# SULL'USO DEGLI ALBERI DI GUASTO NON COERENTI NELL'ANALISI DEI SISTEMI

**Contini S.[1], Cojazzi G.G.M.[1], G. de Cola[2]**
**1. Commissione Europea, Centro Comune Ricerca, IPSC, Ispra, VA, 21020, Italia**
**2. INFOCON, Corso Mazzini, 31/2, Lavagna, GE, 16033, Italia**

## SOMMARIO

L'uso della metodologia dell'albero dei guasti nell'analisi di sicurezza e rischio di installazioni potenzialmente pericolose e' molto diffusa grazie alla notevole sistematicità che essa offre nella descrizione delle cause primarie di guasto del sistema. La quantificazione di un albero dei guasti consente di individuare le parti più deboli sulle quali operare per il miglioramento del progetto. Quando le relazioni logiche fra eventi e rispettive cause sono espresse per mezzo degli operatori logici AND e OR e tutti gli eventi di guasto dei componenti sono rappresentati con variabili binarie in forma positiva, gli alberi sono denominati *coerenti*. La recente diffusione dell'approccio dei Diagrammi di Decisione Binari per l'analisi di funzioni logiche ha consentito di ampliare le applicazioni dell'albero dei guasti grazie alla possibilità di analizzare esattamente anche espressioni logiche contenenti l'operatore NOT per la descrizione di stati di funzionamento. Tali alberi sono denominati *non coerenti*. Il presente lavoro ha l'obiettivo di mostrare i vantaggi derivanti dall'uso di alberi non coerenti per la modellizzazione e l'analisi di stati di guasto complessi, quali ad esempio gli stati di guasto intermedi di un sistema, le procedure di manutenzione e top event condizionati allo stato di funzionamento di uno o più sottosistemi.

## 1.0 INTRODUZIONE

L'Albero dei Guasti (AdG) e' lo strumento più diffuso per la modellizzazione e la quantificazione dei possibili stati di guasto pericolosi di un sistema, denominati Top Events. Per ciascun Top event viene costruito e analizzato il corrispondente albero dei guasti, i cui risultati consentono di individuare i punti deboli del sistema sui quali devono essere allocate le risorse per il miglioramento della sicurezza.

Un albero dei guasti si definisce Coerente se può essere ricondotto ad una funzione booleana contenente solo gli operatori AND-OR. L'analisi probabilistica tipica degli alberi di guasto coerenti e' basata sul calcolo delle combinazioni minime di eventi (Minimal Cut Sets, MCS) il cui verificarsi implica il verificarsi del Top event e sull'applicazione della Kinetic Tree Theory (KITT) [1] all'insieme degli MCS. A causa dei proibitivi tempi di calcolo le grandezze affidabilistiche a livello del Top-Event possono essere calcolate solo introducendo alcune approssimazioni, accettabili per applicazioni nel campo della sicurezza e della valutazione del rischio.

Un albero dei guasti si definisce Non Coerente se almeno una delle combinazioni minime in grado di verificare il Top event contiene degli eventi negati. Un evento negato indica una condizione di funzionamento (logica AND-OR-NOT). Dal punto di vista della modellizzazione questi alberi offrono la possibilità di considerare eventi mutuamente esclusivi. L'analisi di una logica AND-OR-NOT e' più complessa dell'analisi di una logica AND-OR. Il concetto di MCS e' sostituito da quello di Implicante Primo (IP), la cui determinazione e' molto più laboriosa; anche l'interpretazione dei risultati e' più complessa, il che ha portato alla definizione aggiuntiva dei P-Cuts [2]. Un P-Cut non è altro che un MCS formato dagli eventi non negati contenuti nell'implicante primo. La probabilità di guasto di un P-Cut e' maggiore di quella di un Implicante primo. L'approssimazione e' accettabile poiché la probabilità di un evento negato e' prossima a 1.

Un esempio comune di applicazione della logica AND-OR-NOT e' l'Albero degli Eventi (AdE), le cui sequenze si analizzano applicando il metodo denominato Fault Tree Linking, ossia costruendo un albero dei guasti per ogni sequenza incidentale e calcolando di fatto i P-Cuts. Sequenze contenenti

contemporaneamente eventi allo stato affermato e negato vengono cancellate, l'analisi probabilistica si esegue applicando la KITT all'insieme dei P-Cuts considerando di fatto le probabilità degli eventi negati pari ad uno. A parte questa applicazione le logiche non coerenti sono state finora raramente utilizzate.

Nell'ultimo decennio l'approccio dei diagrammi di decisione binari (Binary Decision Diagrams, BDD) [3] e' stato applicato con successo anche agli alberi di guasto [4]. Secondo l'approccio BDD, l'albero viene memorizzato sotto forma di grafo aciclico orientato. Nell'ipotesi di indipendenza statistica degli eventi primari gli algoritmi di analisi probabilistica consentono di calcolare esattamente tutte le grandezze affidabilistiche di interesse ed il loro calcolo non richiede la determinazione degli MCS. Infatti, il grafo, opportunamente elaborato, contiene tutti i modi di guasto disgiunti del sistema, il che consente di ottenere valori esatti di indisponibilità, frequenze incondizionate di guasto e di riparazione per il Top Event. Da queste grandezze si possono calcolare il numero atteso di guasti e di riparazioni, il tempo medio tra due guasti (MTBF), il tempo medio alla riparazione (MTTR) ed il tempo medio al guasto (MTTF), oltre ai valori esatti degli indici di importanza dei componenti.

Applicando l'approccio BDD gli alberi non coerenti possono essere analizzati con lo stesso grado di efficienza dei coerenti. Questo fatto ha portato a riconsiderare l'uso degli alberi non coerenti per l'analisi dei sistemi. Nasce allora la domanda: in quali casi la logica non coerente e' utile e necessaria nelle applicazioni pratiche? Lo scopo principale della presente memoria e' di fornire alcune risposte concrete a tale domanda attraverso la descrizione dei seguenti esempi, tratti da problemi reali, che rappresentano una casistica abbastanza completa di tipologie di alberi non coerenti.

- Data una rete elettrica di alimentazione di un impianto si richiede di determinare non solo la probabilità di non funzionamento di tutti gli utilizzatori, ma anche di quantificare le probabilità di guasto di uno o più utilizzatori e del contemporaneo funzionamento dei rimanenti. In questi casi l'uso di logiche non coerenti facilita enormemente la costruzione degli alberi di guasto.
- Modellizzazione corretta delle procedure di test e manutenzione applicate ad esempio in campo nucleare.
- Determinazione di probabilità condizionate.

In tutti questi casi l' uso di logiche non coerenti facilita enormemente la costruzione degli alberi per i Top events di interesse. Altri esempi di alberi non coerenti sono descritti in [5]. Inoltre nel nuovo campo della security nasce spesso la necessità di trattare eventi mutualmente esclusivi, descrittivi delle possibili alternative di attacco. La descrizione dei possibili metodi di analisi è riportata in [6], alcuni dei quali si basano su una modellizzazione non coerente. Tutti questi problemi si possono agevolmente risolvere solo con metodi in grado di analizzare correttamente le funzioni non coerenti, sia dal punto di vista logico che probabilistico. Inoltre, per il calcolo delle probabilità condizionate e' necessario poter determinare il valore esatto della probabilità dei diversi termini dell'espressione del Top event. L'approccio BDD per l'analisi di analisi degli alberi di guasto non coerenti e' la risposta a queste esigenze.

La presente memoria è organizzata come segue. Nel prossimo capitolo descriviamo formalmente, e mediante esempi, i concetti di coerenza e non coerenza seguiti, nel terzo capitolo, da alcuni esempi di analisi di indisponibilità mediante alberi non coerenti. La quantificazione di questi alberi è riportata nel capitolo quarto preceduta da una breve descrizione delle caratteristiche principali del programma ASTRA 3.0, recentemente sviluppato presso il CCR-Ispra e interamente basato sull'approccio BDD per l'analisi esatta di alberi di guasto coerenti e non coerenti.

## 2.0 COERENZA E NON COERENZA

Un Albero dei Guasti rappresenta graficamente le relazioni logiche fra il guasto dei componenti (eventi primari) ed il guasto del sistema (Top event). Un evento primario e' una proposizione che può descrivere anche un errore umano o un evento esterno. Il Top event e'

una proposizione che descrive il guasto del sistema. A ciascun evento primario si associa una variabile binaria che assume il valore 1 se la proposizione e' vera, 0 se falsa. Conseguentemente anche il Top event può assumere i due soli valori 1, 0. L'analisi richiede innanzitutto la rappresentazione dell'albero sotto forma di funzione logica binaria.

Sia $\Phi(\mathbf{x})$ la funzione booleana del Top event del sistema di *n* variabili binarie rappresentate per mezzo del vettore $\mathbf{x} = [x_1, x_2, \dots, x_n]$. In funzione dei valori delle variabili binarie, avremo $\Phi(\mathbf{x}) = 1$, Top verificato o $\Phi(\mathbf{x}) = 0$, Top non verificato.

Una funzione $\Phi(\mathbf{x})$ si definisce *monotona (non decrescente) o coerente* se e solo se:

a) $\Phi(1_i,\mathbf{x}) \geq \Phi(0_i,\mathbf{x})$, dove $\Phi(1_i,\mathbf{x}) = \Phi(x_1, \dots, x_{i-1}, x_i=1, x_{i+1},\dots, x_n)$ e $\Phi(0_i,\mathbf{x}) = \Phi(x_1, \dots, x_{i-1}, x_i=0, x_{i+1},\dots, x_n)$ per tutte le variabili. In pratica questo significa che lo stato del sistema con l'i-esimo componente guasto non può essere migliore dello stato del sistema con l'i-esimo componente funzionante, per tutte le possibili combinazioni di valori delle variabili di stato degli altri componenti.
   Ne consegue che:
   - $\Phi(\mathbf{1}) = 1$, ossia se tutti i componenti sono guasti $\mathbf{1} = [x_1=1, x_2=1, \dots, x_n=1]$, il sistema è guasto;
   - $\Phi(\mathbf{0}) = 0$, se tutti i componenti sono funzionanti $\mathbf{0} = [x_1=0, x_2=0, \dots, x_n=0]$, il sistema è funzionante.

b) $\Phi(x_1, x_2, \dots, x_{i-1}, x_i=1, x_{i+1},\dots, x_n) \neq \Phi(x_1, x_2, \dots, x_{i-1}, x_i=0, x_{i+1},\dots, x_n)$ per almeno una combinazione di stato delle variabili. In pratica questa condizione significa che tutti i componenti sono rilevanti, vale a dire che non ci possono essere componenti il cui stato non ha mai effetto sullo stato del Top event. In altre parole deve esistere almeno una situazione in cui il sistema funziona se il componente funziona e si guasta se il componente si guasta, ossia: $\Phi(1_i,\mathbf{x^*}) - \Phi(0_i,\mathbf{x^*}) = 1$. In pratica questa condizione richiede che tutti i componenti svolgano uno specifico compito nel sistema. Se un componente non è rilevante, allora $\Phi(1_i,\mathbf{x}) - \Phi(0_i,\mathbf{x}) = 0$ per qualsiasi combinazione di valori delle variabili di stato degli altri componenti. Questa condizione è indicativa di un errore di progetto o più probabilmente di un errore nella costruzione dell'albero dei guasti.

Il soddisfacimento delle condizioni a) e b) [7] consente di applicare l'analisi probabilistica dei sistemi coerenti.

Per meglio chiarire i concetti di coerenza e non coerenza consideriamo la seguente funzione logica $\Phi = a \wedge (b \vee c)$, dove *a, b,* e *c* rappresentano le variabili che definiscono lo stato dei relativi componenti, assunti per semplicità non riparabili. Il grafo in Figura 1 rappresenta gli 8 possibili stati nei quali può evolvere il sistema (3 variabili binarie, $2^3 = 8$ stati). Il sistema evolve dallo stato iniziale *a*= 0, *b*= 0, *c*= 0, indicato con $\Phi(000)=0$, allo stato finale $\Phi(111)=1$ al guastarsi dei componenti, nell'ipotesi che il passaggio da uno stato all'altro avvenga a seguito del guasto di un solo componente.

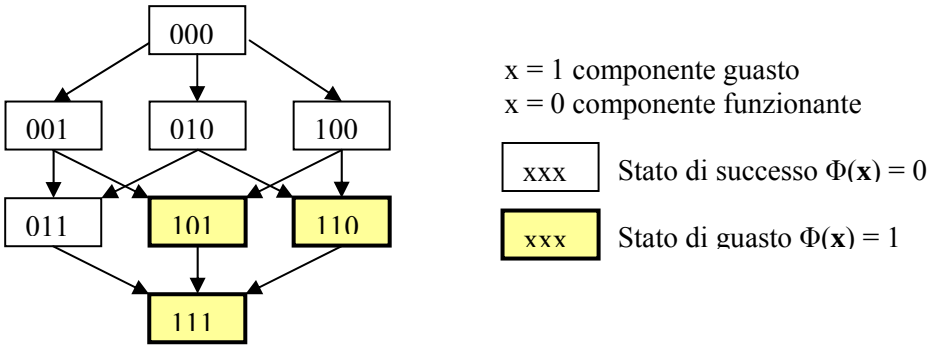


Figura 1. Grafo degli stati della funzione coerente $\Phi = a \wedge (b \vee c)$.

Possiamo notare che il sistema, trovandosi in uno stato di funzionamento, ad esempio Φ(010)=0, cambia stato portandosi in un altro stato di funzionamento Φ(011)=0 o in uno stato di guasto Φ(110)=1 a causa rispettivamente del guasto di *c* o di *a*. Una volta entrato in uno stato di guasto può solo passare, a seguito del guasto di un componente, in un altro stato di guasto.

Quando queste condizioni non sono verificate, la funzione non è più monotona e viene solitamente definita come non coerente.

Vediamo meglio il significato di "*non coerenza*". Consideriamo la funzione $\Phi = a \wedge (b \vee \bar{c})$ dove in questo caso la variabile *c* è negata. Il grafo degli stati di questa funzione è riportato in Figura 2. In questo caso possiamo notare che dallo stato di guasto Φ(100)=1 il sistema può portarsi nello stato di guasto Φ(110)=1 a seguito del guasto di *b* (*b*=1) o nello stato di funzionamento Φ(101)=0 al verificarsi del guasto di *c* ($c = 1$, ossia $\bar{c} = 0$). Il comportamento non coerente si ha, quando il guasto di un componente (*c*=1) implica il passaggio del sistema dallo stato di guasto (Φ = 1) allo stato di successo (Φ = 0).

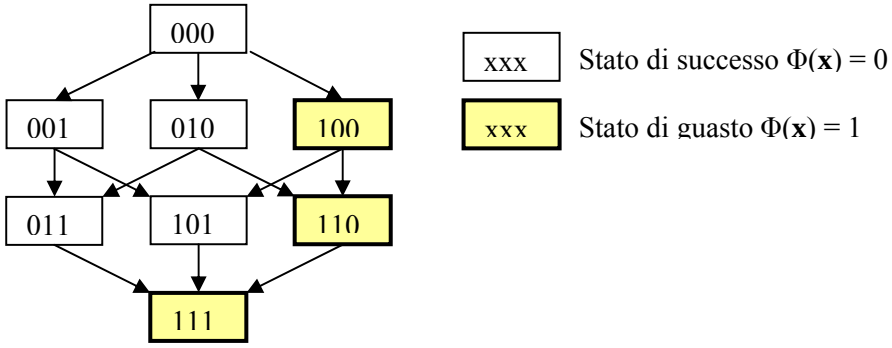


Figura 2. Grafo degli stati della funzione non coerente $\Phi = a \wedge (b \vee \bar{c})$.

E' chiaro che situazioni di questo tipo possono apparire non realistiche, mentre in realtà sono molto frequenti.

Consideriamo, in Figura 3, il serbatoio S contenente la sostanza xy. S e' posto all'interno di un bacino di contenimento.

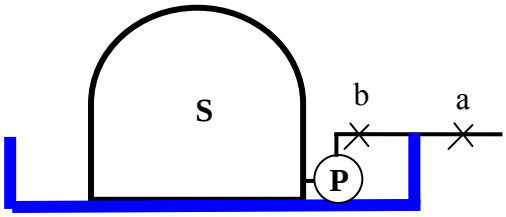


Figura 3. Semplice esempio di sistema con albero non coerente.

Consideriamo i seguenti Top events.

- Top.a: *Rilascio della sostanza xy all'esterno del bacino di contenimento*
- Top.b*: Rilascio della sostanza xy all'interno del bacino di contenimento*

e gli eventi

A = rottura della tubazione nel punto a.

B = rottura della tubazione nel punto b.

Considerando per semplicità solamente questi due eventi, le funzioni logiche dei due Top events sono rispettivamente:

Top.a = $\overline{B} \wedge A$

Top.b = $B$

Top.a e' espresso da una funzione non coerente poiché il verificarsi di B (B=1) rende Top.a=0 (non verificato). Questo non significa che il sistema sia migliorato ma semplicemente che non sussistono più le condizioni che verificano Top.a. Al verificarsi di B si verifica invece Top.b. Infatti i due top events sono mutuamente esclusivi, ossia Top.a $\wedge$ Top.b = $\overline{B} \wedge A \wedge B = 0$.

Esempi di questo tipo sono numerosi. Bedford e Cooke [8] descrivono un sistema di alimentazione elettrica nel quale il guasto di due linee porta il sistema in una situazione di funzionamento critico (linea rimanente sovraccaricata); il guasto di un altro componente riporta invece il sistema in una situazione non pericolosa, anche se il sistema ha subito tre guasti e risulta degradato.

In pratica durante la costruzione degli alberi di guasto non si considerano gli eventi negati, relativi al funzionamento corretto dei componenti, ottenendo quindi alberi "coerenti". Per l'esempio in Figura 3 avremmo: Top.a = A e Top.b = B, e di conseguenza i due Top events diventerebbero eventi indipendenti, vale a dire Top.a $\wedge$ Top.b = $A \wedge B \neq 0$.

Questa approssimazione e' accettabile solo se:

- non esistono eventi mutuamente esclusivi;
- le probabilità di funzionamento sono molto prossime a 1.

Al di fuori di questi casi occorre far uso di funzioni non monotone (non coerenti).

Gli alberi non coerenti possono sembrare un'inutile complicazione. In realta' la disponibilità di un programma di calcolo in grado di analizzare esattamente gli alberi contenenti eventi negati consente di eseguire applicazioni interessanti, alcune delle quali sono descritte nel prossimo capitolo.

## 3.0 ALCUNE APPLICAZIONI DEGLI ALBERI NON COERENTI

### 3.1 Top events di stati intermedi

Le funzioni non coerenti sono molto utili per modellizzare rapidamente diversi stati di guasto parziale del sistema ai quali, ad esempio, possono corrispondere diverse conseguenze, quali danni o necessità di applicare diverse procedure di intervento.

Dato un sistema formato da n componenti, ciascuno descritto da una variabile binaria, il numero totale di stati del sistema e' $N = 2^n$. Escludendo i due stati in cui i componenti sono tutti funzionanti o tutti guasti, gli altri stati nei quali può evolvere il sistema sono stati intermedi (alcuni componenti guasti, altri funzionanti). Se a tali stati intermedi sono associate diverse conseguenze allora diventa importante la loro analisi.

Consideriamo in Figura 4 lo schema semplificato di una rete elettrica (relativa all' alimentazione ai circuiti di raffreddamento del reattore PEC [9]) ed in particolare i quattro carichi: P1 ="Pompa primaria 1", P2 = "Pompa primaria 2", S1 = "Pompa secondaria 1" e S2 = "Pompa secondaria 2" alimentati dalle linee A, B attraverso le barre BT1, BT2, B1, B2, B3 e B4. Supponiamo di essere interessati alla determinazione dell'indisponibilità' dei seguenti stati:

1. Guasto di tutti i carichi.
2. Guasto di uno o più carichi;

e dell'indisponibilità' dei seguenti stati intermedi, a ciascuno dei quali corrisponde l'applicazione di particolari procedure da parte degli operatori:

3. Guasto di una sola pompa primaria con entrambe le pompe secondarie funzionanti;
4. Guasto di una sola pompa secondaria con entrambe le pompe primarie funzionanti;
5. Guasto di P1 e S2 con P2 e S1 funzionanti;
6. Guasto di P2 e S1 con P1 e S2 funzionanti.

Supponiamo dapprima di avere la possibilità di analizzare solo alberi coerenti. Per ciascun top event occorrerebbe costruire un albero diverso. Il problema e' semplice per i primi due Top, in quanto gli alberi risultanti sono coerenti. Per tutti gli altri e' invece necessario derivare dalla definizione del Top event le opportune "condizioni al contorno", ossia condizioni che non possono essere violate durante la costruzione degli alberi. Consideriamo ad esempio il sesto Top event: "Guasto di P2 e S1 con P1 e S2 funzionanti". Affinché P1 e S2 siano funzionanti occorre che le barre BT1 e B3 o B4 siano alimentate. Pertanto la costruzione dell'albero deve essere tale da rispettare queste tre condizioni al contorno.

In generale la costruzione di alberi con condizioni al contorno non e' semplice da eseguire e richiede molta attenzione per evitare di commettere errori. La possibilità invece di analizzare alberi non coerenti ne semplifica drasticamente la costruzione. Infatti, e' sufficiente costruire i quattro alberi per i top events concernenti il guasto di ciascun carico, senza considerare alcuna condizione al contorno, e comporli opportunamente in base all'espressione logica del Top event da analizzare. Per esempio, l'espressione logica relativa al sesto Top event e' la seguente: TOP6 = P2 $\wedge$ S1 $\wedge \neg$ P2 $\wedge \neg$ S2 dove $\wedge$ rappresenta l'operatore AND e $\neg$ l'operatore NOT. Altro esempio: TOP3 = (P1 $\oplus$ P2) $\wedge \neg$ S1 $\wedge \neg$ S2 in cui $\oplus$ rappresenta l'OR esclusivo che richiede l' uso dell' operatore NOT.

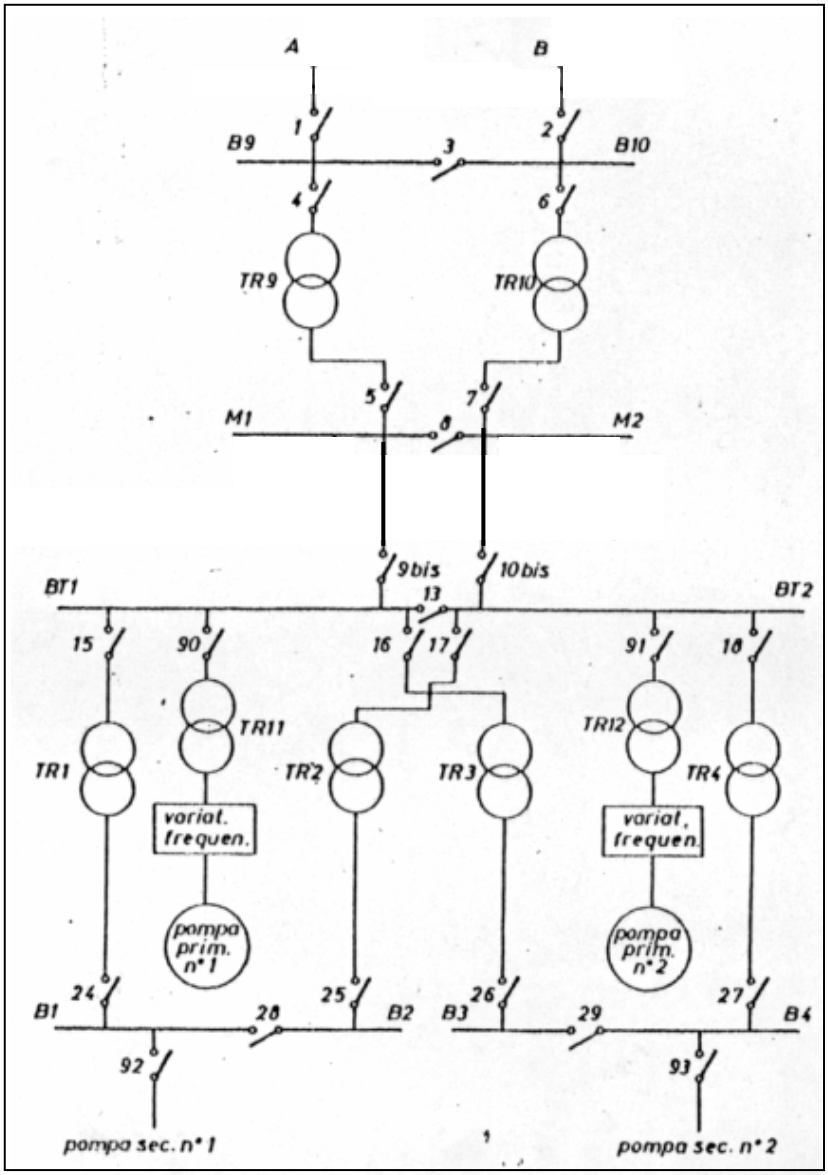


Figure 4. Schema funzionale semplificato di una rete elettrica

La Tabella 1 contiene la definizione dei sei top events sopra elencati e la corrispondente espressione logica.

Tabella 1. Top events e corrispondente espressione logica per l'analisi della rete di Figura 4.

| N | Descrizione Top event | Espressione logica |
|---|---|---|
| 1 | Guasto di tutti i carichi | Top = P1 ∧ P2 ∧ S1 ∧ S2 |
| 2 | Guasto di uno o piu' carichi | Top = P1 ∨ P2 ∨ S1 ∨ S2 |
| 3 | Guasto di una sola pompa primaria con entrambe le pompe secondarie funzionanti | Top = (P1 ⊕ P2) ∧¬(S1∨ S2) |
| 4 | Guasto di una sola pompa secondaria con entrambe le pompe primarie funzionanti | Top = (S1 ⊕ S2) ∧¬(P1∨ P2) |
| 5 | Guasto di P1 e S2 con P2 e S1 funzionanti | Top = (P1 ∧ S2) ∧¬(P2 ∨ S1) |
| 6 | Guasto di P2 e S1 con P1 e S2 funzionanti | Top = (P2 ∧ S1) ∧¬(P1∨ S2) |

### 3.2 Probabilità condizionate

Un'altra applicazione interessante degli alberi non coerenti riguarda il calcolo della probabilità del Top event condizionata al funzionamento di uno o più sottosistemi.

Ad esempio, con riferimento alla rete elettrica di Figura 3, potremmo essere interessati al calcolo della probabilità dell'evento seguente:

Top = "Guasto di tutti i carichi condizionato alla disponibilità di una sola barra BT di alimentazione"

E' necessario applicare il teorema del calcolo delle probabilità condizionate:

$$Pr\{A \mid B\} = Pr\{A \wedge B\} / Pr\{B\}$$

Nel nostro caso A = P1∧P2∧S1∧S2 rappresenta il guasto di tutti i carichi e B = BT1 ⊕ BT2 il guasto di una sola barra. L'espressione logica corrispondente e' pertanto la seguente:

$$Pr\{(P1 \wedge P2 \wedge S1 \wedge S2) \mid (BT1 \oplus BT2)\} = Pr\{P1 \wedge P2 \wedge S1 \wedge S2 \wedge (BT1 \oplus BT2)\} / Pr\{BT1 \oplus BT2\} \quad (1)$$

Il grado di difficoltà nella costruzione dell'albero dei guasti coerente per questo Top e' facilmente intuibile.

### 3.3 Procedure di manutenzione

Consideriamo il sistema di sicurezza in stato di attesa, schematizzato in Figura 5a, formato da due sottosistemi identici in parallelo T1 e T2 Supponiamo che la procedura di manutenzione sia tale per cui quando un sottosistema e' soggetto a test l'altro deve essere necessariamente in posizione on line allo scopo di garantire sempre protezione all'impianto. Tale esempio ricorre ripetutamente in letteratura, a tale scopo si veda ad esempio la [10]. L'albero dei guasti per l'evento "Indisponibilità su domanda del sistema di sicurezza" e' riportato in Figura 5b. L'albero contiene gli eventi mutuamente esclusivi B e D. L'espressione della indisponibilità di T1 e T2 in caso di applicazione corretta della procedura di test, parte destra dell' albero, è la seguente:

$$TOP = (A \vee B \wedge \neg D) \wedge (C \vee \neg B \wedge D) \quad (2)$$

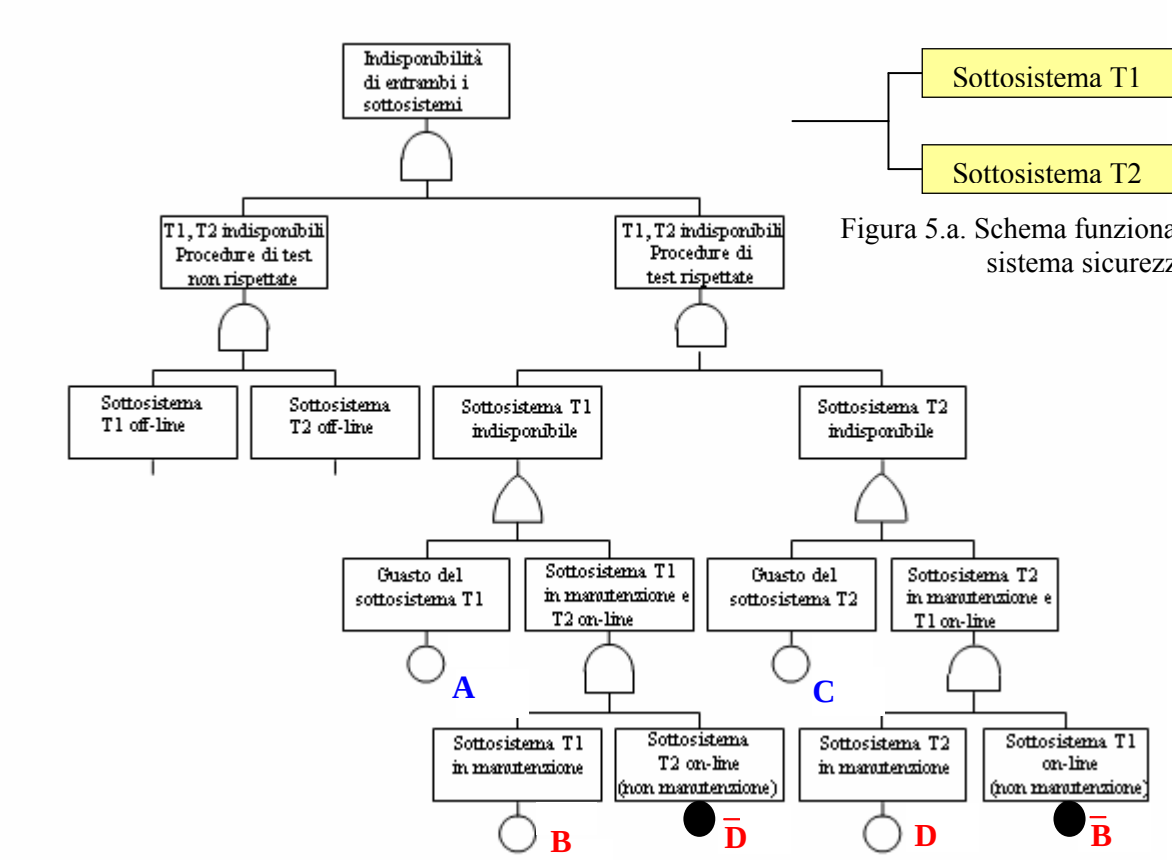





Figura 5.a. Schema funzionale sistema sicurezza

Figura 5b. Albero dei guasti per la procedura di manutenzione di un sistema di protezione. La parte sinistra dell' albero non e' rilevante e' quindi non e' stata riportata. Gli eventi base negato sono indicato con il pallino pieno e con la sovrasegnatura.

## 4.0 LA NUOVA VERSIONE DEL SOFTWARE ASTRA

ASTRA e' il software sviluppato presso il CCR Ispra per l'analisi di alberi di guasto, coerenti e non coerenti, su Personal Computers sotto diverse versioni del sistema operativo Windows.

Gli algoritmi per l'analisi logica e probabilistica si basano interamente sull'approccio BDD (*Binary Decision Diagram*) notevolmente più efficiente di altri approcci proposti in passato [11] [12].

Rispetto ai precedenti approcci basati sul calcolo degli insiemi minimi di guasto (MCS), con l'approccio BDD e' possibile ottenere i valori esatti dell'indisponibilità', delle frequenze incondizionate di guasto e riparazione a livello di sistema, oltre agli indici di importanza più diffusi, senza passare attraverso il calcolo degli MCS.
Infatti, l'albero viene memorizzato sotto forma di grafo aciclico orientato contenente tutti i modi di guasto disgiunti sul quale viene eseguita l'analisi probabilistica. In seguito si ottiene un nuovo BDD contenente tutti gli MCS dal quale si estraggono i più significativi rispetto a valori di cut-off definiti dall'utente.

Inoltre la rappresentazione in termini di BDD consente, in linea di principio di verificare :

- l'equivalenza logica fra due alberi;
- l'individuazione della presenza nell'albero di eventi non rilevanti.

Per quanto riguarda il primo problema si può dimostrare che due alberi sono logicamente equivalenti se, utilizzando lo stesso ordinamento delle variabili [11], i rispettivi BDD sono uguali. In precedenza, per questo tipo di verifica, era necesario controllare l'uguaglianza degli MCS.
La procedura di costruzione di un BDD è tale da eliminare automaticamente eventuali eventi non rilevanti. Infatti con l'espansione di $\Phi(\mathbf{x})$ rispetto a una variabile $x_i$ associata a un evento non rilevante si ha $\Phi(\mathbf{x}) = x_i\ \Phi(1_i i,\mathbf{x}) + \overline{x}_i\ \Phi(0_i,\mathbf{x})$. La non rilevanza di $x_i$ significa che $\Phi(1_i,\mathbf{x}) = \Phi(0_i,\mathbf{x})$ e quindi $\Phi(\mathbf{x}) = \Phi(1_i,\mathbf{x})$, ossia la variabile $x_i$ viene eliminata dal BDD. Pertanto, il confrono fra la lista degli eventi dell'albero dei guasti e quella del BDD consente di verificare l'eventuale presenza di eventi non rilevanti. Questa informazione è indubbiamente molto utile per l'analista il quale potrà facilmente eseguire le opportune verifiche per la rimozione della condizione di non rilevanza.

La procedura di calcolo implementata in ASTRA per l'analisi degli alberi non coerenti, e riassunta in [13], è originale e si basa sulla classificazione delle variabili e sulla loro etichettatura allo scopo di applicare in modo più efficiente gli algoritmi di analisi. Infatti la complessità di tali algoritmi dipende dal tipo di variabile.

In questa sede tuttavia ci limitiamo a elencare solamente le caratteristiche principali di ASTRA, senza entrare nei dettagli della procedura di calcolo descritta in dettaglio in [14].

Le caratteristiche principali di ASTRA 3.0 sono le seguenti:

- capacità di analizzare alberi coerenti e non coerenti (operatori logici consentiti: AND, OR, NOT, XOR, INH, K/N);
- calcolo dei valori esatti dell'indisponibilità, della frequenza di guasto e di riparazione, del numero atteso di guasti e riparazione per il *Top event* e per tutti i modi di guasto minimi (*Minimal Cut Sets*); a richiesta, queste grandezze possono essere ottenute in funzione del tempo con la possibilità di simulare correttamente diverse politiche di test / manutenzione;;
- calcolo dei valori esatti di diversi indici di importanza dei componenti (*Structural Importance*, *Birnbaum, Risk Achievement Worth, Risk Reduction Worth, e Criticality);* a richiesta, queste grandezze possono essere ottenute in funzione del tempo;
- uso della tecnica di cut-off per la determinazione degli MCS significativi con calcolo dell'errore di troncamento;
- uso di condizioni al contorno applicate agli eventi primari;
- editor grafico con associata banca dati di affidabilità dei componenti;
- analisi di sensibilità interattiva eseguita contemporaneamente su tutti gli alberi dell'impianto;
- rappresentazione grafica di tutte le grandezze probabilistiche calcolate in funzione del tempo;
- rappresentazione grafica dell'albero in formato A3 e A4;
- generazione in formato Word della documentazione con i risultati dell'analisi;
- Help in linea;

Dal punto di vista della facilità d'uso ASTRA offre un insieme di semplici comandi per la gestione degli alberi, la costruzione grafica degli stessi e la produzione della documentazione, compresa la rappresentazione grafica dei risultati.

Il software e' interamente scritto in C++ e lavora su piattaforma Windows.

Nel prosieguo del capitolo viene illustrata l'applicazione di ASTRA all'analisi dei diversi Top events della rete elettrica di cui alla tabella 1 e al calcolo della probabilità condizionata espressa dalla (1).

### 4.1 Analisi della rete elettrica

Nella Tabella 2 sono riportati i risultati dell'analisi dei Top events della rete elettrica precedentemente considerata.

Tabella 2. Risultati dell'analisi degli alberi di cui alla tabella 1

| N | Espressione probabilistica del top event | N. MCS | Pr{TOP} | Tempo (sec) |
|---|---|---|---|---|
| 1 | Top = Pr{P1∧P2∧S1∧S2} | 112460 | 3.3252E-05 | 0,172 |
| 2 | Top = Pr{P1∨ P2∨ S1∨S2} | 525 | 7.4018E-03 | 0,203 |
| 3 | Top = Pr{(P1⊕P2) ∧¬(S1∨S2)} | 114 | 4.2076E-03 | 0,188 |
| 4 | Top = Pr{(S1⊕S2) ∧¬(P1∨P2)} | 49 | 2.1038E-03 | 0,188 |
| 5 | Top = Pr{(P1∧S2) ∧¬(P2∨S1)} | 830 | 3.3931E-06 | 0,187 |
| 6 | Top = Pr{(P2∧S1) ∧¬(P1∨S2)} | 670 | 3.3931E-06 | 0,188 |

A partire dagli alberi di guasto associati ai 4 eventi TOP P1, P2, S1 ed S2 e' possibile costruire facilmente tutti i sei Top events della tabella 2 ed i relativi alberi di guasto. Tali alberi differiscono fra loro solo per l'espressione logica del Top event. Di conseguenza essi hanno tutti lo stesso numero di eventi (99) e gates (65). Il tempo di calcolo riportato nell'ultima colonna si riferisce al calcolo della indisponibilità al tempo di missione di 10.000 ore e comprende il tempo necessario per il calcolo del grafo contenente tutti i modi di guasto minimi il cui numero e' riportato nella terza colonna. Opzionalmente, l'uso della tecnica di cut off consente poi di ricavare i modi di guasto più significativi. Si noti che la probabilità dell' evento Top non e' in alcun modo basata sulla probabilità degli MCS e corrisponde alla probabilità che si otterrebbe considerando tutti gli implicanti primi dell' albero non coerente

Come si può notare dal contenuto della tabella, i tempi di calcolo sono estremamente limitati. L'evento più probabile è ovviamente Top2 perchè si riferisce al guasto di uno o più utilizzatori. I meno probabili sono i Top 5 e 6 relativi al guasto di due soli utilizzatori.
Gli alberi sono stati elaborati su PC Fujitzu-Siemens 600 con processore Xeon(TM) operante a 2,40 GHz e con memoria di 1,25 GB of RAM.

### 4.2 Calcolo di probabilità condizionate

Consideriamo di nuovo l'esempio precedente sulla modellizzazione delle probabilità del Top event condizionata allo stato di funzionamento di uno o più sottosistemi. Eseguiamo ora la quantificazione dell'espressione (1):

$$Pr\{(P1\wedge P2\wedge S1\wedge S2) \mid (BT1 \oplus BT2)\} = Pr\{P1\wedge P2\wedge S1\wedge S2 \wedge ( BT1 \oplus BT2) \} / Pr\{BT1 \oplus BT2\}$$

Siccome $A \oplus B = A \wedge\neg B \vee \neg A \wedge B$ segue che:

$$Pr\{Top\} = Pr\{(P1\wedge P2\wedge S1\wedge S2) \mid (BT1\wedge\neg BT2 \vee \neg BT1 \wedge BT2)\} / Pr\{BT1 \oplus BT2\}$$

Poichè $BT1\wedge\neg BT2$ e $\neg BT1 \wedge BT2$ sono esclusivi, possiamo scrivere:

$$Pr\{Top\} = Pr\{(P1\wedge P2\wedge S1\wedge S2) \wedge (BT1\wedge\neg BT2)\} / Pr\{BT1 \oplus BT2\} + \\ + Pr\{(P1\wedge P2\wedge S1\wedge S2) \wedge (\neg BT1 \wedge BT2)\} / Pr\{BT1 \oplus BT2\}$$

Analizzando l'albero, per l'intervallo di missione di 10.000 ore si ottiene il seguente risultato:

Pr{Top} = 4.0529e-3

E' in considerazione l'implementazione in ASTRA di modulo di calcolo basato su wizard per la determinazione di probabilità condizionate.

## 5.0 CONCLUSIONI E SVILUPPI IN CORSO

L' uso di alberi di guasto non coerenti puo' essere di grande efficacia nel modellare ed indagare stati di guasto complessi di sistemi. Nel presente lavoro si sono mostrati i benefici derivanti dall' uso di alberi non coerenti (contenenti eventi negati) per la soluzione di alcune tipologie di problemi reali difficilmente analizzabili con alberi coerenti. In particolare si sono considerate situazioni relative alla modellazione di top complessi, calcolo di probabilità condizionate e di analisi di procedure di manutenzione.

I problemi illustrati sono stati analizzati con ASTRA-3, il nuovo programma di calcolo sviluppato presso il CCR e basato interamente sull'approccio BDD. ASTRA-3 permette il calcolo esatto delle principali grandezze probabilistiche sia per alberi coerenti che per alberi non coerenti.

L'attività in corso su ASTRA-3 riguarda, oltre l'ottimizzazione di alcuni algoritmi e l'implementazione di alcune nuove funzionalità, il completamento della fase di test dei moduli Database affidabilità componenti (collegato all'editor dell'AdG) e SAM (per l'analisi di sensibilità) e il completamento della documentazione.

## 6.0 RINGRAZIAMENTI

Il presente lavoro è stato effettuato nell'ambito dell'attività di Analisi dei Sistemi Applicata alla Salvaguardia Nucleare e alla Security di Infrastrutture Critiche, svolta presso l'Istituto di Protezione e Sicurezza dei Cittadini del CCR-Ispra. Ringraziamo i capi unità A. Poucet e J. Gonçalves per il sostegno al lavoro. ASTRA e' il risultato di programmi quadro di ricerca diretta del CCR, di contratti terzi e di re-investimenti derivanti dalla commercializzazione. Ringraziamo J. Bonnin della DG-RTD, precedentemente alla DG-JRC, e G. Barry, T. Gering e V. Koutsiouris della DG-JRC per il supporto finanziario dato all'attività di aggiornamento di ASTRA.

## RIFERIMENTI BIBLIOGRAFICI

1. Vesely, W., A Time Dependent Methodology for Fault Tree Evaluation, *Nuclear Engineering and Design*, Vol. 13, 1970, pp 337-360.

2. Rauzy, A., Dutuit, Y., Exact and Truncated Computation of Prime Implicants of Coherent and Non-Coherent Fault Trees within Aralia, *Reliability Engineering and System Safety*, Vol 58, 1997.

3. Bryant, R.E., Graph Based Algorithms for Boolean Functions Manipulation, *IEEE Transactions on Computers*, Vol. C-35, 1986

4. Rauzy, A., New Algorithms for Fault Trees Analysis, *Reliability Engineering and System Safety*, Vol 40 Issue 3, 203-211, 1993.

5. Contini, S., Cojazzi, G.G.M., Renda, G.. On the use of non-coherent fault trees in safety and security studies, *accepted at ESREL 2006*, Lisbon, Portugal, 2006.

6. Cojazzi, G.G.M., Contini, S., Renda, G., FT Analysis in Security related Applications: Challenges and Needs, *ESReDA 29th Seminar, Systems Analysis for a More Secure World.*

*Application of System Analysis and RAMS to Security of Complex Systems*. JRC, Ispra, Italy, 2005.

7. Barlow, R.E. & Proschan, *Statistical Theory of Rel-ability and Life Testing, Probability Models*, Holt, Reinhat and Winston, Inc., USA, F. 1975.

8 Bedford, T., Cooke, R., *Probabilistic risk analysis: Foundations and methods. Cambridge University Press*, Cambridge, 2001.

9. Amesz, J., Francocci G., Righini, R., *Analisi di affidabilità mediante metodo Fault Tree applicata ai circuiti principali di raffreddamento del reattore PEC*. Rapporto interno Euratom, PER 328/79, 1979.

10. Sorman, J., NOT logic in Risk Spectrum PSA Professional, in *Probabilistic Safety Assessment and Management 2004*, C. Spitzer, U. Schmocker and V.N. Dang Eds, Berlino, 2004.

11. Contini, S., Recenti sviluppi metodologici nell'analisi degli alberi di guasto, *Convegno Nazionale su Valutazione e Gestione del Rischio negli Insediamenti Civili e Industriali*, Pisa, 6-8 ottobre 1998.

12. Contini, S., Cojazzi, G., Renda, G., De Cola G., La metodologia ASTRA per l'analisi di affidabilita' di sistemi complessi, *VGR 2004*, Pisa, October 2004.

13. Contini, S., Cojazzi, G.G.M., de Cola, G.. On the exact analysis of non-coherent fault trees: the ASTRA package, *PSAM 8, New Orleans*, USA, 2006.

14. Contini, S. "*ASTRA 3.0.Theoretical manual*", EUR Technical Report, in corso di preparazione.