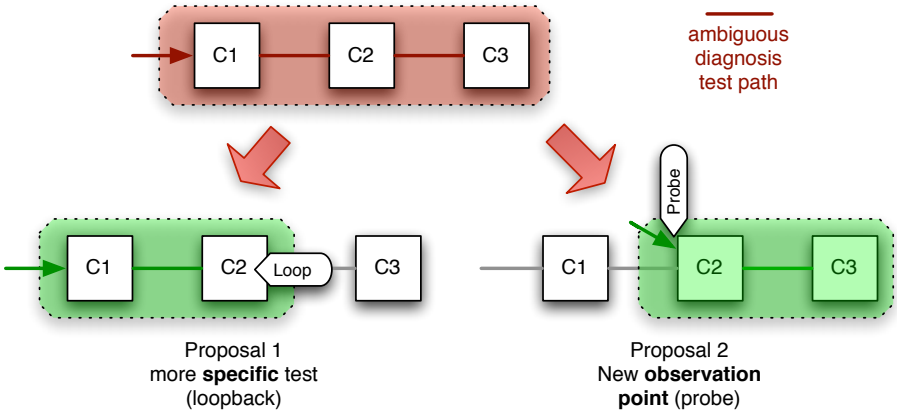


FIGURE 5.13: Suggestion for a new UTS test.

Table 5.5: Test coverage after observability modifications - proposal (1) and (2).

	$\mathbf{t}_1$	$\mathbf{t}_1^{*(1)}$	$\mathbf{t}_1^{*(2)}$
$\mathbf{c}_1$	M	M	–
$\mathbf{c}_2$	M	H	H
$\mathbf{c}_3$	L	–	L

At each iteration, DT is updated with the distances computing using current UTS (line 15).

5.4.3 Algorithm output

Since this approach is intended as a test designer's aid, the proposed solution needs be confirmed step by step as feasible. The algorithm iteratively builds an *Extended* UTS (E_UTS) interpreted by a test/system designer as a reference in two ways. First, it can be considered as the *minimal* modification of the test set necessary to obtain a CTM with an acceptable fault isolation level for all components. Then, since the algorithm incrementally modifies the UTS by starting with covering the most critical situations, the approach can be used as a priority schedule for the design, modification and implementation of a new set of tests to improve the quality and the accuracy of the fault diagnosis process.

Input: System Model (CTM), Available Test Set (ATS).
Output: Modified System Model C$\hat{\text{T}}$M.
Procedure IncreaseFaultIsolation(CTM, ATS)
{
1 $k \leftarrow 0$
2 $UTS(k) = \text{extractTests}(CTM)$
3 $DT \leftarrow \text{distances}(UTS(k))$
4 while (not allComponentIsolated(DT))
5 $(\mathbf{c}_x, \mathbf{c}_y) \leftarrow \text{criticalComponentsPair}(DT)$
6 $t_{ATS} = \text{searchATS\_Test}(\mathbf{c}_x, \mathbf{c}_y, ATS)$
7 if (t_{ATS} exists)
8 $UTS(k+1) \leftarrow UTS(k) \bigcup t_{ATS}$
9 $ATS \leftarrow ATS - t_{ATS}$
10 else
11 $t_{UTS} \leftarrow \text{modifyUTS\_test}(\mathbf{c}_x, \mathbf{c}_y, UTS(k), DT)$
12 $UTS(k+1) \leftarrow UTS(k) \bigcup t_{UTS}$
13 end if
14 $k \leftarrow k+1$
15 $DT \leftarrow \text{distances}(UTS(k))$
16 end while
17 $C\hat{T}M = \text{buildCTM}(UTS(k))$
18 **output** $C\hat{T}M$
}

Figure 5.14: The test set modification algorithm.

5.4.4 Application and results

The algorithm has been integrated in a framework for functional fault diagnosis, and here we report the experimental results on six synthetic CTM, extracted from sample systems, referred to as Board1, ..., Board5. Also a real telecom board has been used to validate the methodology, provided by Cisco Photonics (Cisco1). Their characteristics are summarized in Table 5.6.

Boards 1 to 3 are designed to verify the correct search of tests in the ATS to discriminate the ambiguous components pair. Boards 4 and 5 contain some critical components pairs that cannot be discriminated by any test in the ATS. Cisco board has UTS = ATS allowing UTS test modification only.

Table 5.7 presents the results of the execution of the algorithm. Column 1 reports the initial UTS, Column 2 the number of critical compo-

Table 5.6: Case study boards properties.

Board	**#Components**	**#Tests in UTS**	**#Tests in ATS**
Board1	20	20	40
Board2	20	25	50
Board3	50	33	100
Board4	50	36	100
Board5	50	30	100
Cisco1	55	86	86

nents pairs identified in the original CTM. Column 3 indicates the number of tests in the ATS added to the UTS to remove ambiguities, Column 4 reports the number of critical components pair remaining after test set manipulation.

Columns 5 and 6 present the number of UTS tests to be modified to remove all problems, and the size of the final test set, respectively.

Table 5.7: Result synthesis and comparison.

Board	Init UTS	Crit. *C*s	from ATS	Crit. *C*s	mod UTS	Final UTS (%)
Board1	20	54	7	0	-	27 (+35.00%)
Board2	25	12	3	0	-	28 (+12.00%)
Board3	33	9	3	0	-	36 (+8.33%)
Board4	36	16	4	9	5	45 (+20.00%)
Board5	30	14	4	2	3	37 (+23.33%)
Cisco1	86	29	0	29	11	97 (+12.79%)

In Board1, Board2 and Board3, all critical components pairs are removed by adding existing tests. The increase ratios between the initial and final UTSes are quite different (35% to 8%) but this coefficient can be easily explained considering that smaller boards have a limited degree of freedom for the implementation of orthogonal tests policies, so a larger number of tests is expected to be required to solve specific ambiguities among components pairs. For Board4 and Board5, the number of critical components pairs is quite independent of the required modification, because in the first case multiple ambiguities were solved by the same tests

from UTS, while in the latter case several tests were required to achieve an efficient component diagnosis discrimination.

By referring to the proposed algorithm, it is interesting to look at the distance table (DT), reporting all distances between pairs of components, to get an immediate view of where ambiguity problems may arise. For the last board of our experimental session, Cisco1, we show in Figure 5.15 the computed table, where different colors characterize the different distances, and their criticality. Red areas denote critical distance values related to ambiguous components pairs, that the algorithm will attack first; yellow areas are less critical whereas green ones characterize good distance values.

The application to the real board Cisco1 provided a good validation of the results gathered from the simulation boards.

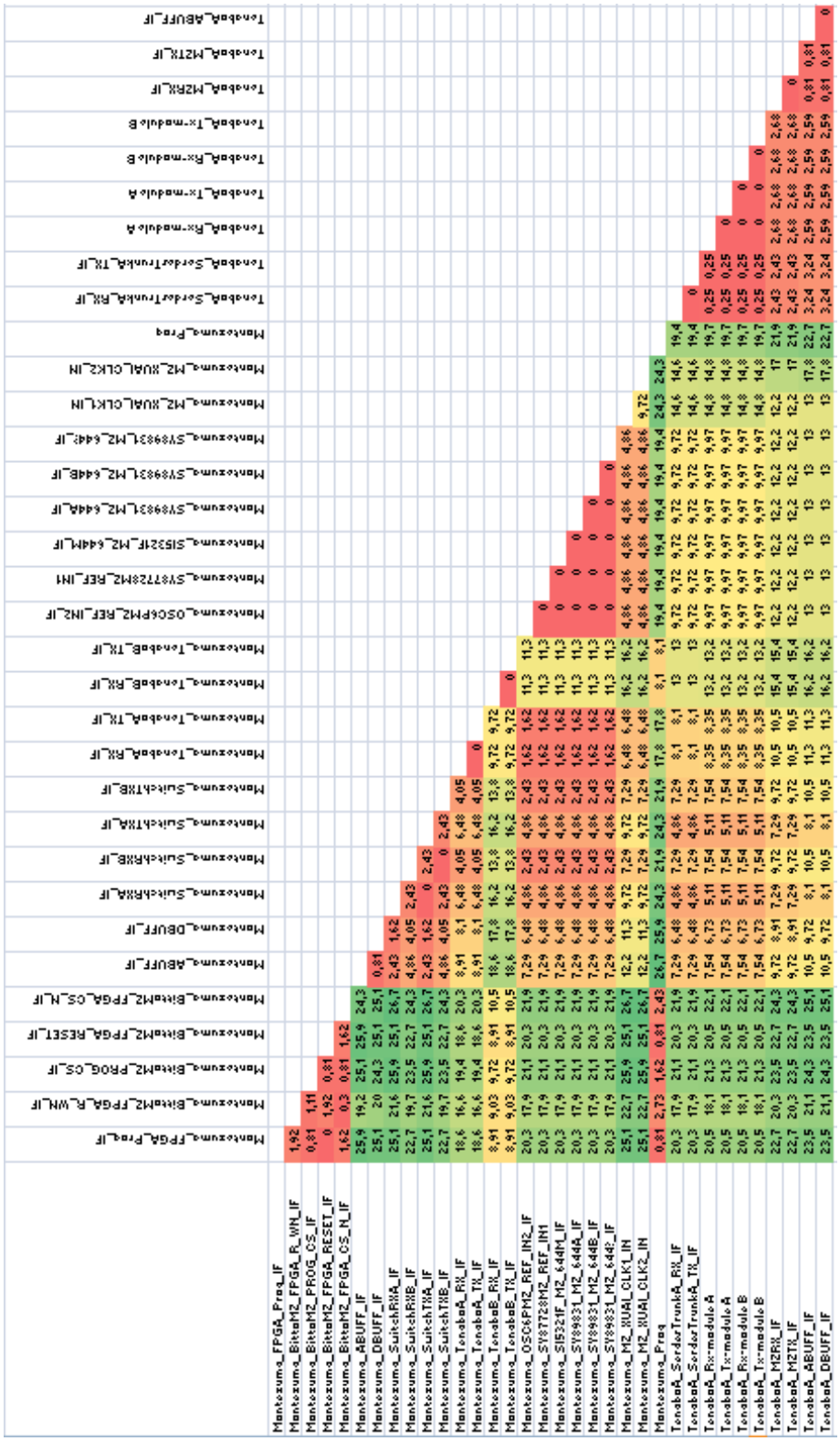


FIGURE 5.15: Distance table for example board Cisco1.

5.5 Chapter summary

In this chapter we presented two possible approach aiming at improve the quality of the AF2D method.

First, an in-depth analysis of the sensitivity of the parameters of the causal model at the basis of BBN-based diagnosis approaches. The approach is relative general and it can be applied to a discrete range of diagnostic BBN-based methodologies. Specifically an evaluation of sensitivity and robustness with respect to the fundamental model at the basis of the AF2D method, focusing on the relations between executed tests, outcomes and syndromes. Experimental analysis performed on synthetic benchmarks have proved the methodology to be robust, able to tolerate non-systematic mistakes the test expert might introduce during the development of the system model.

Furthermore, we introduced a quantitative metric for the evaluation of the diagnostic resolution of test set has been defined, to identify ambiguities in the system model in relation to the selection of the test set, which could affect the quality of a fault diagnosis strategy. The metric have been proposed to overcome the complexity of a direct inspection of the BBN properties, and to simplify an automatic search strategy for a test suite accuracy improvement.

The metric has been defined on a vector space context and its mathematical properties have been exploited to provide a localization criteria for critical ambiguous components pairs. Based on such concept, an incremental algorithm has been proposed to support the user in the definition of a new test set with improved efficiency and effectiveness. Experimental results are reported, showing the correctness and relevance of the approach.

Conclusions 6

In this thesis a methodology called incremental Automatic Functional Fault Detective (AF2D) has been presented to approach the problem of system diagnosis, with a particular focus on the accurate identification of a faulty candidate producing an observable misbehavior.

Given the growing complexity of electronic devices and systems, their diagnosis is increasing in complexity from the point of view of both fault modeling and computational complexity of algorithms designed for automatic fault identification and isolation. The challenge of the design of an effective methodology would impact the efficiency of the manufacturing process, and this is particularly true for the digital system manufacturing field. The sooner a failure root cause is correctly understood, the more important is the reduction of diagnosis time, and the higher is the yield can be achieved.

The proposed AF2D framework exploits a representation of system at high level of abrastraction, using a modeling based on Bayesian Belief Networks (BBNs). The adoption of a model with coarse granularity allows the description of complex, deeply interconnected systems; at the same time, the methodology provides systems and test engineers with the primitives required for the description of relationships between system components and the potential candidates for fault localization, as well as between those latter and the outcomes of the tests of a target diagnostic suite. Probabilities are provided as qualitative labels from a coarse-grained scale and not as quantitative values from a fine-grained scale, in order to simplify and accelerate the system and fault modeling

phase. Underneath the BBN, an inference engine is in charge of performing the exact calculation of the probability of each candidate to be charged as the cause root of the observable symptoms, for both complete and partial syndromes.

Three main research directions within the application of the diagnostic methodoly are tackled in this thesis: the initial scouting of a system for fault detection, the exploration of the solution space for a fast identification of the failure cause, and the quantitative robustness analysis of the models with respect to diagnostic precision and fault isolation resolution.

For the first problem, cost-effective policies to identify a good subset of tests providing good coverage of the system under inspection are proposed, applying an Integer Linear Programming (ILP) optimization approach to the BBN model. Test suites obtained with this method are compared to other test suites generated using more fine-grained models of the same system, in order to provide a reasonable justification for the adoption of a model at an high level of abstraction. Furthermore, an hill-climbing technique is proposed for sorting tests used to scout the system searching a fault, with the aim to optimize the time spent for detection, and to increase the amount of time available for the real diagnostic process. This leads to an improvement of efficiency since the probability of specific failures, with respect to others, happens to be higher in specific time windows of the system manufacturing.

A considerable effort is devoted to the adaptive part of the methodology, targeting the minimization of the cost of each tests session without recurring to a static test sequencing approach. In particular, a geometrical interpretation of BBN parameters is proposed, and a quantitative evaluation through a metric distance within a vector space is used to obtain an optimal step-by-step selection of tests to be executed. Optimization exploits the history the diagnosis for the system under inspection, represented by test outcomes collected during the session. The selection aims at maximizing the relative information that a test outcome would apport to diagnostic conclusions, reducing the amount of redundancy with respect to previously executed tests while minimizing the cost of executing it.

In the same geometrical context, a metric for the identification of a stop condition is proposed, whose purpose is to interrupt the diagnosis when no further information from remaining tests would refine the diagnostic conclusion any longer. Both simulated synthetic systems and

industrial case studies have been used to validate the robustness and the efficiency of the proposal.

In the last part, a sensitivity analysis of the robustness of the diagnostic conclusions is carried on, in order to investigate the claim of correctness of a fault model at an high level of abstraction like the BBN. Given the complexity of such a task, a statistical oriented validation is proposed, developing a quantitative comparison of the BBN model of a system with a fine-grained modeled, well established and mature methodology. Indeed, targeting a stuck-at fault model on combinational circuits, a BBN-compatible model is extracted for benchmark circuits used for fault detection and diagnosis problems, and the results of AF2D methodology are compared with the results provided by an ATPG tool. Statistical correlation underlines how the high-level model can be reasonably used for diagnosis; furthermore, it points out how the amount of detailed information about the internal behavior of the system that cannot be investigated (being unaffordable because of complexity and/or time or cost constraints) is not critical for the obtention of valid system diagnosis.

Besides this problem, an approach is suggested to improve the diagnostic resolution of existing test suites, with a minimal modification both of number of tests (impacting diagnosis time) and of their specific coverage of components (impacting test development effort). This algorithm is verified on synthetic systems and in a real industrial scenario.

6.1 Future extensions

In the future, specific extensions could be proposed to adapt the AF2D methodology in order to accept specific constraints appearing in industrial scenarios. In particular, two main aspects of diagnostic testing have been considering with simplification assumptions, to be relaxed for the application of the methodology on a larger scale.

Test cost evaluation An approach usually implemented in the industry is to consider the test cost as a function of test execution time. The reason for this assumption is the fact that test execution time can be easily extracted from previous diagnostic sessions, if an opportune infrastructure is in place, and its forecast for future instances its relatively accurate.

Other recurrent or non-recurrent test costs (e.g., test set-up operations) are not usually modeled because they are hard to estimate accu-

rately, as for instance their execution time may be subject to a large variability. However, some operation patterns exist that can be used to evaluate the impact of the execution of a test in a specific condition. This evolution of the methodology, has a side effect, would also improve the adaptive test selection policy, taking into account a more reliable estimation of test execution time (or in general, cost) while computing the optimal tradeoff with the information carried by the test outcome.

Test execution constraints When dealing with real industrial contexts, the case where all tests of the test suite can be executed is relatively rare. Rather, it is more likely that there are preconditions to be matched before a particular test can be launched; a set-up or configuration operation might be required, or simply the execution of a preliminary test could be necessary in order to ensure that the system is in a consistent state and the outcome of the targeted test is significant.

Because of this, the exploration for the search of the next test to be executed among the potential candidates of the test suite is to be organized using some model primitives (e.g, test order constraint, pre- and post- conditions, mutual exclusive execution) to be implemented alongside the BBN model, aiming at a more accurate selection of the optimal test sequencing policy.

Application of AF2D in other scenarios The AF2D framework has been used to target a particular family of digital systems, as the experimental results have reported. However, the model is in principle general enough to deal with different family of devices, or even non-electronic systems. For instance, an approach under investigation is the possibility to adopt the methodology in the domain of soft-verification on microprocessors, where the components of the system are the constitutive elements of the microprocessor itself, while tests are in this context particular instruction or microinstruction, whose produced results correspondency with expected values represents the PASS or FAIL outcome. For this goal, given that the modeling complexity appears to be higher, an opportune evolution of the 4-layers BBN fault model is required in order to tune the methodology to target this particular diagnosis.

Bibliography

[ABF90] M. Abramovici, M.A. Breuer, and A.D. Friedman. *Digital systems testing and testable design*. IEEE press New York, 1990.

[Abr05] A. Abraham. Rule-based expert systems. *Handbook of Measuring System Design*, 2005.

[ABS+09] Luca Amati, Cristiana Bolchini, Fabio Salice, F. Franzoso, and al. A incremental approach for functional diagnosis. In *IEEE Intl. Symp. Defect and Fault Tolerance of VLSI Systems.*, pages 392–400, 2009.

[ABS10] Luca Amati, Cristiana Bolchini, and Fabio Salice. Test selection policies for faster incremental fault detection. In *Proc. IEEE Intl Symp on Defect and Fault Tolerance in VLSI Systems, DFT*, pages 310–318, 2010.

[ABS11] Luca Amati, Cristiana Bolchini, and Fabio Salice. Optimal test-set selection for fault diagnosis improvement. In *Proc. IEEE Intl Symp on Defect and Fault Tolerance in VLSI Systems, DFT*, 2011.

[ABSF10a] Luca Amati, Cristiana Bolchini, Fabio Salice, and F. Franzoso. Improving fault diagnosis accuracy by automatic test set modification. In *Proc. IEEE Int. Test Conference*, 2010.

[ABSF10b] Luca Amati, Cristiana Bolchini, Fabio Salice, and Federico Franzoso. A formal condition to stop an incremental automatic functional diagnosis. In *Proc. 13th EUROMICRO Conf. on Digital System Design - Architectures, Methods and Tools*, pages 637–643, 2010.

[AFT08] Rajsekhar Adapa, Edward Flanigan, and Spyros Tragoudas. A novel test generation methodology for

adaptive diagnosis. In *Proc. Intl. Symp. on Quality Electronic Design*, pages 242–245, 2008.

[Agi04] Agilent. *Agilent AFD Tool*, 2004.

[Agr96] G. Agre. Diagnostic bayesian networks. *Computers and Artificial Intelligence*, 16(1), 1996.

[AJA98] AA Al-Jumah and T. Arslan. Artificial neural network based multiple fault diagnosis in digital circuits. In *Circuits and Systems, 1998. ISCAS'98. Proceedings of the 1998 IEEE International Symposium on*, volume 2, pages 304–307. IEEE, 1998.

[ALR04] A. Avižienis, J.C. Laprie, and B. Randell. Dependability and its threats: a taxonomy. *Building the Information Society*, pages 91–120, 2004.

[Ama11a] Luca Amati. Optimal Test-Suite Sequencing for Bayesian-Network Based Diagnosis. Technical Report 2011.3, Politecnico di Milano, 2011.

[Ama11b] Luca Amati. Parameters Sensitivity Analysis in Bayesian Network-Based Diagnosis. Technical Report 2011.3, Politecnico di Milano, 2011.

[BBMR05] A. Beygelzimer, M. Brodie, S. Ma, and I. Rish. Test-based diagnosis: Tree and matrix representations. In *Integrated Network Management, 2005. IM 2005. 2005 9th IFIP/IEEE International Symposium on*, pages 529–542. IEEE, 2005.

[BdJV+08a] R. Boumen, I. de Jong, J. Vermunt, JM van de Mortel-Fronczak, and JE Rooda. Risk-based stopping criteria for test sequencing. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(6):1363–1373, 2008.

[BDJV+08b] R. Boumen, ISM De Jong, JWH Vermunt, JM van de Mortel-Fronczak, and JE Rooda. Test sequencing in complex manufacturing systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(1):25–37, 2008.

[Bel06] C.M. Bishop and SpringerLink (Service en ligne). *Pattern recognition and machine learning*, volume 4. Springer New York, 2006.

[BMSS09] S. Biswas, R. Mall, M. Satpathy, and S. Sukumaran. A model-based regression test selection approach for embedded applications. *ACM SIGSOFT Software Engineering Notes*, 34(4):1–9, 2009.

[BPG05] Z. Bluvband, R. Polak, and P. Grabov. Bouncing failure analysis (bfa): the unified fta-fmea methodology. In *Reliability and Maintainability Symposium, 2005. Proceedings. Annual*, pages 463 – 467, 24-27, 2005.

[BRdJ+09] R. Boumen, S. Ruan, I. de Jong, JM Van De Mortel-Fronczak, JE Rooda, and KR Pattipati. Hierarchical test sequencing for complex systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(3):640–649, 2009.

[BS07a] S. Butcher and J. Sheppard. Improving diagnostic accuracy by blending probabilities: Some initial experiments. In *Proceedings of the 18th International Workshop on Principles of Diagnosis (DX-07)*. Citeseer, 2007.

[BS07b] S.G.W. Butcher and J.W. Sheppard. Asset-specific Bayesian diagnostics in mixed contexts. In *Autotestcon, 2007 IEEE*, pages 113–122. IEEE, 2007.

[BS09] S. Butcher and J.W. Sheppard. Distributional Smoothing in Bayesian Fault Diagnosis. *Instrumentation and Measurement, IEEE Transactions on*, 58(2):342–349, 2009.

[BV04] S.P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge Univ Pr, 2004.

[CF02] M. Catelani and A. Fort. Soft fault detection and isolation in analog circuits: some results and a comparison between a fuzzy approach and radial basis function networks. *Instrumentation and Measurement, IEEE Transactions on*, 51(2):196–202, 2002.

[Cor01] T.H. Cormen. *Introduction to algorithms*. The MIT press, 2001.

[CPU07] Y. Chen, R.L. Probert, and H. Ural. Model-based regression test suite generation using dependence analysis. In *Proceedings of the 3rd international workshop on Advances in model-based testing*, pages 54–62. ACM, 2007.

[CSK+09] K. Choi, S. Singh, A. Kodali, K.R. Pattipati, J.W. Sheppard, S.M. Namburu, S. Chigusa, D.V. Prokhorov, and L. Qiao. Novel classifier fusion approaches for fault diagnosis in automotive systems. *Instrumentation and Measurement, IEEE Transactions on*, 58(3):602–611, 2009.

[CZL+04] M. Chen, AX Zheng, J. Lloyd, MI Jordan, and E. Brewer. Failure diagnosis using decision trees. In *Autonomic Computing, 2004. Proceedings. International Conference on*, pages 36–43. IEEE, 2004.

[Dar03] A. Darwiche. A differential approach to inference in bayesian networks. *Journal of the ACM (JACM)*, 50(3):280–305, 2003.

[DB97] AL Dexter and M. Benouarets. Model-based fault diagnosis using fuzzy matching. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 27(5):673–682, 1997.

[DC93] P. Dagum and R.M. Chavez. Approximating probabilistic inference in bayesian belief networks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(3):246–255, 1993.

[DD01] FJ Dıez and M. Druzdzel. Fundamentals of canonical models. In *Proc. IX Conferencia de la Asociacin Espanola para la Inteligencia Artificial.*, 2001.

[Der00] L. Derere. Case-based reasoning: diagnosis of faults in complex systems through reuse of experience. In *Test Conference, 2000. Proceedings. International*, pages 27–34. IEEE, 2000.

[DH00] D. Du and F. Hwang. *Combinatorial group testing and its applications*. World Scientific Pub Co Inc, 2000.

[DKW87] J. De Kleer and B.C. Williams. Diagnosing multiple faults. *Artificial intelligence*, 32(1):97–130, 1987.

[DL97] P. Dagum and M. Luby. An optimal approximation algorithm for bayesian inference. *Artificial Intelligence*, 93(1-2):1–27, 1997.

[DM03] P. Drineas and Y. Makris. Independent test sequence compaction through integer programming. In *Proc. Int. Conf. Computer Design*, pages 380–386, 2003.

[Eri05] Cliford A. Ericson. *Hazard Analysis Techniques for System Safety*. Wiley, 2005.

[ETB07] E. Echavarria, T. Tomiyama, and GJW Bussel. Fault diagnosis approach based on a model-based reasoner and a functional designer for a wind turbine. an approach towards self-maintenance. 75:012078, 2007.

[FMM01] W.G. Fenton, T.M. McGinnity, and L.P. Maguire. Fault diagnosis of electronic systems using intelligent techniques: A review. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 31(3):269–281, 2001.

[FMM05] B. Fenton, TM McGinnity, and L.P. Maguire. Intelligent systems technology in the fault diagnosis of electronic systems. *Intelligent Knowledge-Based Systems*, pages 1276–1313, 2005.

[GH02] H. Guo and W. Hsu. A survey of algorithms for real-time bayesian network inference. In *Joint Workshop on Real-Time Decision Support and Diagnosis*, 2002.

[Ha94] DS Ha. ATALANTA: An ATPG Tool. *Bradley Dep. Electrical Engineering, Virginia Polytechnic and State University*, 1994.

[HCdK92] W. Hamscher, L. Console, and J. de Kleer, editors. *Readings in model-based diagnosis*. Morgan Kaufmann Publishers Inc., 1992.

[Hec08] D. Heckerman. A tutorial on learning with bayesian networks. *Innovations in Bayesian Networks*, pages 33–82, 2008.

[Hen87] M. Henrion. Some practical issues in constructing belief networks. In *Proc. Conf. on Uncertainty in Artificial Intelligence*, pages 161–174, 1987.

[Hen88] M. Henrion. Propagating uncertainty in bayesian networks by probabilistic logic sampling. 2:149–163, 1988.

[HKO+09] Y. Higami, Y. Kurose, S. Ohno, H. Yamaoka, H. Takahashi, Y. Shimizu, T. Aikyo, and Y. Takamatsu. Diagnostic test generation for transition faults using a stuck-at ATPG tool. In *Proc. Int. Test Conference*, pages 1–9, 2009.

[HL90] D.E. Heckerman and Stanford University. Knowledge Systems Laboratory. *A tractable inference algorithm for diagnosing multiple diseases*. Citeseer, 1990.

[HMDPJ08] Y. Huang, R. McMurran, G. Dhadyalla, and R. Peter Jones. Probability based vehicle fault diagnosis: Bayesian network method. *Journal of Intelligent Manufacturing*, 19(3):301–311, 2008.

[HN84] S.L. Hakimi and K. Nakajima. On adaptive system diagnosis. *IEEE Transactions on Computers*, pages 234–240, 1984.

[HST+06] Y. Higami, K.K. Saluja, H. Takahashi, S. Kobayashi, and Y. Takamatsu. Compaction of pass/fail-based diagnostic test vectors for combinational and sequential circuits. In *Proc. Asia and South Pacific Conf Design Automation*, page 6, 2006.

[HTFF05] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

[HW09] S. Holst and H.J. Wunderlich. Adaptive debug and diagnosis without fault dictionaries. *Journal of Electronic Testing*, 25(4):259–268, 2009.

[IEC90] IEC IEC. 61025: Fault tree analysis. Technical report, 1990.

[Ise05] R. Isermann. Model-based fault-detection and diagnosis-status and applications. *Annual Reviews in control*, 29(1):71–85, 2005.

[Ise06] R. Isermann. *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer, 2006.

[Jen96] F.V. Jensen. *An introduction to Bayesian networks*, volume 210. UCL press London, 1996.

[JLO90] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational statistics quarterly*, 4:269–282, 1990.

[KDBK10] S. Krishnan, K.D. Doornbos, R. Brand, and H.G. Kerkhoff. Block-level Bayesian diagnosis of analogue electronic circuits. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1767–1772, 2010.

[KP83] J.H. Kim and J. Pearl. A computational model for causal and diagnostic reasoning in inference systems. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 190–193. Citeseer, 1983.

[KPFS92] K. Kubiak, S. Parkes, WK Fuchs, and R. Saleh. Exact evaluation of diagnostic test resolution. In *Design Automation Conference, 1992. Proceedings., 29th ACM/IEEE*, pages 347–352. IEEE, 1992.

[KSC+08] A. Kodali, S. Singh, K. Choi, K. Pattipati, S.M. Namburu, S. Chigusa, D.V. Prokhorov, and L. Qiao. Dynamic set-covering for real-time multiple fault diagnosis. In *Aerospace Conference, 2008 IEEE*, pages 1–11. IEEE, 2008.

[KY95] G.J. Klir and B. Yuan. *Fuzzy sets and fuzzy logic*. Prentice Hall New Jersey, 1995.

[LC05] C. Liu and K. Chakrabarty. Design and analysis of compact dictionaries for diagnosis in scan-bist. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 13(8):979–984, 2005.

[LCLH98] D.B. Lavo, B. Chess, T. Larrabee, and I. Hartanto. Probabilistic mixed-model fault diagnosis. International Test Conference, 1998.

[Lee01] B.H. Lee. Using bayes belief networks in industrial fmea modeling and analysis. In *Reliability and Maintainability Symposium, 2001. Proceedings. Annual*, pages 7–15. IEEE, 2001.

[LGTL85] W.S. Lee, DL Grosh, F.A. Tillman, and C.H. Lie. *Fault Tree Analysis, Methods, and Applications A Review*, volume 34. IEEE, 1985.

[LHH07] Z. Li, M. Harman, and R.M. Hierons. Search algorithms for regression test case prioritization. *IEEE Transactions on Software Engineering*, pages 225–237, 2007.

[LHZQ09] S. Lin, Z. He, Y. Zhang, and Q. Qian. An evolutionary ann based on rough set and its application in power grid fault diagnosis. In *Intelligent Systems and Applications. ISA 2009. International Workshop on*, pages 1–4. IEEE, 2009.

[LS88] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 157–224, 1988.

[LS98] Vasilica Lepar and Prakash P. Shenoy. A comparison of lauritzen-spiegelhalter, hugin, and shenoy-shafer architectures for computing marginals of probability distributions. In *Proc. 14th Conf. on Uncertainty in Artificial Intelligence*, pages 328–337. Morgan Kaufmann, 1998.

[Mac03] D.J.C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge Univ Pr, 2003.

[ME02] D. Manley and B. Eklow. A model based automated debug process. In *Proc. IEEE Intl. Board Test Workshop*, pages 3–1, 2002.

[Mit97] T.M. Mitchell. *Machine learning*. McGraw Hill, 1997.

[MJ98] A.L. Madsen and F.V. Jensen. Lazy propagation in junction trees. In *In Proc. 14th Conf. on Uncertainty in Artificial Intelligence*. Citeseer, 1998.

[MR04a] Arnljot Hoyland Marvin Rausand. *System Reliability Theory: Models, Statistical Methods, and Applications*. Wiley, 2004.

[MR04b] Arnljot Hoyland Marvin Rausand. *System Reliability Theory: Models, Statistical Methods, and Applications*. Wiley, 2004.

[Mur98] Kevin Murphy. A brief introduction to graphical models and bayesian networks, 1998.

[NASa] NASA. *Fault Tree Handbook with Aerospace Applications*.

[NASb] NASA. *Probabilistic Risk Assessment Procedures Guide*.

[Nik00] D. Nikovski. Constructing bayesian networks for medical diagnosis from incomplete and partially correct statistics. *Knowledge and Data Engineering, IEEE Transactions on*, 12(4):509–516, 2000.

[NJ98] A.E. Nicholson and N. Jitnah. Using mutual information to determine relevance in bayesian networks. In *Proc. Int. Conf. Artificial Intelligence: Topics in AI, LNCS 1531*, pages 399–410, 1998.

[NM09] S.N. Neophytou and M.K. Michael. Test set generation with a large number of unspecified bits using static and dynamic techniques. *IEEE Transactions on Computers*, pages 301–316, 2009.

[NSL08] M. Natu, A.S. Sethi, and E.L. Lloyd. Efficient probe selection algorithms for fault diagnosis. *Telecommunication Systems*, 37(1):109–125, 2008.

[NUR] NUREG. *Fault Tree Handbook*.

[OMCE05] C. O'Farrill, M. Moakil-Chbany, and B. Eklow. Optimized reasoning-based diagnosis for non-random, board-level, production defects. In *IEEE International Conference on Test, 2005.*, page 7. IEEE, 2005.

[PCM05] C. Pous, J. Colomer, and J. Melendez. Improving Fault Dictionary Techniques with Artificial Intelligence Methods for Linear Electronic Analog Circuits Diagnosis. 2005.

[PCMdlR03] C. Pous, J. Colomer, J. Melendez, and J. de la Rosa. Case base management for analog circuits diagnosis improvement. *Case-Based Reasoning Research and Development*, pages 1065–1065, 2003.

[Pea86] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artificial intelligence*, 29(3):241–288, 1986.

[Pea00] J. Pearl. *Causality*, volume 1. Cambridge University Press, Cambridge, UK, 2000.

[PFT+07] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, et al. *Numerical recipes*. Cambridge Univ Press, 3 edition, 2007.

[Pom11] I. Pomeranz. Gradual diagnostic test generation and observation point insertion based on the structural distance between indistinguished fault pairs. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 2011.

[PR87a] Y. Peng and J.A. Reggia. A probabilistic causal model for diagnostic problem solving part i: Integrating symbolic causal inference with numeric probabilistic inference. *Systems, Man and Cybernetics, IEEE Transactions on*, 17(2):146–162, 1987.

[PR87b] Y. Peng and J.A. Reggia. A probabilistic causal model for diagnostic problem solving part ii: Diagnostic strategy. *Systems, Man and Cybernetics, IEEE Transactions on*, 17(3):395–406, 1987.

[PR97] I. Pomeranz and S.M. Reddy. On dictionary-based fault location in digital logic circuits. *Computers, IEEE Transactions on*, 46(1):48–59, 1997.

[PR00] I. Pomeranz and S.M. Reddy. A diagnostic test generation procedure based on test elimination by vector omission for synchronous sequential circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 19(5):589–600, 2000.

[PR02] I. Pomeranz and S.M. Reddy. On dictionary-based fault location in digital logic circuits. *IEEE Trans. Computers*, 46(1):48–59, 2002.

[PR10] I. Pomeranz and S.M. Reddy. Gradual diagnostic test generation based on the structural distance between indistinguished fault pairs. In *2010 25th International Symposium on Defect and Fault Tolerance in VLSI Systems*, pages 349–357. IEEE, 2010.

[PRO07] Reliability analysis tools and their use, 2007.

[PRR02] I. Pomeranz, L.N. Reddy, and SM Reddy. COMPACTEST: A method to generate compact test sets for combinational circuits. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 12(7):1040–1049, 2002.

[Pub11] ReliaSoft Publishing. Reliability engineering resource web site, 2011.

[Pyta] Python. Python documentation.

[Pytb] Python. Python/c api reference manual.

[RB85] J. Richman and K. R. Bowden. The modern fault dictionary. In *Proc. Intl. Test Conf.*, pages 696–702, 1985.

[RBM+05] I. Rish, M. Brodie, S. Ma, N. Odintsova, A. Beygelzimer, G. Grabarnik, and K. Hernandez. Adaptive diagnosis in distributed systems. *Neural Networks, IEEE Transactions on*, 16(5):1088–1109, 2005.

[RH93] G. Rothermel and M.J. Harrold. A safe, efficient algorithm for regression test selection. In *Software Maintenance, 1993. CSM-93, Proceedings., Conference on*, pages 358–367. IEEE, 1993.

[Ris06] I. Rish. Information-theoretic approaches to cost-efficient diagnosis. 2006.

[RSP+99a] V. Raghavan, M. Shakeri, K. Pattipati, M. Inc, and MA Natick. Optimal and near-optimal test sequencing algorithms with realistictest models. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 29(1):11–26, 1999.

[RSP+99b] V. Raghavan, M. Shakeri, KR Pattipati, M. Inc, and MA Natick. Test sequencing problems arising in test planning and design fortestability. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 29(2):153–163, 1999.

[RTPPH04] S. Ruan, F. Tu, K.R. Pattipati, and A. Patterson-Hine. On a multimode test sequencing problem. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 34(3):1490–1499, 2004.

[RUCH01] G. Rothermel, R.H. Untch, C. Chu, and M.J. Harrold. Prioritizing test cases for regression testing. *IEEE Transactions on software engineering*, pages 929–948, 2001.

[RZY+09] S. Ruan, Y. Zhou, F. Yu, K.R. Pattipati, P. Willett, and A. Patterson-Hine. Dynamic multiple-fault diagnosis with imperfect tests. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 39(6):1224–1236, 2009.

[SA09] M.A. Shukoor and V.D. Agrawal. A Two Phase Approach for Minimal Diagnostic Test Set Generation. In *2009 European Test Symposium*, pages 115–120. IEEE, 2009.

[SB06] J. Sheppard and S. Butcher. On the Linear Separability of Diagnostic Models. In *AUTOTESTCON Conference Record, New York: IEEE Press*. Citeseer, 2006.

[SB07] JW Sheppard and S.G.W. Butcher. A formal analysis of fault diagnosis with d-matrices. *Journal of Electronic Testing*, (4):309–322, 2007.

[SBB+98] J. Saxena, K.M. Butler, H. Balachandran, D.B. Lavo, B. Chess, T. Larrabee, and F.J. Ferguson. On applying non-classical defect models to automated diagnosis. In *Test Conference, 1998. Proceedings., International*, pages 748–757. IEEE, 1998.

[SBKM06] J. Sheppard, S. Butcher, M. Kaufman, and C. MacDougall. Not-so-naive bayesian networks and unique identification in developing advanced diagnostics. Citeseer, 2006.

[She92] J.W. Sheppard. Explanation-based learning with diagnostic models. In *AUTOTESTCON'92. IEEE Systems Readiness Technology Conference, Conference Record*, pages 159–166. IEEE, 1992.

[Shu09] M.A. Shukoor. *Fault Detection and Diagnostic Test Set Minimization*. PhD thesis, 2009.

[SK05] J.W. Sheppard and M.A. Kaufman. A Bayesian approach to diagnosis and prognosis using built-in test. *Instrumentation and Measurement, IEEE Transactions on*, 54(3):1003–1018, 2005.

[SMH+91] M.A. Shwe, B. Middleton, DE Heckerman, M. Henrion, EJ Horvitz, HP Lehmann, and GF Cooper. Probabilistic diagnosis using a reformulation of the internist-1/qmr knowledge base. *Methods of information in Medicine*, 30(4):241–255, 1991.

[SQL] SQLite. Sqlite.

[SR93] J. Stephens and V. Raghavan. On Single-Fault Set Diagnosability in the PMC Model. *IEEE Transactions on Computers*, 42(8):981–983, 1993.

[Sri93] S. Srinivas. A generalization of the noisy-or model. In *Proceedings of the ninth conference on uncertainty in artificial intelligence*, pages 208–215, 1993.

[SS91a] W.R. Simpson and J.W. Sheppard. An intelligent approach to automatic test equipment. In *Proc. Int. Test Conf*, pages 419–425. Citeseer, 1991.

[SS91b] W.R. Simpson and J.W. Sheppard. System complexity and integrated diagnostics. *Design & Test of Computers, IEEE*, 8(3):16–30, 1991.

[SS94] W.R. Simpson and J.W. Sheppard. *System test and diagnosis*. Springer, 1994.

[SS96a] J.W. Sheppard and W.R. Simpson. Improving the accuracy of diagnostics provided by fault dictionaries. In *VLSI Test Symposium, 1996., Proceedings of 14th*, pages 180–185. IEEE, 1996.

[SS96b] W.R. Simpson and J.W. Sheppard. Encapsulation and diagnosis with fault dictionaries. In *AUTOTESTCON'96,'Test Technology and Commercialization'. Conference Record*, pages 441–446. IEEE, 1996.

[SS98] J.W. Sheppard and W.R. Simpson. Managing conflict in system diagnosis. *Computer*, 31(3):69–76, 1998.

[SS04a] M. Steinder and AS Sethi. Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. *Computer Networks*, 45(4):537–562, 2004.

[SS04b] M. Steinder and A.S. Sethi. A survey of fault localization techniques in computer networks. *Science of Computer Programming*, 53(2):165–194, 2004.

[SS08] P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. *Classic Works of the Dempster-Shafer Theory of Belief Functions*, pages 499–528, 2008.

[SSB06] KM Silva, BA Souza, and NSD Brito. Fault detection and classification in transmission lines based on wavelet transform and ann. *Power Delivery, IEEE Transactions on*, 21(4):2058–2063, 2006.

[Sta03] D.H. Stamatis. *Failure mode and effect analysis: FMEA from theory to execution*. Asq Pr, 2003.

[SW06] J.W. Sheppard and T.J. Wilmering. Recent advances in ieee standards for diagnosis and diagnostic maturation. In *Aerospace Conference, 2006 IEEE*, pages 9–pp. IEEE, 2006.

[SYAU07] F. Sahin, M.Ç. Yavuz, Z. Arnavut, and O. Uluyol. Fault diagnosis for airplane engines using bayesian networks and distributed particle swarm optimization. *Parallel Computing*, 33(2):124–143, 2007.

[TCXAS09] Yongning Tang, Guang Cheng, Zhiwei Xu, and Ehab Al-Shaer. Community-base fault diagnosis using incremental belief revision. In *Proc. Int. Conf. Networking, Architecture, and Storage*, pages 121–128, 2009.

[TK08] S. Theodoridis and K. Koutroumbas. *Pattern recognition*. Academic Press, 2008.

[Too03] S.T. Tools. *TetraMAX ATPG*, 2003.

[TP03] F. Tu and KR Pattipati. Rollout strategies for sequential fault diagnosis. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 33(1):86–99, 2003.

[Tur97] J. Turino. Test economics in the 2lst century. *Design Test of Computers, IEEE*, 1997.

[VRYK03] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S.N. Kavuri. A review of process fault detection and diagnosis:: Part i: Quantitative model-based methods. *Computers & Chemical Engineering*, 27(3):293–311, 2003.

[VWE+06] T. Vo, Z. Wang, T. Eaton, P. Ghosh, H. Li, Y. Lee, W. Wang, H. Jun, R. Fang, D. Singletary, et al. Design for Board and System Level Structural Test and Diagnosis. In *Test Conference, 2006. ITC'06. IEEE International*, pages 1–10. IEEE, 2006.

[WSKR06] K.R. Walcott, M.L. Soffa, G.M. Kapfhammer, and R.S. Roos. Timeaware test suite prioritization. In *Proceedings of the 2006 international symposium on Software testing and analysis*, pages 1–12. ACM, 2006.

[WWZP09] B. Wang, W. Wei, W. Zeng, and K.R. Pattipati. Fault localization using passive end-to-end measurement and sequential testing for wireless sensor networks. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON'09. 6th Annual IEEE Communications Society Conference on*, pages 1–10. IEEE, 2009.

[ZCRT07] W. Zou, W.T. Cheng, S.M. Reddy, and H. Tang. Speeding up effect-cause defect diagnosis using a small dictionary. In *VLSI Test Symposium, 2007. 25th IEEE*, pages 225–230. IEEE, 2007.

[ZRB05] A.X. Zheng, I. Rish, and A. Beygelzimer. Efficient test selection in active diagnosis via entropy approximation. In *Proceedings of UAI*, volume 5. Citeseer, 2005.

[ZWGC09] Z. Zhang, Z. Wang, X. Gu, and K. Chakrabarty. Physical defect modeling for fault insertion in system reliability test. In *Test Conference, 2009. ITC 2009. International*, pages 1–10. IEEE, 2009.

[ZWGC10a] Z. Zhang, Z. Wang, X. Gu, and K. Chakrabarty. Board-level fault diagnosis using bayesian inference. In *28th VLSI Test Symposium (VTS)*, pages 244–249. IEEE, 2010.

[ZWGC10b] Z. Zhang, Z. Wang, X. Gu, and K. Chakrabarty. Board-level fault diagnosis using an error-flow dictionary. In *IEEE International Test Conference*, 2010.

[ZWGC10c] Z. Zhang, Z. Wang, X. Gu, and K. Chakrabarty. Optimization and Selection of Diagnosis-Oriented Fault-Insertion Points for System Test. In *2010 19th IEEE Asian Test Symposium*, pages 429–432. IEEE, 2010.

List of Acronyms

BBN Bayesian Belief Network i

AI Artificial Intelligence 19

ES Expert Systems 19

AF2D incremental Automatic Functional Fault Detective i

CTM Components-Tests Matrix 1

initTS Initial Test Set 88

ILP Integer Linear Programming i

nOR Noisy-OR 56

JPD Joint Probability Distribution 49

PSV Partial Syndrome Vector

VCN Valid Conclusion Node 65

FCC Faulty Candidate Components 18

AV Attraction Vector 131

RR Rejection Region 131

ATE Automatic Test Equipment 33

ANN Artificial Neural Networks 37

CBR Case-Based Reasoning 38

ATPG Automatic Test Pattern Generator 88

DAG Direct Acyclic Graph 50

List of Figures

List of Tables