ELSEVIER

Reliability Engineering and System Safety 71 (2001) 249–260

RELIABILITY ENGINEERING & SYSTEM SAFETY

www.elsevier.com/locate/ress

Improving the analysis of dependable systems by mapping fault trees into Bayesian networks

A. Bobbio[a], L. Portinale[a], M. Minichino[b], E. Ciancamerla[b,*]

[a]*Dipartimento di Scienze e Tecnologie Avanzate, Università del Piemonte Orientale "A. Avogadro", C.so Borsalino 54, 15100 Alessandria, Italy*
[b]*ENEA–CRE Casaccia, Via Anguillarese 301, 00060 Rome, Italy*

Abstract

Bayesian Networks (BN) provide a robust probabilistic method of reasoning under uncertainty. They have been successfully applied in a variety of real-world tasks but they have received little attention in the area of dependability. The present paper is aimed at exploring the capabilities of the BN formalism in the analysis of dependable systems. To this end, the paper compares BN with one of the most popular techniques for dependability analysis of large, safety critical systems, namely Fault Trees (FT). The paper shows that any FT can be directly mapped into a BN and that basic inference techniques on the latter may be used to obtain classical parameters computed from the former (i.e. reliability of the Top Event or of any sub-system, criticality of components, etc). Moreover, by using BN, some additional power can be obtained, both at the modeling and at the analysis level. At the modeling level, several restrictive assumptions implicit in the FT methodology can be removed and various kinds of dependencies among components can be accommodated. At the analysis level, a general diagnostic analysis can be performed. The comparison of the two methodologies is carried out by means of a running example, taken from the literature, that consists of a redundant multiprocessor system.

Keywords: Dependable systems; Probabilistic methods; Bayesian networks; Fault tree analysis

1. Introduction

Fault Tree Analysis (FTA) is a very popular and diffused technique for the dependability modeling and evaluation of large, safety-critical systems [1,2], like the Programmable Electronic Systems (PES). The technique is based on the identification of a particular undesired event to be analyzed (e.g. system failure), called the Top Event (TE). The construction of the Fault Tree (FT) proceeds in a top/down fashion, from the events to their causes, until failures of basic components are reached. The methodology is based on the following assumptions: (i) events are binary events (working/not-working); (ii) events are statistically independent; and (iii) relationships between events and causes are represented by means of logical AND and OR gates. However, some FT tools relax the last assumption and allow the inclusion of the NOT gate and related (e.g. XOR) gates. In FTA, the analysis is carried out in two steps: a qualitative step in which the logical expression of the TE is derived in terms of prime implicants (the minimal cut-sets); a quantitative step in which, on the basis of the probabilities assigned to the failure events of the basic components, the probability of occurrence of the TE (and of any internal event corresponding to a logical sub-system) is calculated.

On the other hand, Bayesian Networks (BNs) have become a widely used formalism for representing uncertain knowledge in probabilistic systems and have been applied to a variety of real-world problems [3]. BNs are defined by a directed acyclic graph in which discrete random variables are assigned to each node, together with the conditional dependence on the parent nodes. Root nodes are nodes with no parents, and marginal prior probabilities are assigned to them. The main feature of BN is that it is possible to include local conditional dependencies into the model, by directly specifying the causes that influence a given effect.

The quantitative analysis of a BN may proceed along two lines. A forward (or predictive) analysis, in which the probability of occurrence of any node of the network is calculated on the basis of the prior probabilities of the root nodes and the conditional dependence of each node. A more standard backward (diagnostic) analysis that concerns the computation of the posterior probability of any given set of variables given some observation (the evidence), represented as instantiation of some of the variables to one of their admissible values.

The aim of the present paper is to compare the modeling

* Corresponding author.
E-mail address: ciancamerlae@casaccia.enea.it (E. Ciancamerla).

0951-8320/01/$ - see front matter
PII: S0951-8320(00)00077-6

and the decision power of the FTA and BN methodologies in the area of the dependability analysis. First, an algorithm is presented to convert an FT into a BN and it is shown how the results obtained from a FT analysis can be cast in the BN setting. Subsequently, various modeling extensions, available in the BN language [4–6], are investigated. In particular, it is shown how the deterministic binary AND/OR connection among components can be overcome, by introducing probabilistic gates. Special and convenient forms of probabilistic gates are the *noisy* gate and gate with *leak*. Moreover, *n*-ary (or multi-state) components can be easily accommodated into the picture, and various kinds of common-cause-failure dependencies can be taken into account. The inclusion of local dependencies in a BN may avoid a complete state-space description (as it is required in Markovian or Petri net models), making the formalism an appealing candidate for dependability analysis. However, to the best of our knowledge, very few attempts are documented in the literature [4,5,7,8] to import the BN formalism in the area of dependability.

At the analysis level, beyond the usual measures available in FT analysis, the BN methodology is able to perform a diagnostic assessment, given some observation, and to compute measures of the severity of each single component, or of any subset of components jointly considered, conditioned on the occurrence of the TE.

Dependability engineers are accustomed to deal with structured and easy-to-handle tools that provide a guideline for building up models starting from the system description. The present work is aimed at showing that it is possible and convenient to combine a structured methodology like FT with the modeling and analytical power of BN. The computation of posterior probabilities given some evidence can be particularized to obtain very natural importance measures (like, for instance, the posterior probability of the basic components given the TE), or backtrace diagnostic information [6]. On the other hand, the modeling flexibility of the BN formalism can accommodate various kinds of statistical dependencies that cannot be included in the FT formalism.

The comparison of the two methodologies is carried on through the analysis of an example. The example (taken from Ref. [9]), consists of a redundant multiprocessor system, with local and shared memories, local mirrored disks and a single bus. A brief description of the basic characteristics of BNs is reported in Section 2. Section 3 describes the mapping algorithm from FT to BN and illustrates the conversion on the chosen example. Various new modeling issues are considered in Section 4, while Section 5 is devoted to discuss new analysis issues offered by BN.

2. Bayesian networks

Bayesian Networks (also known as *belief nets*, *causal networks*, *probabilistic dependence graphs*, etc.) are a widely used formalism for representing uncertain knowledge in Artificial Intelligence [10,11]. They have become the standard methodology for the construction of systems relying on probabilistic knowledge and have been applied in a variety of real-worlds tasks [3]. The main features of the formalism are a graphical encoding of a set of conditional independence assumptions and a compact way of representing a joint probability distribution between random variables.

Bayesian Networks [10] are usually defined on discrete random variables, even if some extensions have been proposed for extending the formalism to some form of continuous random variables; this paper, however, deals only with the discrete case.

A BN is a pair $N = \langle\langle V, E\rangle, P\rangle$ where $\langle V, E\rangle$ are the nodes and the edges of a Directed Acyclic Graph (DAG), respectively, and P is a probability distribution over V. Discrete random variables $V = \{X_1, X_2, \ldots, X_n\}$ are assigned to the nodes, while the edges E represent the causal probabilistic relationship among the nodes.

In a BN, we can then identify a qualitative part (the topology of the network represented by the DAG and a quantitative part (the conditional probabilities). The qualitative part represents a set of conditional independence assumptions that can be captured through a graph-theoretic notion called *d-separation* [10]. This notion has been shown to model the usual set of independence assumptions that a modeler assumes when considering each edge from variable X to variable Y as a direct dependence (or as a cause–effect relationship) between the events represented by the variables.

The quantitative analysis is based on the conditional independence assumption. Given three random variables X, Y, Z, X is said to be conditionally independent from Y given Z if $P(X, Y|Z) = P(X|Z)P(Y|Z)$. Because of these assumptions, the quantitative part is completely specified by considering the probability of each value of a variable conditioned by every possible instantiation of its parents (i.e. by considering only local conditioning). These local conditional probabilities are specified by defining, for each node, a Conditional Probability Table (CPT). The CPT contains, for each possible value of the variables associated to a node, all the conditional probabilities with respect to all the combination of values of the variables associated to the parent nodes. Variables having no parents are called *root variables* and marginal prior probabilities are associated with them. According to these assumptions (*d-separation* and conditional independence), the joint probability distribution P of random variables V can be factorized as in Eq. (1)

$$P[X_1, X_2, \ldots, X_n] = \prod_{i=1}^{n} P[X_i|\text{Parent}(X_i)] \tag{1}$$

The basic inference task of a BN consists of computing the posterior probability distribution on a set of query variables Q, given the observation of another set of variables

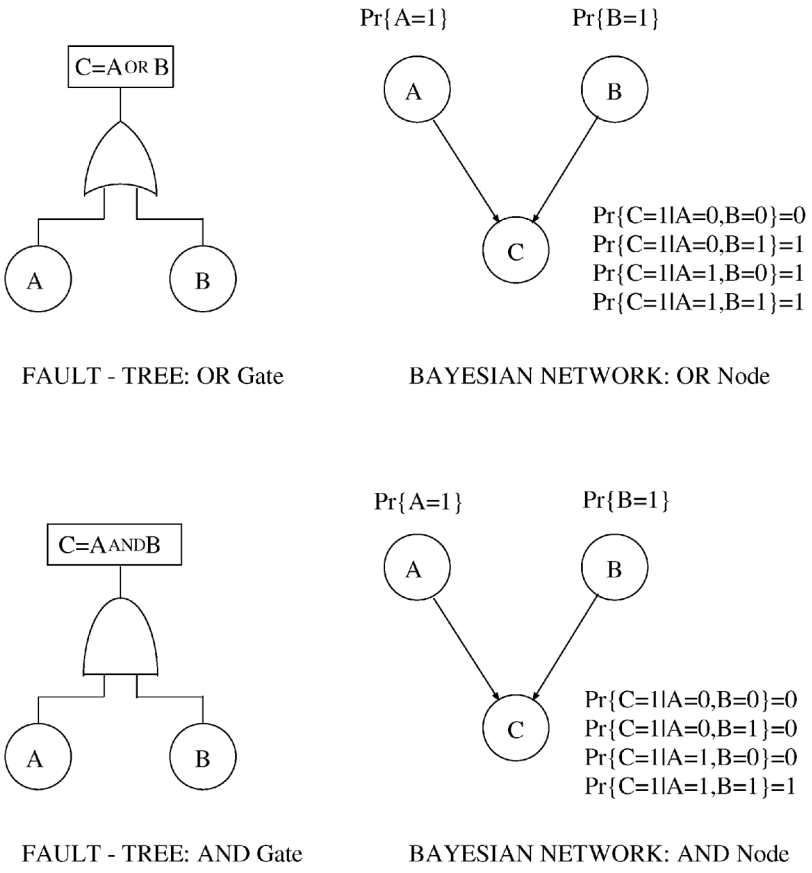

Fig. 1. The OR and AND gate in FT and BN representation.

E called the *evidence* (i.e $P(Q|E)$). Particular attention has been paid to an instantiation of the above problem in which the query set *Q* is a singleton composed of just one variable and the problem is applied to each variable of the net (but the evidence ones). Different classes of algorithms have been developed that compute the marginal posterior probability $P(X|E)$ for each variable *X*, given the evidence *E*. While this computation may be sufficient in several applications, there may be cases requiring the computation of the posterior joint probability of a given set *Q* of variables. In Section 5 these measures and their computation are revisited.

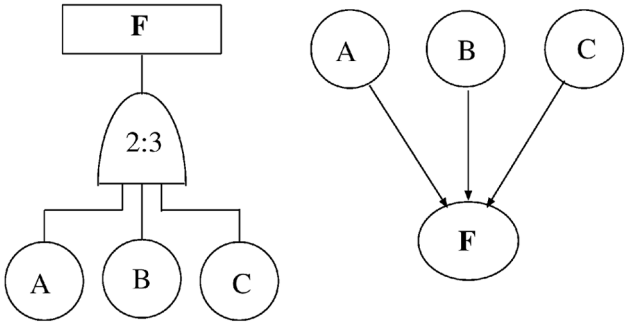

Fig. 2. The 2:3 gate in FT and BN representation.

3. Mapping fault trees to Bayesian networks

In order to arrive to the conversion algorithm from FT to BN, the basic assumptions of the standard FTA methodology are recalled:

(i) events are binary events (working/not-working);
(ii) events are statistically independent;
(iii) relationships between events and causes are represented by logical AND and OR gates;
(iv) the root of the FT is the undesired Top Event (TE), to be analyzed.

We adopt the following convention. Given a generic binary component C we denote with $C = 1$ or simply with *C* the component failure and with $C = 0$ or $\bar{C}$ the component working. The quantification of the FT requires the assignment of a probability value to each leaf node. Since the computation is performed at a given mission time *t*, the failure probabilities of the basic components at time *t* should be provided. In the usual hypothesis that component failures

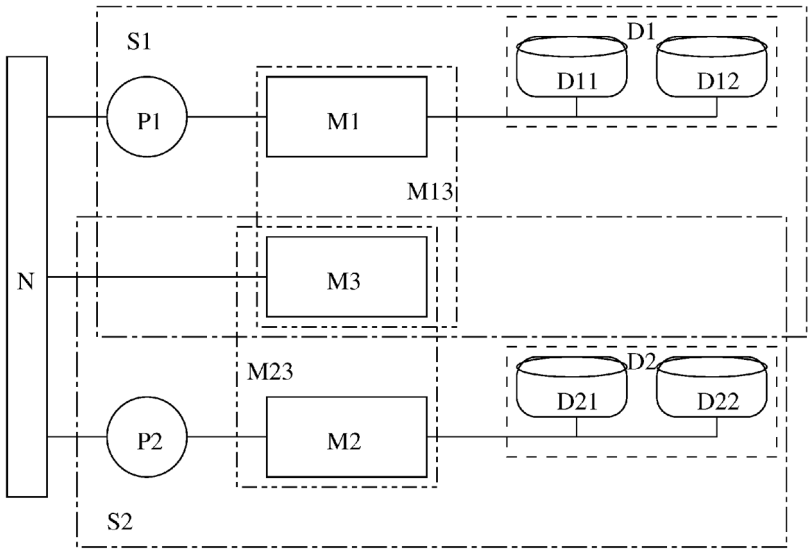

Fig. 3. A redundant multiprocessor system.

are exponentially distributed, the probability of occurrence of the primary event ($C = 1 =$ faulty) is $P(C = 1, t) = 1 - e^{-\lambda_C t}$, where λ_C is the *failure rate* of component C.

We first show how an FT can be converted into an equivalent BN and then we show, in Section 4, how assumptions (i)–(iii) can be relaxed in the new formalism. To proceed step by step, Fig. 1 shows the conversion of an OR and an AND gate into equivalent nodes in a BN. Parent nodes A and B are assigned prior probabilities (coincident with the probability values assigned to the corresponding basic nodes in the FT), and child node C is assigned its CPT. Since, the OR and AND gates represent deterministic causal relationships, all the entries of the corresponding CPT are either 0s or 1s.

A common extension to many FT packages is to have implicit ($k : n$) gates. Implicit gate means, that the FT solver must explicit the Boolean expression of the ($k : n$) function. The BN representation of a (2:3) is given in Fig. 2. The CPT assigned to node F has the following entries (these are also either 0s or 1s):

$$\begin{aligned}
&\Pr\{F = 1|A = 0, B = 0, C = 0\} = 0\\
&\Pr\{F = 1|A = 0, B = 0, C = 1\} = 0\\
&\Pr\{F = 1|A = 0, B = 1, C = 0\} = 0\\
&\Pr\{F = 1|A = 1, B = 0, C = 0\} = 0\\
&\Pr\{F = 1|A = 0, B = 1, C = 1\} = 1\\
&\Pr\{F = 1|A = 1, B = 0, C = 1\} = 1\\
&\Pr\{F = 1|A = 1, B = 1, C = 0\} = 1\\
&\Pr\{F = 1|A = 1, B = 1, C = 1\} = 1
\end{aligned} \tag{2}$$

It is clear that, any Boolean function from the nodes A, B and C to the node F can be made explicit in the BN representation of Fig. 2 by only modifying the corresponding CPT.

According to the translation rules for the basic gates, it is straightforward to map an FT into a binary BN, i.e. a BN with every variable V having two admissible values: *false* ($\bar{V}$) corresponding to a *normal* or *working* value and *true* (V) corresponding to a *faulty* or *not-working* value. The conversion algorithm proceeds along the following steps:

- for each *leaf node* (i.e. primary event or system component) of the FT, create a *root node* in the BN; however, if more leaves of the FT represent the same primary event (i.e. the same component), create just one root node in the BN;
- assign to root nodes in the BN the prior probability of the corresponding leaf node in the FT (computed at a given mission time t);
- for each *gate* of the FT, create a corresponding *node* in the BN;
- label the node corresponding to the gate whose output is the TE of the FT as the *Fault* node in the BN;
- connect nodes in the BN as corresponding gates are connected in the FT;
- for each gate (OR, AND or k:n) in the FT assign the equivalent CPT to the corresponding node in the BN (see Figs. 1 and 2).

Due to the very special nature of the gates appearing in a FT, non-root nodes of the BN are actually deterministic nodes and not random variables and the corresponding CPT can be assigned automatically. The prior probabilities on the root nodes are coincident with the corresponding probabilities assigned to the leaf nodes in the FT.

The mapping algorithm is illustrated through the following example concerning the redundant multiprocessor system shown in Fig. 3 and taken from Ref. [9]. The system

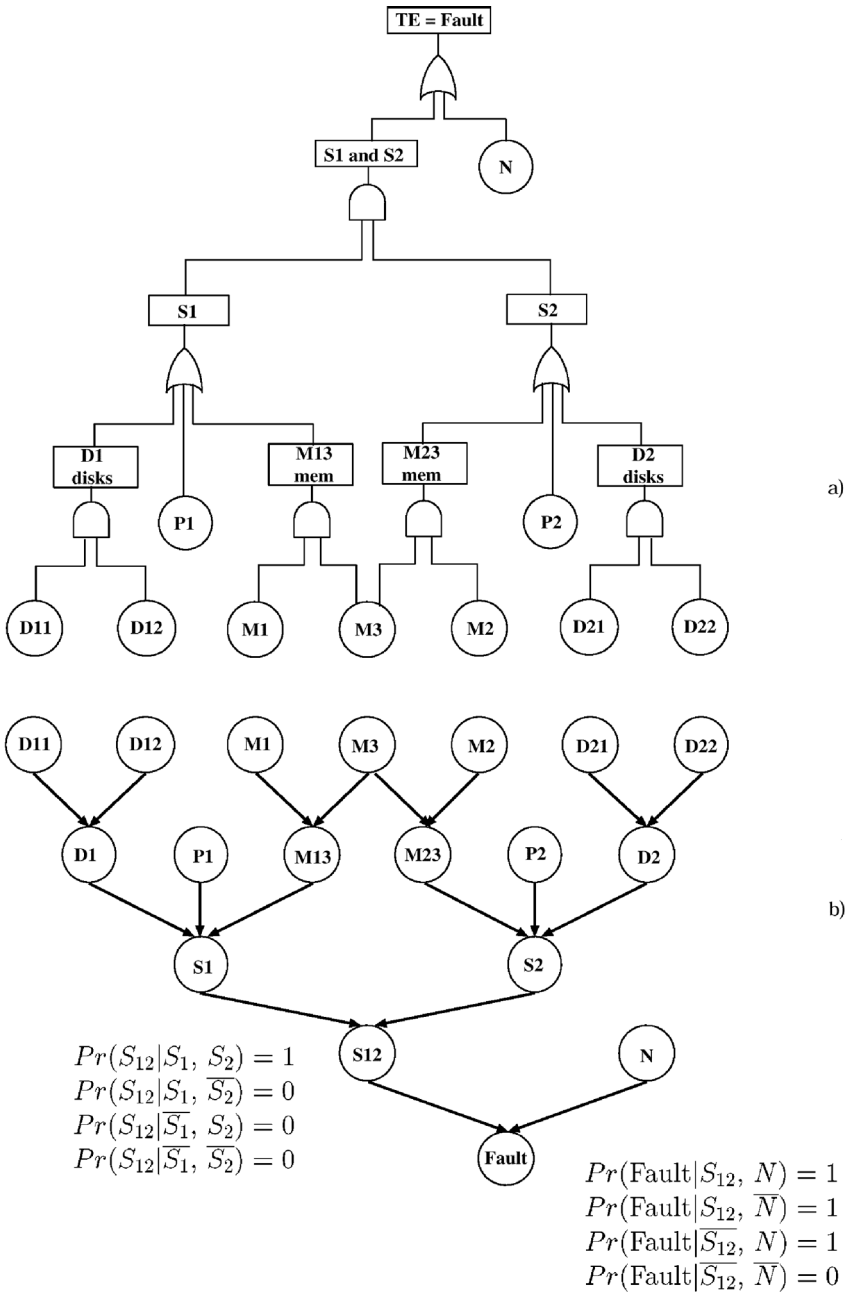

Fig. 4. (a) Fault tree and (b) Bayesian network for the multiprocessor system.

is composed of a bus N connecting two processors P_1 and P_2 having access to a local memory bank each (M_1 and M_2) and, through the bus to a shared memory bank M_3, so that if the local memory bank fails, the processor can use the shared one. Each processor is connected to a mirrored disk unit. If one of the disks fails, the processor switches on the mirror. The whole system is functional if the bus N is functional and one of the processing subsystems is functional. Fig. 3 also shows the partitioning into logical subsystems, i.e. the processing subsystems S_i ($i = 1, 2$), the mirrored disk units D_i ($i = 1, 2$) and the memory subsystems M_{i3} ($i = 1, 2$). The FT for this system is shown in Fig. 4a. The logical expression of the TE as a function of the *minimal cut sets* is given by the following expression:

$$\begin{aligned}\text{TE} = {} & N + D_{11}D_{12}D_{21}D_{22} + D_{11}D_{12}M_2M_3 + D_{11}D_{12}P_2 \\ & + M_1M_3D_{21}D_{22} + M_1M_2M_3 + M_1M_3P_2 + P_1D_{12}D_{22} \\ & + P_1M_2M_3 + P_1P_2 \end{aligned} \quad (3)$$

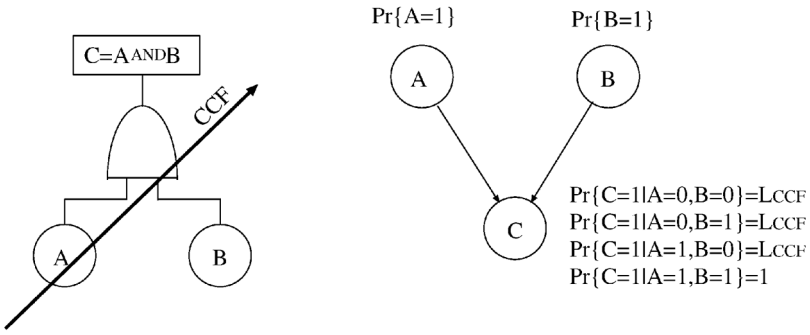

Fig. 5. The CCF representation in BN.

For ease of comparison, and for evidencing the connection between leaf nodes in the FT and root nodes in the BN, the structure of the corresponding BN is represented in the same Fig. 4b. As an example, in Fig. 4b, the CPT entries for the node *Fault* (corresponding to an OR gate) and for the node S_{12} (corresponding to an AND gate) are also shown. In order to quantify both models, the failure probabilities of each component are then assigned to the leaf nodes of the FT, and to the root nodes of the BN as prior probability (see Section 5.1).

4. Modeling issues

The mapping procedure described in Section 3 shows that each FT can be naturally converted into a BN. However, BNs are a more general formalism than FTs; for this reason, there are several modeling aspects underlying BNs that may make them very appealing for dependability analysis. In the following sections, we examine a number of extensions of the standard FT methodology, and we show how they can be cast into a BN framework.

4.1. Probabilistic gates: common cause failures

Differently from FT, the dependence relations among variables in a BN are not restricted to be deterministic. This corresponds to being able to model uncertainty in the behavior of the *gates*, by suitably specifying the conditional probabilities in the *CPT* entries. Probabilistic gates may reflect an imperfect knowledge of the system behavior, or may avoid the construction of a more detailed and refined model. A typical example is the incorporation of Common Cause Failures (CCF). Common cause failures are usually modeled in FT by adding an OR gate, directly connected to the TE, in which one input is the system failure, the other input the CCF leaf to which the probability of failure due to common causes is assigned. In the BN formalism, such additional constructs are not necessary, since the probabilistic dependence is included in the CPT. Fig. 5 shows an AND gate with CCF and the corresponding BN. The value L_{CCF} is the probability of failure of the system due to common causes, when one or both components are up.

4.2. Noisy gates

Of particular attention for reliability aspects is one peculiar modeling feature often used in building BN models: *noisy gates*. As mentioned in Section 2, when specifying CPT entries one has to condition the value of a variable on every possible instantiation of its parent variables, making the number of required entries exponential with respect to the number of parents. By assuming that the node variable is influenced by any single parent independently of the other ones (*disjunctive interaction* [10]), noisy gates reduce this effort by requiring a number of parameters linear in the number of parents.

Consider for example the subsystem S_1 in Fig. 3: it fails if either the disk unit D_1 or the processor P_1 or the memory subsystem M_{13} fails. Since node S_1 in the BN of Fig. 4 has three parent nodes, this implies that the modeler has to provide eight CPT entries for completely specify this local model.[1] Of course if this local model is a deterministic logical OR (as in the example), only three probabilities (equal to 1) are sufficient. Consider now the case where the logical OR interaction is *noisy* or probabilistic: even if one of the components of S_1 fails, there is a (possibly small) positive probability that the subsystem works. This correspond to the fact that the system may maintain some functionality or may be able to reconfigure (with some probability of reconfiguration success) in the presence of particular faults.

BN can still avoid the complete CPT specification by adopting the so-called *noisy-or model*. Given a binary variable Y having the set of parent binary variables $X_1,\ldots,X_n$, the noisy-or model requires to specify n parameters $p_1,\ldots,p_n$, where each p_i is interpreted as the probability of Y true given that X_i is true and all the other parents are false (i.e. $p_i = P(Y|\bar{X}_1,\ldots X_i,\ldots,\bar{X}_n)$). By assuming that each X_i

[1] The whole local model is composed of 16 entries, but only 8 have to be provided independently because $P(Y|X) = 1 - P(\bar{X}|Y)$.

influences Y independently from each other, the local model is completely specified if we further assume that Y is false if none of the parents is true. Indeed it can be shown [10] that if X is a particular instantiation of $X_1,\ldots,X_n$ and π_x is the set of true variables in X, then

$$P(Y|X) = 1 - \prod_{X_i \in \pi_x} (1 - p_i) \tag{4}$$

Returning to the example, we may for instance model the fact that S_1 works with probability 0.01 even if the disk unit D_1 has failed. This means that $p_{D_1} = P(S_1|D_1, \bar{P}_1, \bar{M}_{13}) = 0.99$; suppose now that the system can also recover if other components of S_1 fail, but with a smaller probability (e.g. $p_{P_1} = p_{M_{13}} = 0.995$). Then we can, for instance, compute the probability of the subsystem S_1 failed when both D_1 and P_1 have failed and M_{13} is still working as $P(S_1|D_1, P_1, \bar{M}_{13}) = 1 - (0.01 \times 0.005) = 0.99995$. As one can expect, such a probability is larger than the probability of S_1 failed given that only D_1 has failed.

The noisy gate does not allow to model the fact that the subsystem may fail even if all its components are functional, since it assumes that $P(S_1|\bar{D}_1, \bar{P}_1, \bar{M}_{13}) = 0$. However, we have already noticed that it is often necessary, in reliability modeling, to include causes of failure (usually referred to as *common causes*) that determine the system to go down even in the presence of components up. To this end, an extension of the noisy-or can be adopted, called *noisy-or* with *leak*. In this model, one assumes that there is a positive probability (called the *leak* or *background probability*) of having Y true even when all parents X_i are false. This can be modeled by considering the influence between X_i and Y as changed by adding an unknown parent L: the leak (or the common cause in reliability terminology). In this way $P(Y|\bar{X}_1,\ldots,\bar{X}_n)$ is interpreted as $P(Y|\bar{X}_1,\ldots,\bar{X}_n, L) = l$ and then Eq. (4) becomes

$$P(Y|X) = 1 - \left[(1 - l) \prod_{X_i \in \pi_X} (1 - p_i) \right] \tag{5}$$

In the example, we may assign to S_1 a common cause failure probability $l_{cc} = 0.02$, even if all components are functional. This term accounts for the fact that we missed some unknown cause of failure, either because we do not know it precisely or because we do not deem appropriate to build up a finer representation for the system. In this case, if we still compute the probability of the subsystem S_1 failed when both D_1 and P_1 have failed and M_{13} is still working, we obtain $P(S_1|D_1, P_1, \bar{M}_{13}) = 1 - [0.01 \times 0.005(1 - l_{cc})] = 0.999951$ which is slightly larger than in the previous case when no leak probability was present.

Similar considerations can be made for variables corresponding to AND gates of the FT, by taking into account the *noisy-and model* (dual to noisy-or). The modeler has to specify the parameters $p_i = P(Y|X_1,\ldots,\bar{X}_i,\ldots X_n)$ for getting

$$P(Y|X) = \prod_{X_i \notin \pi_X} p_i \tag{6}$$

Consider for instance, the mirrored disk subsystem D_1 that fails if both disks D_{11} and D_{12} fail. However, in a more refined view of the model, we can suppose that the mirrored connection is not perfect, and there is a small probability (e.g. 0.001) that the disk subsystem D_1 fails when a single disk is up: (i.e. $p_{D_{11}} = P(D_1|\bar{D}_{11}, D_{12}) = p_{D_{12}} = P(D_1|D_{11}, \bar{D}_{12}) = 1 \times 10^{-3}$). We can then compute the probability of D_1 failing when both disks are functional as $P(D_1|\bar{D}_{11}, \bar{D}_{12}) = 10^{-3} \times 10^{-3} = 1 \times 10^{-6}$.

Notice that the noisy-and gate is not appropriate to model the *coverage* [12] probability of the reconfiguration process in a redundant system (where the coverage c is defined as the probability that the reconfiguration process is able to restore the system in a working state when a redundant element fails). Suppose that in the memory subsystem M_{13} there is a small probability (e.g. $1 - c$) that the subsystem M_{13} does not recover the failure of a single memory bank: (i.e. $p_{M_1} = P(M_{13}|\bar{M}_1, M_3) = p_{M_3} = P(M_{13}|\bar{M}_3, M_1) = 1 - c$).

The above assumption does not entail that M_{13} must have a positive probability of failure when both banks are functional, and the complete CPT must be specified in this case. In fact, the noisy-and model assumes that the subsystem certainly fails when all involved components fail (i.e. $P(M_{13}|M_1, M_3) = 1$). If this is not the case, the noisy-and can be generalized with a leak parameter as the noisy-or.

4.3. Multi-state variables

All the above considerations concerned binary variables. The use of *multi-state* or *n*-ary variables can be very useful in many applications [13–15], where it is not sufficient to restrict the component behavior to the dichotomy working/not-working. Typical scenarios are possible that require the incorporation of *multi-state* components: the possible presence of various failure modes [13,14] (short versus open, stuck at 0 versus stuck at 1, covered versus uncovered), the different effect of the failure modes on the system operation (e.g. fail-safe/fail-danger), or various performance levels between normal operation and failure [15] (Section 4.4).

By dealing with variables having more than two values, BNs can allow the modeler to represent a multi-valued component by means of different values of the variable representing the component itself.

Suppose to consider a three-state component whose states are identified as *working* (*w*), *fail-open* (*f-o*) and *fail-short* (*f-s*). In FT the component failure modes must be modeled as two independent binary events (*w*/*f-o*) and (*w*/*f-s*); however, to make the model correct, an XOR gate must be inserted between *f-o* and *f-s* since they are mutually exclusive events. On the contrary, BN can include *n*-ary variables by adjusting the entries of the *CPT*. A useful generalization of the noisy-gate construct is called *noisy-max*, and may be applied to *n*-ary variables, with the only

constraint of having an order defined on the set of the admissible values of the variables [16].

4.4. Sequentially dependent failures

Another modeling issue that may be quite problematic to deal with using FT is the problem of components failing in some dependent way. For instance, the abnormal operation of a component may induce dependent failures on other ones. Suppose for instance that we refine the description of the multiprocessor system by adding the component power supply (PS) such that, when failing, causes a system failure, but it may also induce the processors to break down. In the FT representation, a new input PS should be added to the TE of Fig. 4 to represent a new cause of system failure as shown in Fig. 6a; however, modeling the dependence between the failure of PS and the failure of processor P_i ($i = 1, 2$) is not possible in the FT formalism.

Notice that, in the BN model one may be even more precise, by resorting to a multi-state model for the power supply; indeed, a more realistic situation could be the following: PS is modeled with three possible modes: *correct*, *defective* and *dead* where the first corresponds to a nominal behavior, the second to a defective working mode where an abnormal voltage is provided, while the last corresponds to a situation where PS cannot work at all. Of course, *dead* mode causes the whole system to be down, but we want to model the fact that when PS is in the *defective* mode the processors increase their conditional dependence to break down. This can be modeled in a very natural way, by considering the variable PS to have three values corresponding to the above modes and by setting the entries in the CPT in a suitable way.

5. Analysis issues

Typical analyses performed on a FT involve both qualitative and quantitative aspects. In particular, any kind of quantitative analysis exploits the basics of the qualitative analysis, thus the minimal cut-sets computation. Minimal cut-sets are the prime implicants of the TE and are usually obtained by means of minimization techniques on the set of logical functions represented by the Boolean gates of the FT. Given the set of minimal cut-sets, usual quantitative analysis involves:

- the computation of the overall unreliability of the system corresponding to the unreliability of the TE (i.e. $P(Fault)$);
- the computation of the unreliability of each identified subsystem, corresponding to the unreliability of each single gate;
- the importance of each minimal cut-set, corresponding to the probability of the cut-set itself by assuming the statistical independence among components.

In particular, if each component c_i has probability of failure $P(c_i)$, the importance of a cut-set CS is given by $P(\mathrm{CS}) = \prod_{c_i \in \mathrm{CS}} P(c_i)$. Notice that such a quantity refers to the a-priori failure probability of each component.

As shown in Section 3, any FT can be mapped into a BN where non-root nodes are deterministic. Any analysis performed on a FT can be performed on the corresponding BN; moreover, other interesting measures can be obtained from the BN, that cannot be evaluated in an FT. Let us first consider the basic analyses of a FT and how they are performed in the corresponding BN:

- *unreliability of the TE:* this corresponds to computing the prior probability of the variable *Fault*, that is $P(Q|E)$ with $Q = Fault$ and $E = 0$;
- *unreliability of a given subsystem:* this corresponds to computing the prior probability of the corresponding variable S_i, that is $P(Q|E)$ with $Q = S_i$ and $E = 0$.

Differently from the computations performed on an FT, the above computations in a BN do not require the determination of the cut-sets. However, any technique used for cut-set determination in the FT can be applied in the BN: indeed, the Boolean functions modeled by the gates in the FT are modeled by non-root nodes in the BN.

Concerning the computation of the cut-set importance, it is worth noting that BN may directly produce a more accurate measure of such an importance, being able to provide the posterior probability of each cut-set given the fault. Indeed, posing a query having the node *Fault* as evidence and the root variables R as queried variables allows one to compute the distribution $P(R|Fault)$; this means that the posterior probability of each mode of each component (just *working* and *faulty* in the binary case) can be obtained. Once cut-sets are known, the computation of the posterior importance is just a matter of marginalization on $P(R|Fault)$.

Related to the above issue is another aspect that is peculiar of the use of BN with respect to FT: the possibility of performing *diagnostic problem-solving* on the modeled system. In fact, in many cases the system analyst may be interested in determining the possible explanations of an exhibited fault in the system. Cut-set determination is a step in this direction, but it may not be sufficient in certain situations. Classical diagnostic inference on a BN involves:

- computation of the posterior marginal probability distribution on each component;
- computation of the posterior joint probability distribution on subsets of components;
- computation of the posterior joint probability distribution on the set of all nodes, but the evidence ones.

The first kind of computation is perhaps the most popular one when using BN for diagnosis (see for instance DXpress [16] or MSBN [17] or HUGIN [18] tools). One advantage is

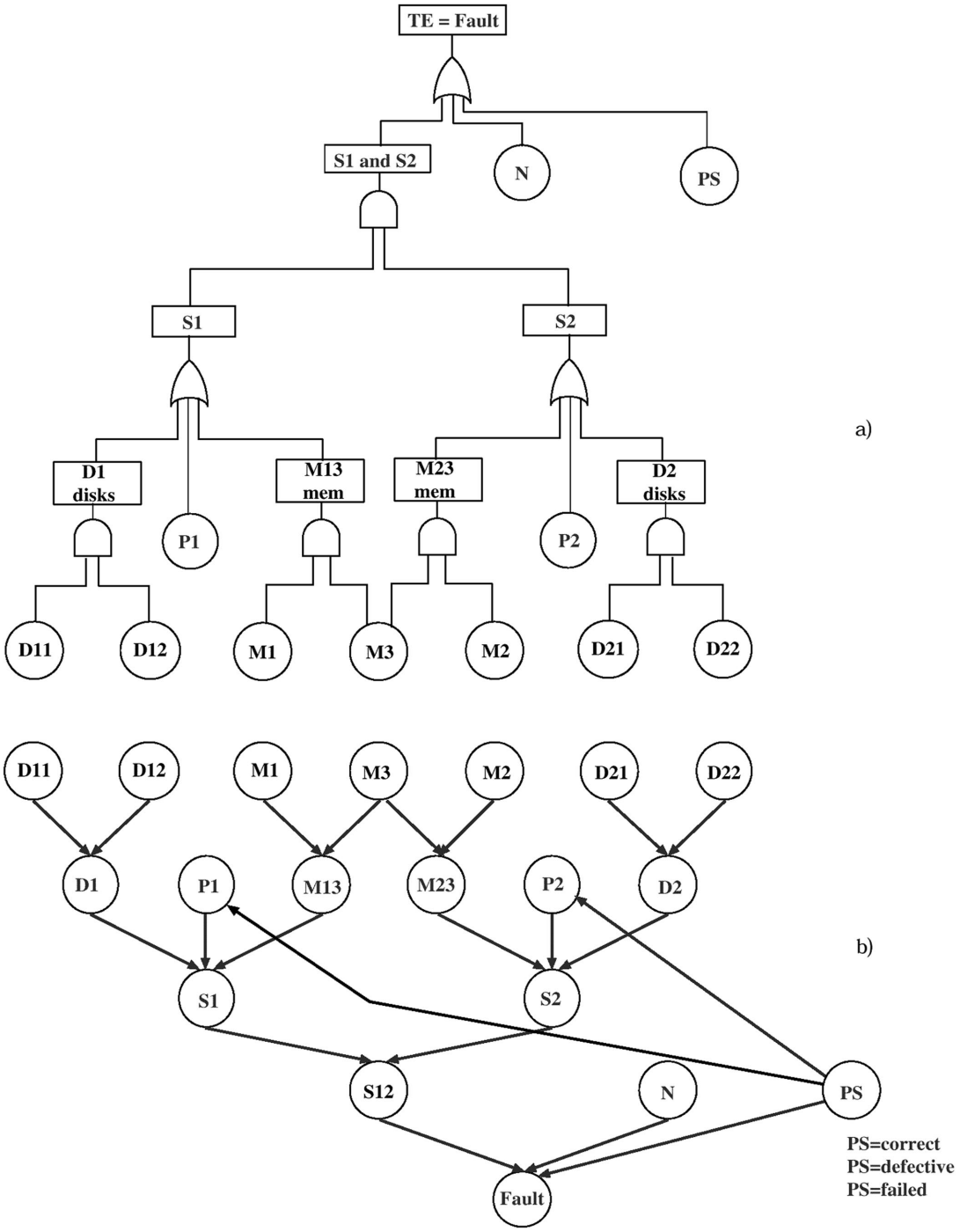

Fig. 6. Adding power supply: (a) fault tree; (b) Bayesian network.

that there exist well-established algorithms that can compute the marginal posterior probability of each node by considering this task as if it was a single query (i.e. it is not necessary to pose more queries of the type $P(Q|E)$, each time considering Q equal to the node for which we want the posterior distribution). Moreover, this kind of computation is very useful for determining the criticality of the components of the system, in case a fault is observed [4]. The main disadvantage is that considering only the marginal posterior probability of components is not always appropriate for a precise diagnosis [10]; in many cases the right way for characterizing diagnoses is to consider scenarios involving more variables (for example all the components). The other two kinds of computation address exactly this point.

Table 1
Component failure rate, prior and posterior probabilities

Component C	Fail. rate λ_C (f/h)	Prior fail. prob. Pr(C)	Posterior fail. prob. Pr(C|TE)
Disk D_{ij}	$\lambda_D = 8.0 \times 10^{-5}$	0.32968	0.98436
Proc P_i	$\lambda_P = 5.0 \times 10^{-7}$	0.00025	0.02252
Mem M_j	$\lambda_M = 3.0 \times 10^{-8}$	0.000015	0.000015
Bus N	$\lambda_N = 2.0 \times 10^{-9}$	0.00001	0.000081

5.1. Example

Consider the multiprocessor system of Fig. 3. The failure distribution of all components is assumed to be exponential with the failure rates (expressed in f/h units) given in Table 1. The dependability measures are required to be evaluated at a mission time $t = 5000$ h. The failure probabilities of each component, evaluated at $t = 5000$ h is reported in the second column of Table 1 (prior failure probabilities).

Forward propagation on the BN allows us to compute the unreliability of the TE as the a priori probability of system failure, i.e. $P(Fault) = 0.012313$.

If we observe that the system is faulty at time t, the marginal posterior fault probabilities of each component are then computed, and are reported on the third column of Table 1. This measures may be useful to provide an indication of the criticality of each single component if the system is faulty; observe that the severity ranking based on posteriors is different from the one based on priors. We can also notice that the most critical component is in this case each disk. However, this information is not completely significant from the diagnostic point of view; indeed, by considering the disk units, the only way of having the fault is to assume that all the disks D_{11}, D_{12}, D_{21}, D_{22} have failed at the same time (indeed, it is the only minimal cut-set involving only disks). This information (the fact that all disks have to be jointly considered faulty to get the fault) is not directly derived by marginal posteriors on components. In fact, for diagnostic purposes a more suitable analysis should consider the posterior joint probability of all the components given the system fault as evidence. This analysis corresponds to search the most probable state given the fault, over the state space represented by all the possible instantiations of the root variables (i.e. components).

In this case, we can determine by means of BN inference that the most probable diagnosis (i.e. the most probable state given the system fault) is exactly the one corresponding to the faulty value of all the disks and the normal value of all the other components; in particular, we obtain:

$$P(\bar{N}, \bar{M}_1, \bar{M}_2, \bar{M}_3, \bar{P}_1, \bar{P}_2, D_{11}, D_{12}, D_{21}, D_{22} | Fault) = 0.95422$$

Notice that the above diagnosis does not correspond to the cut-set $\{D_{11}, D_{12}, D_{21}, D_{22}\}$, since the latter does not imply that the unmentioned components are working (i.e. assigned to the normal value); anyway, the posterior probability of the cut-sets can be naturally computed in the BN setting by a query on all the disk variables conditioned on the observation of the fault.

In a reliability context, it seems more reasonable to restrict the attention only to root variables. However, an alternative way of characterizing diagnoses is to view them as complete assignments to all the variables but the evidence ones; this corresponds to search over the state space of all the possible instantiations of every variable in the BN. When the task consists in computing the most probable of such diagnoses it is called MPE computation [10] (where MPE stands for *most probable explanation*).

An interesting possibility offered by BN is that there exist algorithms able to produce diagnoses (either viewed as only-root assignments or all-variable assignments) in order of their probability of occurrence, without exploring the whole state space; they are usually called *any-time* algorithms, since the user can stop the algorithm at any time, by getting an approximate answer that is improved if more time is allocated to the algorithm. For example, an algorithm based on the model described in Ref. [19] is able to provide the most probable diagnoses, given the observation of the fault, in the multiprocessor system at any desired level of precision. By specifying a maximum admissible error ϵ in the posterior probability of the diagnoses, the algorithm is able to produce every diagnosis D, in decreasing order of their occurrence probability, with an estimate $P'(D|Fault)$ such that its actual posterior probability is $P(D|Fault) = P'(D|Fault) \wedge \epsilon$.

In the example, by requiring diagnoses to be root assignments and $\epsilon = 1 \times 10^{-6}$ the first three diagnoses are, in order:

$$d_1 : (\bar{N}, \bar{M}_1, \bar{M}_2, \bar{M}_3, \bar{P}_1, \bar{P}_2, D_{11}, D_{12}, D_{21}, D_{22})$$

$$d_2 : (\bar{N}, \bar{M}_1, \bar{M}_2, \bar{M}_3, \bar{P}_1, P_2, D_{11}, D_{12}, \bar{D}_{21}, \bar{D}_{22})$$

$$d_3 : (\bar{N}, \bar{M}_1, \bar{M}_2, \bar{M}_3, P_1, \bar{P}_2, \bar{D}_{11}, \bar{D}_{12}, D_{21}, D_{22})$$

The first one represents the already mentioned most probable diagnoses with all disks faulty, while the second and the third one are two symmetrical diagnoses: d_2 represents a fault caused by disk failures in the first sub-system and a processor failure in the second; d_3 represents a fault caused by disk failures in the second sub-system and a processor failure in the first.

Posterior probabilities are then computed within the

given error level as

$$P(d_1|Fault) = 0.954223$$

$$P(d_2|Fault) = P(d_3|Fault) = 98.87 \times 10^{-4}$$

The algorithm guarantees that any further diagnosis has a posterior probability smaller or equal than $P(d_2|Fault)$. It is worth noting that this result is in general obtained without exploring the whole state space, that in this case is equal to $2^{10} = 1024$ states, 10 being the number of components.

Similar results may be obtained for complete variable assignments. Notice that if the BN has only deterministic non-root nodes, a root assignment uniquely corresponds to a complete assignment with the same posterior, because given a particular assignment of modes to components, the assignment to non-root variables is deterministically obtained. This is no longer true if we introduce uncertainty at inner levels as it is usually done within BN.

Concerning the complexity of probabilistic computation using BN, the general problem of computing posterior probabilities on an arbitrary net is known to be NP-hard [20], however the problem reduces to a polynomial complexity if the structure of the net is such that the underlying undirected graph contains no cycles. Even if in practice this restriction is not often satisfied, research on efficient probabilistic computation has produced considerable results showing that acceptable computation can be performed even in networks with general structure and hundreds of nodes [21–23].

6. Conclusions and current research

Bayesian networks provide a robust probabilistic method of reasoning with uncertainty and are becoming widely used for dependability analysis of safety critical systems as the Programmable Electronic Systems (PES). During PES lifecycle, dependability analysis are performed at different levels addressing hardware, software and the whole system. Here, we have dealt with BN versus FT, a very popular technique for hardware dependability analysis. BN versus FT can address interesting questions allowing both forward and backward analysis; moreover, BN are more suitable to represent complex dependencies among components and to include uncertainty in modeling. Due to the presence of the software component, there is a major concern on how the overall PES dependability is evaluated. Software faults and human errors introduce design faults in PES. Probabilistic analysis of software dependability is a formidable task, for which no proven method is available. BN seems to be promising for sounder assessment of software dependability [24]. Overall system dependability, due the impact of design faults, is not well understood. The evaluation of overall system dependability can be obtained by considering and combining all the different sources of relevant information (evidence), including software and hardware dependability analysis [7]. The possibility of use of BN to support the evaluation of overall system dependability along the acceptance process of safety critical PES is under exploration. Although the use of BN seems to be promising at different levels of PES dependability analysis, they do not provide a direct mechanism for representing temporal dependencies, that are well implemented in popular techniques for dependability analysis, such as Markov Chains and Stochastic Petri Nets.

References

[1] Henley EJ, Kumamoto H. Reliability engineering and risk assessment. Englewood Cliffs, NJ: Prentice Hall, 1981.

[2] Leveson NG. Safeware: system safety and computers. Reading, MA: Addison-Wesley, 1995.

[3] Heckermann D, Wellman M, Mamdani A, editors. Real-world applications of Bayesian networks. Communications of the ACM 1995;38(3).

[4] Almond G. An extended example for testing Graphical Belief. Technical Report 6: Statistical Sciences Inc., 1992.

[5] Portinale L, Bobbio A. Bayesian networks for dependability analysis: an application to digital control reliability. Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, UAI-99, 1999. p. 551–8.

[6] Bobbio A, Portinale L, Minichino M, Ciancamerla E. Comparing fault trees and Bayesian networks for dependability analysis. Proceedings of the 18th International Conference on Computer Safety, Reliability and Security, SAFECOMP99, vol. 1698, 1999. p. 310–22.

[7] Fenton N, Littlewood B, Neil M, Strigini L, Sutcliffe A, Wright D. Assessing dependability of safety critical systems using diverse evidence. IEEE Proc Software Engng 1998;145(1):35–9.

[8] Torres-Toledano JG, Sucar LE. Bayesian networks for reliability analysis of complex systems, Lecture notes in artificial intelligence, vol. 1484. Berlin: Springer, 1998 (p. 195–206).

[9] Malhotra M, Trivedi K. Dependability modeling using Petri nets. IEEE Trans Reliabil 1995;R-44:428–40.

[10] Pearl J. Probabilistic reasoning in intelligent systems. Los Altos, CA: Morgan Kaufmann, 1989.

[11] Neapolitan RE. Probabilistic reasoning in expert systems. New York: Wiley, 1990.

[12] Dugan JB, Trivedi KS. Coverage modeling for dependability analysis of fault-tolerant systems. IEEE Trans Comput 1989;38:775–87.

[13] Garribba S, Guagnini E, Mussio P. Multiple-valued logic trees: meaning and prime implicants. IEEE Trans Reliabil 1985;R-34:463–72.

[14] Doyle SA, Dugan JB, Patterson-Hine A. A combinatorial approach to modeling imperfect coverage. IEEE Trans Reliabil 1995;44:87–94.

[15] Wood AP. Multistate block diagrams and fault trees. IEEE Trans Reliabil 1985;R-34:236–40.

[16] Knowledge Industries. DXpress 2.0, 1996.

[17] Microsoft Corporation. Microsoft Belief Network Tools.

[18] Andersen SK, Olesen KG, Jensen FV. HUGIN — a shell for building Bayesian belief universes for expert systems. Proceedings of the 11th IJCAI, Detroit, MI, 1989. p. 1080–5.

[19] Portinale L, Torasso P. A comparative analysis of Horn models and Bayesian networks for diagnosis, Proceedings of the 5th Italian Conference on Artificial Intelligence. Berlin: Springer, 1997.

[20] Cooper G. The computation complexity of probabilistic inference using Bayesian belief networks. Artific. Intell 1990;33:393–405.

[21] Kjaerulff U. Aspects of efficiency improvements in Bayesian

networks. Technical Report. Thesis, Faculty of Technology and Science, Aalborg University, 1993.
[22] Poole N, Zhang L. Exploiting causal independence in Bayesian network inference. J Artific Intell Res 1996;5:301–28.
[23] D'Ambrosio M, Takinawa. Multiplicative factorization of noisy-max. Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence UAI99, Stockholm, 1999. p. 622–30.
[24] Delic KA, Mazzanti F, Strigini L. Formalising engineering judgement on software dependability via Belief Networks. Technical Report, SHIP, 1995.