

# Diagnosi di Sistemi ad Eventi Discreti

Maria Paola Cabasino

6 novembre 2007

## 1 Introduzione

Un importantissimo contributo sulla diagnosi dei sistemi ad eventi discreti è stato dato dal lavoro di Stephane Lafortune e altri collaboratori tra cui Meera Sampath, Raja Sengupta e altri. Questo documento è tratto dai lavori [1, 2].

La diagnosi di un guasto in un sistema ad eventi discreti si sviluppa in due principali fasi: la costruzione del modello del sistema ad eventi discreti che deve essere diagnosticato, seguito dalla costruzione del "protocollo diagnostico", in altre parole l'insieme di regole impiegate per la scoperta e la localizzazione del guasto.

Il comportamento del sistema è modellato con un linguaggio regolare ed è rappresentato da un automa a stati finiti. In breve, un linguaggio è detto diagnosticabile se è possibile scoprire, con un ritardo finito, il verificarsi di certi eventi non osservabili, detti eventi difettosi o guasti. Esiste una procedura sistematica per la scoperta e l'isolamento dei guasti usando i diagnosticatori. Il diagnosticatore è una macchina a stati finiti costruita a partire dal modello di macchina a stati finiti del sistema. Questa macchina esegue la diagnosi osservando il comportamento on-line del sistema. Gli stati del diagnosticatore portano informazioni sul guasto e il verificarsi del guasto può essere scoperto (con un ritardo finito) esaminando questi stati. Le condizioni necessarie e sufficienti affinché un linguaggio sia diagnosticabile sono basate sul diagnosticatore. Quindi, il diagnosticatore ha due scopi:

- la verifica off-line delle proprietà di diagnosticabilità del sistema,
- la scoperta e la localizzazione on-line dei guasti.

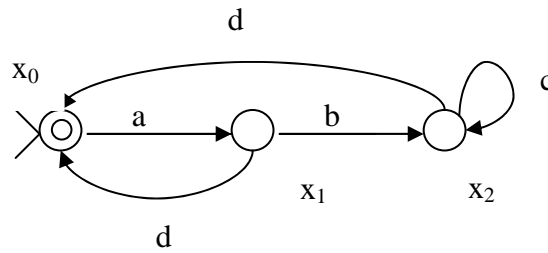


Figura 1: Automa deterministico

### 1.1 Il modello del sistema

Il sistema da diagnosticare è modellato come un automa a stati finiti:

$$G = (X, E, \delta, x_0, X_m)$$

dove  $X$  è lo spazio di stato,  $E$  è l'insieme di eventi,  $\delta : X \times E \rightarrow X$  è la funzione di transizione parziale,  $x_0$  è lo stato iniziale del sistema e  $X_m$  è l'insieme degli stati finali. Nel caso in cui per il sistema considerato non abbia senso definire un insieme di stati finali, l'automa a stati finiti diventa:  $G = (X, E, \delta, x_0)$ .

**Esempio 1.1.** In Figura 1 è mostrato un automa a stati finiti  $G$  con  $X = \{x_0, x_1, x_2\}$ , alfabeto  $E = \{a, b, c, d\}$ , stato iniziale  $x_0$  e insieme di stati finali  $X_m = \{x_0\}$ . La funzione di transizione  $\delta$  è data dalla seguente tabella:

$\delta$	$a$	$b$	$c$	$d$
$x_0$	$x_1$			
$x_1$		$x_2$		$x_0$
$x_2$			$x_2$	$x_0$

Una delle ragioni per cui si usano gli automi a stati finiti per rappresentare un sistema ad eventi discreti è la loro facilità nell'analizzare il comportamento del sistema. Usando delle efficienti strutture dati, come i puntatori, noi possiamo facilmente percorrere il loro diagramma stato-transizione, avanti e indietro, e così avere risposta sulle proprietà dei linguaggi generato e marcato dell'automa.

Il modello  $G$  rende conto sia del comportamento normale, che di quello di guasto del sistema. Il comportamento del sistema è descritto dal linguaggio a prefisso-chiuso  $L = L(G)$  generato da  $G$ .

Un automa non deterministico è caratterizzato da eventi non osservabili. Le  $\varepsilon$ -transizioni di un automa non deterministico sono eventi che si verificano nell'automa, ma che non sono visibili, o

osservabili, dall'esterno. Questa perdita di osservabilità è dovuta all'assenza di un sensore che rileva l'occorrenza dell'evento o al fatto che l'evento accade in una locazione remota, ma non è comunicato al sito centrale; questa è una tipica situazione nei sistemi di informazione distribuiti. Gli eventi di guasto che non causano un immediato cambiamento nella lettura del sensore sono un esempio di eventi non osservabili. Invece che etichettare tutte le transizioni dovute ad eventi non osservabili con  $\varepsilon$  e ottenere un automa non deterministico come modello del sistema, definiamo gli eventi di queste transizioni come "autentici", ma caratterizziamo questi eventi come non osservabili.

In altre parole, il nostro modello del sistema sarà un automa deterministico il cui insieme di eventi è diviso in due insiemi disgiunti:

$$E = E_o \cup E_u,$$

dove  $E_o$  rappresenta l'insieme degli eventi osservabili e  $E_u$  rappresenta l'insieme degli eventi non osservabili. Gli eventi osservabili nel sistema possono essere o comandi generati dal controllore o letture del sensore successive all'esecuzione del comando del controllore o un cambiamento del valore letto dal sensore. Gli eventi non osservabili possono essere guasti o altri eventi che causano cambiamenti nello stato del sistema, non rilevati dai sensori.

Consideriamo  $E_f \subseteq E$  l'insieme degli eventi di guasto che devono essere diagnosticati. Assumiamo, senza perdita di generalità, che  $E_f \subseteq E_u$ , dal momento che un guasto osservabile può essere facilmente diagnosticato. Il nostro obiettivo è di identificare il manifestarsi dei guasti, se ne accadono, date le tracce di eventi osservati dal sistema, in cui ci sono solo gli eventi osservabili appartenenti a  $E_o$ . Dividiamo l'insieme dei guasti in insiemi disgiunti non vuoti corrispondenti ai diversi tipi di guasto:

$$E_f = E_{f1} \cup E_{f2} \cup \dots \cup E_{fm}.$$

Indichiamo con  $\Pi_f = \{1, 2, \dots, m\}$  questa partizione. Essa è motivata dalle seguenti considerazioni:

- Una strumentazione inadeguata può rendere impossibile la diagnosi del singolo guasto.
- Non possiamo pretendere di identificare singolarmente il verificarsi di ogni singolo evento di guasto. Possiamo solamente essere interessati a conoscere se ne è accaduto uno, appartenente a un insieme in cui l'effetto dell'insieme dei guasti sul sistema è lo stesso. Quando si dice "un guasto di tipo  $F_i$  si è verificato" significa che si è verificato qualche evento appartenente all'insieme  $E_{fi}$ .

Faremo le seguenti assunzioni sul sistema da analizzare:

1. Il linguaggio  $L$  generato da  $G$  è vivo, ossia esiste una transizione per ogni stato  $x$  appartenente a  $X$ ; in altre parole il sistema non può raggiungere uno stato morto nel quale nessun evento è possibile.
2. Non esiste in  $G$  alcun ciclo di eventi non osservabili.

La prima assunzione, che riguarda la vivezza del linguaggio, è fatta per motivi di semplicità; mentre la seconda assunzione assicura che le osservazioni avvengano con una certa regolarità. Dal momento che la scoperta dei guasti è basata sulle transizioni osservabili del sistema, richiediamo che il sistema non generi sequenze arbitrariamente lunghe di eventi non osservabili.

Introduciamo ora la nozione di proiezione che verrà utilizzata nel seguito.

Definiamo la *proiezione*  $P : E^* \rightarrow E_o^*$  dove:

$$\begin{cases} P(\varepsilon) = \varepsilon \\ P(\sigma) = \sigma, & \text{se } \sigma \in E_o ; \\ P(\sigma) = \varepsilon, & \text{se } \sigma \in E_u ; \\ P(s\sigma) = P(s)P(\sigma), & s \in E^*, \sigma \in E . \end{cases}$$

$P$  "cancella" semplicemente gli eventi non osservabili in una traccia.

**Esempio 1.2.** Consideriamo la stringa  $s = ab\varepsilon_1a\varepsilon_2$ , dove l'insieme degli eventi osservabili  $E_o = \{a, b\}$  e l'insieme degli eventi non osservabili  $E_u = \{\varepsilon_1, \varepsilon_2\}$ . La proiezione è pari a  $P(s) = aba$ .

## 2 L'Osservatore

Per il problema di diagnosi dei guasti abbiamo già detto che considereremo, per le ragioni sopra spiegate, i guasti come eventi non osservabili. Abbiamo quindi che il modello del sistema è rappresentato da un automa non deterministico, dove compaiono delle transizioni non osservabili. Un automa non deterministico  $G_{nd}$  può essere sempre convertito in un automa deterministico che ha un linguaggio equivalente, cioè un automa che genera e accetta lo stesso linguaggio dell'automata  $G_{nd}$ . Lo spazio di stato dell'automata deterministico equivalente sarà un sottinsieme dell'insieme potenza<sup>1</sup> dello spazio di stato dell'automata non deterministico. Ciò implica che se l'automata non

<sup>1</sup>Dato un insieme  $A$ , la notazione  $2^A$  indica l'insieme potenza (power set) di  $A$ , cioè l'insieme di tutti i sottoinsiemi di  $A$ .

deterministico è a stati finiti, allora l'equivalente automa deterministico sarà pure a stati finiti. Presenteremo ora un algoritmo per la conservazione e trasformazione del linguaggio dall'automa non deterministico a quello deterministico. Chiameremo il risultante automa deterministico *osservatore* corrispondente all'automa non deterministico. Indicheremo con  $Obs(G_{nd})$  (o  $G_{obs}$  quando non ci sarà possibilità di confusione) l'osservatore di  $G_{nd}$ . Questa terminologia è ispirata al concetto di osservatore della teoria dei sistemi; esso cattura il fatto che l'equivalente automa deterministico (l'osservatore) tiene traccia della stima dello stato dell'automa non deterministico su transizioni etichettate dagli eventi in  $E$ .

### Procedura per costruire l'osservatore $Obs(G_{nd})$ dell'automa non deterministico $G_{nd}$

Consideriamo l'automa non deterministico  $G_{nd} = (X, E, f_{nd}, x_0, X_m)$ , dove gli elementi tra parentesi sono rispettivamente l'insieme degli stati  $X$ , l'insieme degli eventi  $E$  (con  $E = E_o \cup E_u$ ), la relazione di transizione  $f_{nd}$ , lo stato iniziale  $x_0$  e l'insieme degli stati finali  $X_m$ . L'osservatore sarà pari a  $Obs(G_{nd}) = (X_{obs}, E_o, f_{obs}, x_{0,obs}, X_{m,obs})$ , (da notare che l'insieme degli eventi è ora costituito dai soli eventi osservabili) e sarà costruito come segue.

**Step 1:** Definisci  $x_{0,obs} = \varepsilon R(x_0)$ <sup>2</sup>. Imponi  $X_{obs} = \{x_{0,obs}\}$ .

**Step 2:** Per ogni  $B \in X_{obs}$  ed  $e \in E$ , definiamo

$$f_{obs}(B, e) = \varepsilon R(\{x \in X : (\exists x_e \in B)[x \in f_{nd}(x_e, e)]\})$$

ogniqualevolta  $f_{nd}(x_e, e)$  è definito per qualche  $x_e \in B$ . In questo caso, aggiungi lo stato  $f_{obs}(B, e)$  a  $X_{obs}$ . Se  $f_{nd}(x_e, e)$  non è definita per nessun  $x_e \in B$ , allora  $f_{obs}(B, e)$  non è definita.

**Step 3:** Ripeti lo Step 1 finchè l'osservatore  $Obs(G_{nd})$  è stato costruito.

**Step 4:**  $X_{m,obs} = \{B \in X_{obs} : B \cap X_m \neq \emptyset\}$ .

Spieghiamo ora i passi chiave di questo algoritmo.

- L'idea di questa procedura è di partire con  $x_{0,obs}$  come lo stato iniziale dell'osservatore che è costituito dallo stato iniziale di  $G_{nd}$  più tutti gli stati che vengono raggiunti da questo con transizioni non osservabili ( $\varepsilon$ -transizioni). Identifichiamo poi tutti gli eventi in  $E$  che sono abilitati da uno o più stati in  $x_{0,obs}$  e mettiamo tali eventi in un insieme

---

<sup>2</sup> $\varepsilon R(x) = f_{nd}(x, \varepsilon)$ , ossia è l'insieme di tutti gli stati che vengono raggiunti a partire da  $x$  con  $\varepsilon$ -transizioni.

che chiamiamo insieme attivo di  $x_{0,obs}$ . Per ogni evento contenuto nell'insieme attivo, identifichiamo tutti gli stati in  $X$  che possono essere raggiunti a partire da uno stato in  $x_{0,obs}$ . Estendiamo poi questo insieme di stati per includere tutti gli stati che vengono raggiunti mediante  $\varepsilon$ -transizioni, questa operazione restituisce  $f_{obs}(x_{0,obs}, e)$  di  $Obs(G_{nd})$ . Questa transizione, cioè l'evento che porta dallo stato  $x_{0,obs}$  allo stato  $f_{obs}(x_{0,obs}, e)$ , è quindi aggiunta al diagramma di transizione degli stati dell'osservatore  $Obs(G_{nd})$ .

Il significato di quanto detto è che un "osservatore esterno" che conosce il modello del sistema  $G_{nd}$ , ma osserva solo le transizioni di  $G_{nd}$  etichettate dagli eventi in  $E$  inizierà con  $x_{0,obs}$  che equivale alla sua stima dello stato di  $G_{nd}$ . Poi osservando l'evento  $e \in E$ , questo osservatore esterno aggiornerà la sua stima dello stato a  $f_{obs}(x_{0,obs}, e)$ , infatti, questo insieme rappresenta tutti gli stati dove  $G_{nd}$  potrebbe essere dopo aver eseguito la stringa  $e$ , preceduta e/o seguita da  $\varepsilon$ .

- La procedura è ripetuta per ogni evento nell'insieme attivo di  $x_{0,obs}$  e dopo, per ogni stato che è stato creato come immediato successore di  $x_{0,obs}$ , e così via per ogni successore di  $x_{0,obs}$ . Chiaramente nel caso peggiore il numero di stati che vengono creati con questa procedura non è più grande dell'insieme di tutti i sottoinsiemi non vuoti di  $X$ .
- Infine ogni stato di  $Obs(G_{nd})$  che contiene uno stato marcato di  $G_{nd}$  è considerato marcato dal punto di vista dell'osservatore  $Obs(G_{nd})$ . Questo perchè questo stato è raggiungibile da  $x_{0,obs}$  con una stringa contenuta in  $\mathcal{L}_m(G_{nd})$ .

Da notare che l'algoritmo appena definito per la costruzione dell'osservatore è un caso particolare dell'algoritmo dato per la costruzione di un automa non deterministico. Infatti, l'automa non deterministico qui considerato è un caso particolare di automa non deterministico, in cui il non determinismo è dato solo dalla presenza delle transizioni non osservabili o  $\varepsilon$ -transizioni (non esiste infatti il non determinismo dovuto a transizioni con la stessa etichetta uscenti dallo stesso stato).

Le proprietà importanti dell'osservatore  $Obs(G_{nd})$  sono:

1.  $Obs(G_{nd})$  è un automa deterministico.
2.  $\mathcal{L}(Obs(G_{nd})) = \mathcal{L}(G_{nd})$ .
3.  $\mathcal{L}_m(Obs(G_{nd})) = \mathcal{L}_m(G_{nd})$ .

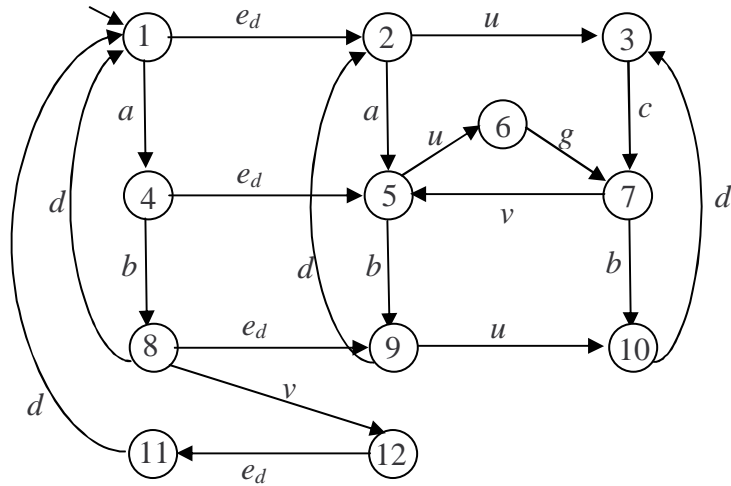


Figura 2: Automa G dell'Esempio 2.1

Il primo risultato è ovvio; gli altri due risultati seguono direttamente dall'algoritmo di costruzione di  $Obs(G_{nd})$ .

**Esempio 2.1.** Consideriamo l'automa  $G$  in Figura 2. L'insieme di eventi non osservabili è pari a  $E_u = \{e_d, u, v\}$ . L'osservatore  $G_{obs} = Obs(G)$  è mostrato in Figura 3. Lo stato  $\{8, 9, 10, 11, 12\}$  raggiunto dopo la stringa  $ab$  in  $G_{obs}$  sta a significare che dopo che si osserva  $ab$ , tutti i possibili stati dove  $G$  potrebbe essere sono 8, 9, 10, 11 e 12.

### 3 Il Diagnosticatore

In molte applicazioni dove il modello del sistema contiene eventi non osservabili, potremmo essere interessati a determinare se alcuni eventi non osservabili *potrebbero essere accaduti* o *devono essere accaduti* nella stringa di eventi eseguita dal sistema. Questo è il problema della *diagnosi di eventi*. Se questi eventi non osservabili di interesse modellano dei guasti dei componenti del sistema, allora la conoscenza che uno di questi eventi sia accaduto è molto importante quando monitoriamo le prestazioni del sistema. Il punto chiave è che più noi continuiamo a osservare il comportamento del sistema, più possiamo ridurre l'incertezza sui prefissi delle stringhe di eventi eseguiti dal sistema. Per esempio consideriamo l'esempio 2.1 ( $G$  è mostrato in Figura 2 ed  $E_u = \{e_d, u, v\}$ ). Dopo che osserviamo la stringa  $t = a$ , non sappiamo se il sistema ha eseguito l'evento non osservabile  $e_d$  o meno. Però dopo che osserviamo la stringa  $s = tg = ag$ , noi sappiamo con certezza che l'evento non osservabile deve essere accaduto. In questo modo possiamo diagnosticare il verificarsi dell'evento non osservabile  $e_d$  dopo che abbiamo osservato

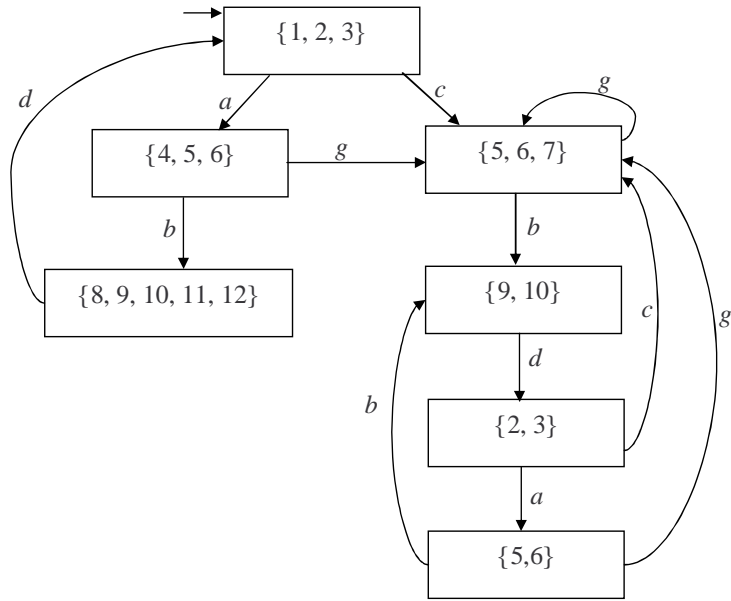


Figura 3: Osservatore  $Obs(G)$  dell'automa  $G$  in Figura 2

s. Possiamo automatizzare questo tipo di deduzione circa il passato modificando la procedura di costruzione dell'osservatore e includendo esplicitamente l'informazione sull'evento  $e_d$  in esso. Chiameremo questo osservatore modificato *diagnostizzatore*.

Il diagnostizzatore è un automa a stati finiti che tiene traccia del verificarsi degli eventi non osservabili, diagnosticando, se possibile, la loro occorrenza. Indichiamo il diagnostizzatore con  $Diag(G)$  o  $G_{diag}$ . Esso è molto simile all'osservatore con la differenza che sono attaccate delle *etichette* agli stati dell'automa  $G$  negli stati di  $Diag(G)$ . Per semplicità assumiamo di voler diagnosticare un solo evento non osservabile  $e_d \in E_u$ . Se volessimo diagnosticare più di un evento, possiamo o costruire un diagnostizzatore per ogni evento che deve essere diagnosticato o costruire un singolo diagnostizzatore che simultaneamente tiene traccia di tutti gli eventi non osservabili. Verranno usate due tipi di etichette:  $N$  per indicare "No,  $e_d$  non si è ancora verificato" e  $Y$  per indicare "Sì,  $e_d$  si è verificato". Quando attacchiamo un'etichetta allo stato dell'automa  $x \in X$ , scriveremo  $xN$  o  $xY$  per abbreviare rispettivamente la notazione  $(x, N)$  o  $(x, Y)$ .

Richiamiamo brevemente la procedura per la costruzione dell'osservatore  $Obs(G)$  di un automa  $G$  con eventi non osservabili introdotta nella sezione 2. Le modifiche da apportare alla costruzione di  $Obs(G)$  allo scopo di costruire  $Diag(G)$  sono:

**M1.** Quando consideriamo gli stati che vengono raggiunti con transizioni non osservabili a partire dallo stato  $x_0$  di  $G$ :

- (a) Attacca l'etichetta  $N$  agli stati che possono essere raggiunti da  $x_0$  con stringhe non osservabili in  $[E_u \setminus e_d]^*$ ;
- (b) Attacca l'etichetta  $Y$  agli stati che possono essere raggiunti da  $x_0$  con stringhe non osservabili che contengono almeno una occorrenza di  $e_d$ ;
- (c) Se uno stato  $z$  può essere raggiunto sia eseguendo  $e_d$  che non eseguendolo, allora nello stato iniziale di  $\text{Diag}(G)$  crea:  $zN$  e  $zY$ .

**M2.** Quando costruisci gli altri stati raggiungibili del diagnosticatore:

- (a) Segui le regole per la funzione di transizione di  $\text{Obs}(G)$ , attaccando le etichette agli stati di  $G$  nel modo sopra indicato;
- (b) Propagazione dell'etichetta  $Y$ : cioè, qualunque stato raggiungibile dallo stato  $zY$  deve mantenere l'etichetta  $Y$  ad indicare che  $e_d$  si è verificato raggiungendo lo stato  $z$  e quindi anche nel processo di raggiungimento del nuovo stato.

**M3.** Non è definito alcun insieme di stati marcati per il diagnosticatore.

Ricapitolando,  $\text{Diag}(G)$  ha come insieme di eventi  $E_o$ , è un automa deterministico e genera il linguaggio  $\mathcal{L}(\text{Diag}(G)) = P[\mathcal{L}(G)]$ . Ogni stato di  $\text{Diag}(G)$  è un sottoinsieme di  $X \times \{N, Y\}$ . Una conseguenza della modifica M1(c) è che non c'è una mappatura uno a uno fra gli stati di  $\text{Diag}(G)$  e gli stati di  $\text{Obs}(G)$ .

L'esempio seguente illustra quando e come le modifiche sopra riportate entrano nella costruzione del diagnosticatore.

**Esempio 3.1.** *La figura 4 mostra il diagnosticatore  $G_{\text{diag}}$  dell'automa  $G$  mostrato in Figura 2, dove  $e_d$  è l'evento che deve essere diagnosticato. Se confrontiamo l'osservatore  $G_{\text{obs}}$  in Figura 3 con il diagnosticatore  $G_{\text{diag}}$  in Figura 2, possiamo notare che  $G_{\text{diag}}$  ha più stati di  $G_{\text{obs}}$ , ciò è dovuto al fatto che alcuni stati di  $G$  potrebbero apparire due volte in uno stato di  $G_{\text{diag}}$ , una volta con l'etichetta  $N$  e una volta con l'etichetta  $Y$ . Questo è il caso che avviene dopo l'osservazione della stringa  $abd$ , dove nello stato del diagnosticatore compare sia  $1N$  che  $1Y$ .*

*Il diagramma di transizione dello stato di  $G_{\text{diag}}$  mostra che dopo che viene osservato l'evento  $c$  o l'evento  $g$ , siamo sicuri che l'evento  $e_d$  si è verificato dal momento che tutti gli stati di  $G$  che compaiono negli stati del diagnosticatore hanno la stessa etichetta  $Y$ . Mentre potremmo non sapere esattamente lo stato di  $G$ , sappiamo con certezza che l'evento  $e_d$  è accaduto. D'altra parte,*

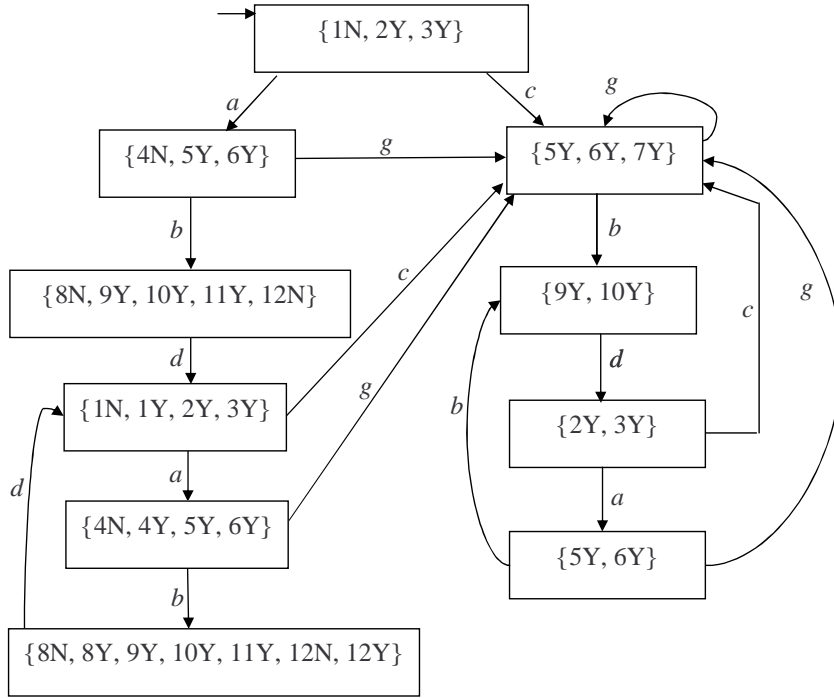


Figura 4: Diagnosticatore  $G_{diag}$  di  $G$  in Figura 2

se la stringa osservata non contiene nessun evento  $c$  o  $g$ , siamo negli stati di  $G_{diag}$  dove abbiamo stati sia con l'etichetta  $N$  che con l'etichetta  $Y$ . Quindi, anche se è possibile che l'evento  $e_d$  sia accaduto, noi non siamo certi sulla sua occorrenza.

Come si può notare dall'Esempio 3.1 possiamo trarre conclusioni sul verificarsi dell'evento  $e_d$  dall'esame degli stati del diagnosticatore. Formalizziamo questo processo. La diagnosi on-line dell'evento  $e_d$  è condotta tenendo traccia del corrente stato del diagnosticatore in risposta agli eventi osservati eseguiti dal sistema  $G$ . Possiamo avere tre tipi di stato nel diagnosticatore:

- Se tutti gli stati di  $G$  nello stato corrente di  $Diag(G)$  hanno etichetta  $N$ , allora siamo sicuri che l'evento  $e_d$  non si è ancora verificato. Chiameremo uno stato di questo tipo uno *stato negativo*.
- Se tutti gli stati di  $G$  nello stato corrente di  $Diag(G)$  hanno etichetta  $Y$ , allora siamo sicuri che l'evento  $e_d$  si è verificato ad un certo punto nel passato. Chiameremo uno stato di questo tipo uno *stato positivo*.
- Se lo stato corrente di  $Diag(G)$  contiene almeno uno stato di  $G$  con etichetta  $N$  e almeno uno stato di  $G$  con etichetta  $Y$ , allora l'evento  $e_d$  potrebbe o meno essersi verificato precedentemente. Chiameremo uno stato di questo tipo uno *stato incerto*. In questo caso,

esistono due stringhe  $s_1$  ed  $s_2$  in  $\mathcal{L}(G)$  tali che  $P(s_1) = P(s_2)$  (quindi, portano allo stesso stato in  $\text{Diag}(G)$ ), dove  $s_1$  contiene il guasto ed  $s_2$  no.

## 4 Diagnosticabilità

Riconsideriamo l'automa  $G$  in Figura 2 e il suo diagnosticatore in Figura 4 e consideriamo le seguenti due stringhe in  $\mathcal{L}(G)$ :  $s_N = (abd)^m$  e  $s_Y = e_d(abd)^m$ . La stringa  $s_Y$  ha la seguente proprietà: dato un qualunque  $n \in \mathbb{N}$ , esiste sempre un  $m$  tale che  $\|s_Y/e_d\| > n$ . Quando si verifica questo caso diremo che " $s_Y$  è arbitrariamente lunga dopo  $e_d$ ". In altre parole, il suffisso di  $s_Y$  dopo  $e_d$  cicla in un loop di  $G$ . Chiaramente  $P(s_Y) = P(s_N)$ . Così, se  $s_Y$  è eseguito dal sistema, allora non saremo mai in grado di diagnosticare con certezza l'occorrenza di  $e_d$ : dopo il verificarsi di  $e_d$ ,  $P(s_N) = P(s_Y)$  ciclerà negli stati incerti del diagnosticatore  $G_{diag}$ . Quando ciò accade, diciamo che il manifestarsi di  $e_d$  in  $s_Y$  è *non diagnosticabile*. Formalizziamo ora la nozione di diagnosticabilità. E' semplice dare la definizione di diagnosticabilità per linguaggi che sono vivi, cioè linguaggi che non contengono stringhe di terminazione: un linguaggio  $L$  è detto *vivo* se per qualunque  $s \in L$  esiste un  $e$  tale che  $se \in L$ . In termini di automa, un automa che non ha stati di deadlock genera un linguaggio vivo.

**Definizione 4.1.** *L'evento non osservabile  $e_d$  è non diagnosticabile nel linguaggio vivo  $\mathcal{L}(G)$  se esistono due stringhe  $s_N$  ed  $s_Y$  in  $\mathcal{L}(G)$  che soddisfano le seguenti condizioni: (i)  $s_Y$  contiene  $e_d$  ed  $s_N$  no; (ii)  $s_Y$  è arbitrariamente lunga dopo  $e_d$ ; ed (iii)  $P(s_N) = P(s_Y)$ . Quando non esiste una coppia di stringhe come quella appena definita,  $e_d$  è detto **diagnosticabile** in  $\mathcal{L}(G)$ .*

Per proibire situazioni in cui  $G$  continua a eseguire eventi non osservabili dopo il verificarsi di  $e_d$ , che significa che nessuna diagnosi può essere fatta in quanto lo stato di  $\text{Diag}(G)$  non viene mai aggiornato, è consuetudine assumere che  $G$  non abbia cicli di eventi non osservabili che sono raggiungibili dopo una qualunque occorrenza di  $e_d$ . D'ora in poi assumeremo che questa ipotesi sia sempre valida. Quando la proprietà di diagnosticabilità è soddisfatta, siamo sicuri che l'evento  $e_d$  si è verificato, quindi  $\text{Diag}(G)$  entrerà in uno stato positivo in un numero finito di eventi dopo il verificarsi di  $e_d$ . Per vedere ciò, osserviamo che:

- (i)  $G$  non ha cicli di eventi non osservabili dopo  $e_d$  (per ipotesi).
- (ii)  $\text{Diag}(G)$  non può ciclare in un ciclo di stati incerti in quanto questo contraddirebbe la diagnosticabilità.
- (iii)  $\text{Diag}(G)$  ha un insieme di stati finito.

Quando il diagnosticatore ha un ciclo di stati incerti, potenzialmente potrei non essere certo sul verificarsi del guasto  $e_d$  per un numero di eventi (osservabili) arbitrariamente lungo se questo non esce mai dal ciclo. Se riusciamo a trovare due stringhe  $s_Y$  ed  $s_N$  dove  $e_d \in s_Y$ ,  $e_d \notin s_N$ ,  $P(s_N) = P(s_Y)$ , e  $P(s_Y)$  entra e non esce mai dal ciclo di stati incerti del diagnosticatore, rileviamo una violazione della diagnosticabilità. Consideriamo l'Esempio 3.1 (vedi Figure 2 e 4), si può notare che possiamo associare al ciclo di stati incerti in  $G_{diag}$  il ciclo "2  $\rightarrow$  5  $\rightarrow$  9  $\rightarrow$  2" in  $G$ , ed inoltre qualunque stringa che rimane in questo ciclo in  $G$  deve contenere l'evento  $e_d$  dal momento che questi stati hanno etichetta "Y" in  $G_{diag}$ . Possiamo anche associare a questo ciclo di stati incerti in  $G_{diag}$  il ciclo "1  $\rightarrow$  4  $\rightarrow$  8  $\rightarrow$  1" in  $G$ , ed inoltre qualunque stringa che rimane in questo ciclo in  $G$  non deve contenere l'evento  $e_d$  dal momento che questi stati hanno etichetta "N" in  $G_{diag}$ . Possiamo usare questi due cicli in  $G$  per costruire due stringhe  $s_Y$  ed  $s_N$  che soddisfano le condizioni precedentemente enunciate e quindi violano la diagnosticabilità.

In generale, possiamo associare ad un ciclo di stati incerti in  $Diag(G)$  due cicli in  $G$ , uno che include solo stati con etichetta  $Y$  negli stati incerti e uno contenente solo stati con etichetta  $N$  negli stati incerti, chiameremo tale ciclo in  $Diag(G)$  un *ciclo indeterminato*. Per definizione, la presenza di un ciclo indeterminato implica una violazione della diagnosticabilità. Viceversa, una violazione della diagnosticabilità genererà la presenza di un ciclo indeterminato in  $Diag(G)$ , dal momento che  $Diag(G)$  è deterministico ed ha uno spazio di stato finito. Questo ci permette di concludere che la proprietà di diagnosticabilità può essere testata analizzando i cicli di stati incerti in  $Diag(G)$ . Se uno qualunque di questi è indeterminato, allora la diagnosticabilità è violata.

E' importante sottolineare che la presenza di un ciclo di stati incerti in un diagnosticatore non implica necessariamente il fatto di non riuscire a non diagnosticare il verificarsi di  $e_d$ . Consideriamo il seguente esempio.

**Esempio 4.1.** *Consideriamo il sistema e il diagnosticatore mostrato in Figura 5 dove  $e_d$ , l'evento che deve essere diagnosticato, è l'unico evento non osservabile del sistema. Questo diagnosticatore ha un ciclo di stati incerti. Però, non possiamo formare un ciclo nel sistema dai valori che compaiono negli stati incerti del diagnosticatore, aventi etichetta  $Y$ . L'unico ciclo del sistema che può causare il fatto che il diagnosticatore rimanga in tale ciclo di stati incerti è il ciclo "7  $\rightarrow$  11  $\rightarrow$  12  $\rightarrow$  7", e questi stati hanno tutti etichetta  $N$  nei corrispondenti stati del diagnosticatore. Il ciclo di stati incerti del diagnosticatore non è quindi indeterminato. Grazie all'assenza di cicli indeterminati, possiamo dire che nel sistema il verificarsi dell'evento  $e_d$  è sempre diagnosticabile. Infatti, se l'evento  $e_d$  si manifesta, il diagnosticatore lascerà il ciclo di stati incerti ed entrerà nello*

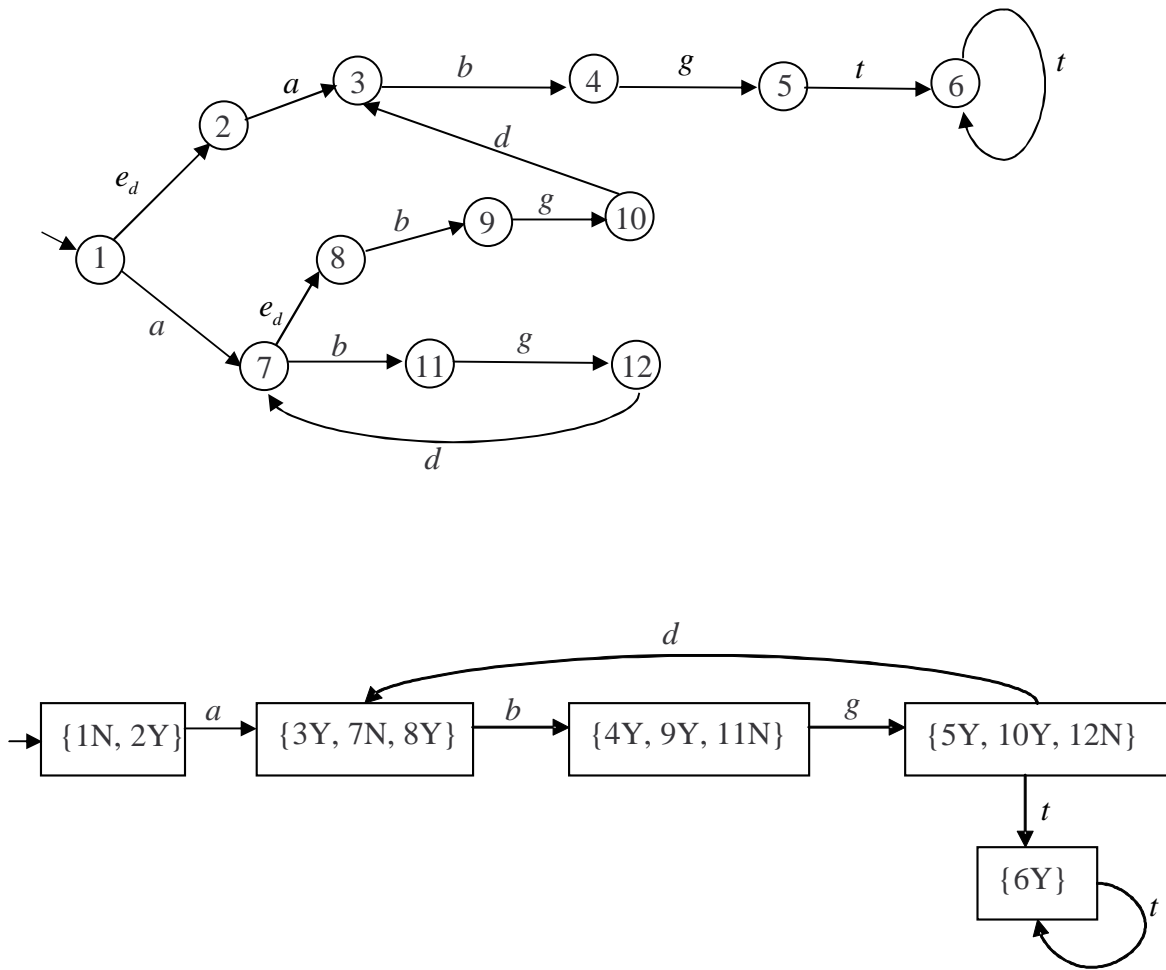


Figura 5: Sistema e corrispondente diagnosticatore per l'evento non osservabile  $e_d$ .

stato  $6Y$  con l'osservazione dell'evento  $t$ . In questo esempio, il fatto che il sistema possa ciclare negli stati  $7$ ,  $12$ , e  $11$ , causando il fatto che il diagnosticatore cicli nel suo ciclo di stati incerti, non è visto come una perdita di diagnosticabilità, dal momento che le stringhe che causano questo "ciclare" non contengono  $e_d$ .

C'è da notare come questo approccio dia la possibilità non solo di dire se il sistema sia diagnosticabile o meno, ma se è diagnosticabile possiamo dire con quale ritardo scopriamo che il guasto si è verificato, ossia dopo quanti passi dall'evento di guasto siamo in grado di dire che il guasto si è verificato. Il numero di passi, ossia il numero di transizioni che dobbiamo osservare dopo il guasto per scoprire la sua occorrenza, viene calcolato facilmente a partire dal sistema. In questo esempio, il numero di passi dopo il guasto che mi fanno capire che il guasto si è verificato è 6 (il suffisso della stringa dopo  $e_d$  è "bgdbgt").

In altre parole, la diagnosticabilità richiede che ogni evento di guasto conduca a delle osservazioni

distinte, sufficienti per permettere unicamente l'identificazione del guasto con un ritardo finito.

Il caso di guasti multipli dello stesso tipo, ossia appartenenti allo stesso insieme della partizione, richiede particolare attenzione. Quando più di un guasto dello stesso tipo, detto  $f_i$ , si manifesta lungo una traccia  $s$  di  $L$ , la definizione sopra data di diagnosticabilità, non richiede che ognuna di queste occorrenze sia scoperta. E' sufficiente essere capaci di concludere, dopo un numero limitato di eventi dopo il verificarsi del primo guasto, che lungo  $s$ , un guasto dell'insieme  $E_{f_i}$  è avvenuto.

Facciamo un esempio.

**Esempio 4.2.** *Consideriamo il sistema rappresentato in Figura 6(a). Gli eventi  $a, b, c$  e  $d$  sono osservabili,  $z$  è un evento non osservabile, mentre  $g_1, g_2$  e  $g_3$  rappresentano i guasti.*

$$E_o = \{a, b, c, d\}, E_u = \{z, g_1, g_2, g_3\}, E_f = \{g_1, g_2, g_3\}$$

*Consideriamo che lo stato iniziale del sistema sia  $x_1$ . Se scegliamo le partizioni:*

$$E_{f1} = \{g_1, g_2\}, E_{f2} = \{g_3\}$$

*non è richiesto che si debba distinguere tra i guasti  $g_1$  e  $g_2$ . Il diagnosticatore per questo primo sistema è mostrato in Figura 6(b). Il sistema illustrato è diagnosticabile con un numero di eventi  $n_1 = 2$  e  $n_2 = 1$ . Infatti considerate le parole  $\omega_1 = bg_1g_2c$ ,  $\omega_2 = bg_1zc$ ,  $\omega_3 = bg_1g_3d$ , in tutte possiamo identificare come traccia che termina in un evento di guasto, la traccia  $s = bg_1$ , quindi poichè  $n_1$  è il numero massimo di transizioni del sistema in cui si può scoprire il verificarsi di un guasto dopo  $s$ , risulta  $n_1 = 2$ . Allo stesso modo per l'insieme  $E_{f2}$  possiamo considerare la parola  $\omega_1 = bg_1g_3d$ . La traccia  $s$  in questo caso è pari a  $s = bg_1g_3$ , conseguentemente il numero massimo di transizioni del sistema in cui si può scoprire il verificarsi di un guasto dopo  $s$  risulta  $n_2 = 1$ . Il fatto che il sistema sia diagnosticabile è facilmente osservabile anche dal diagnosticatore, in quanto questo non contiene cicli di stati incerti, gli unici tre cicli sono infatti relativi ad uno stato negativo (cappio con l'evento  $a$ ) e due stati positivi (cappi con gli eventi  $c$  e  $d$ ). D'altra parte se la partizione fosse stata:*

$$E_{f1} = \{g_1\}, E_{f2} = \{g_2\} \text{ e } E_{f3} = \{g_3\},$$

*allora il sistema sarebbe stato non diagnosticabile dal momento che non è possibile determinare il manifestarsi dell'evento  $g_2$ . Infatti se osservo la stringa  $bc$  non sono in grado di dire se il guasto  $g_2$  si è verificato o meno, perchè i due eventi potrebbero appartenere indifferentemente alle due parole  $\omega_1 = bg_1g_2c$  e  $\omega_2 = bg_1zc$ . Quindi  $g_2$  non è diagnosticabile perchè non conduce a delle osservazioni distinte, sufficienti per permettere unicamente l'identificazione del guasto con un*

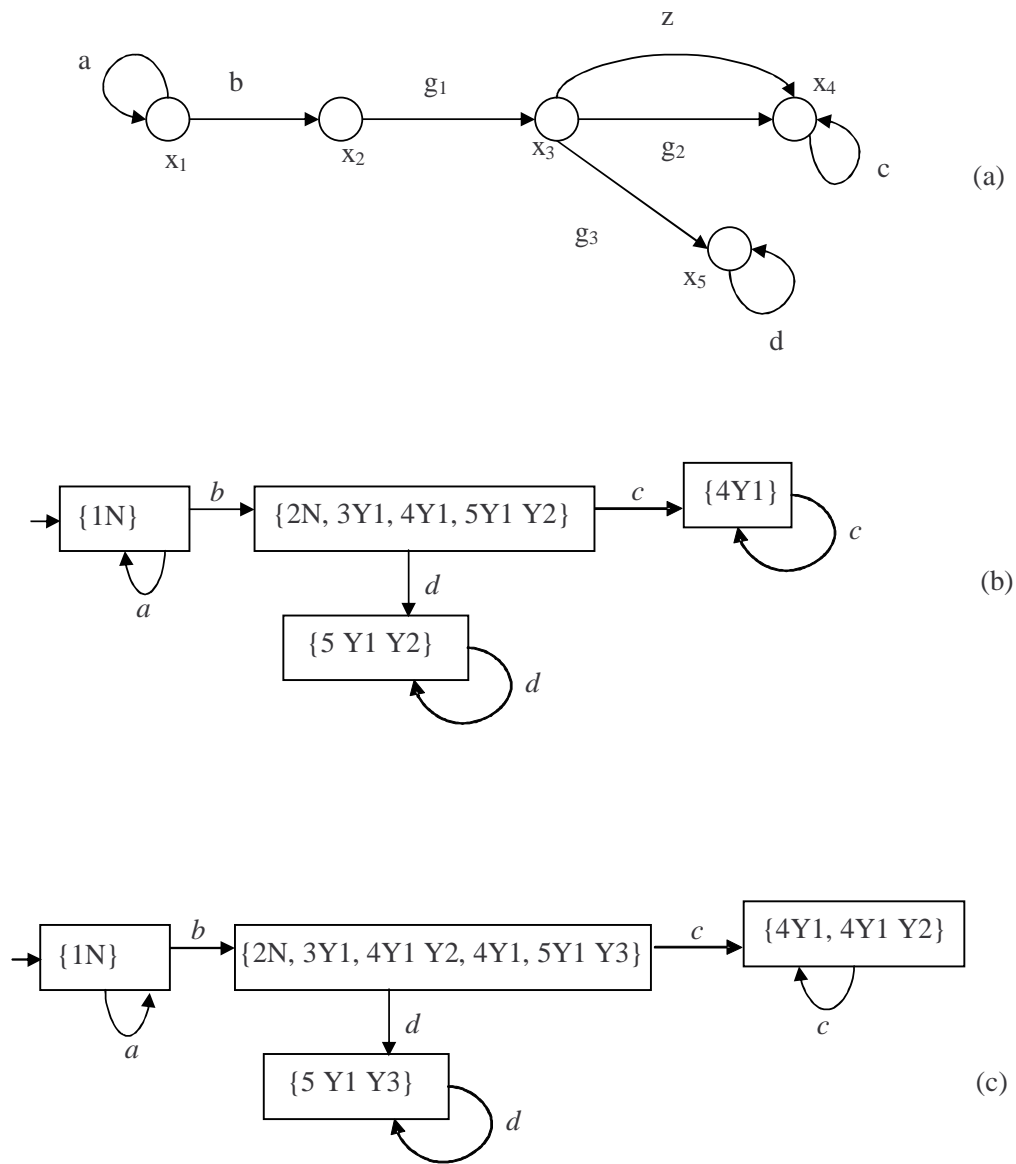


Figura 6: (a) Sistema con guasti multipli. (b) Diagnostico con due classi di guasto:  $E_{f1} = \{g_1, g_2\}, E_{f2} = \{g_3\}$ . (c) Diagnostico con tre classi di guasto:  $E_{f1} = \{g_1\}, E_{f2} = \{g_2\}$  e  $E_{f3} = \{g_3\}$ .

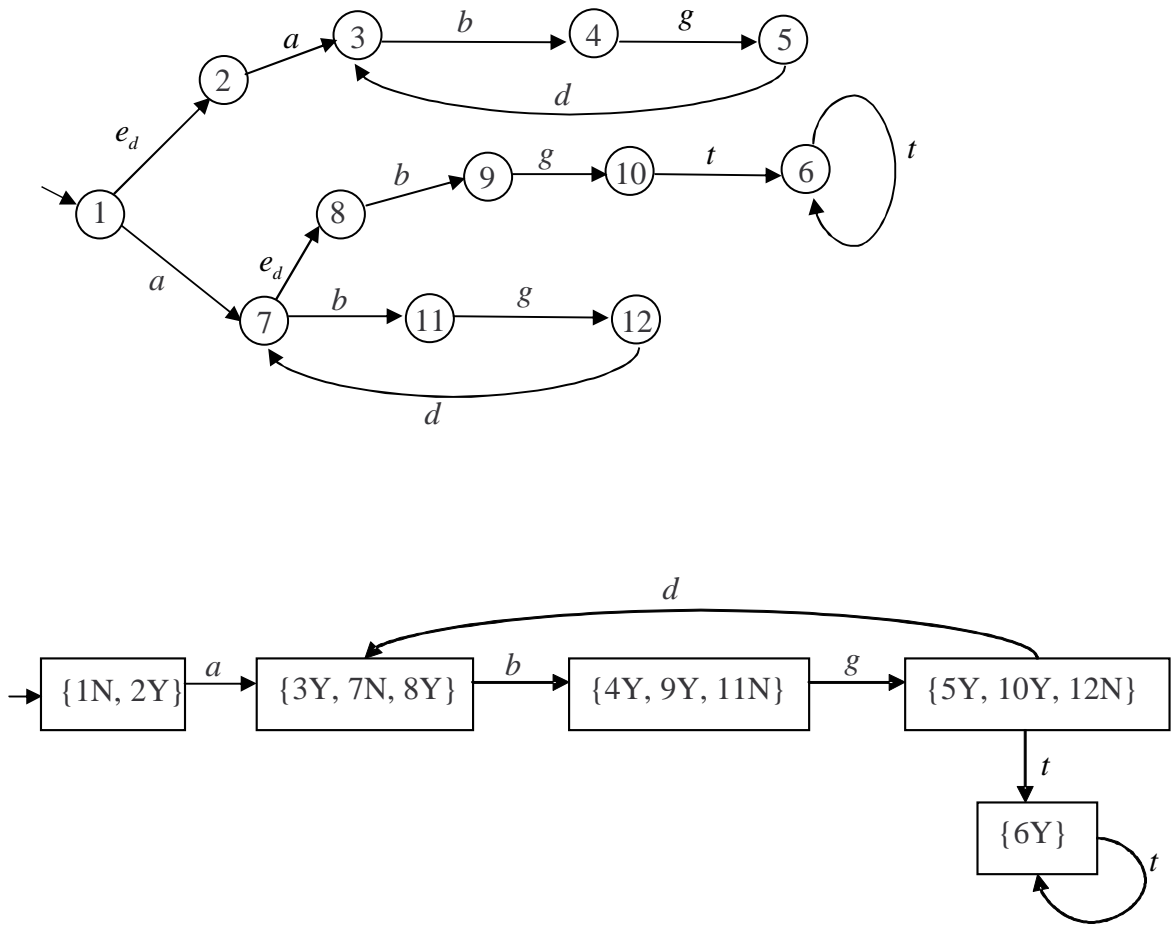


Figura 7: Sistema e corrispondente diagnosticatore per l'evento non osservabile  $e_d$ .

ritardo finito. Questo fatto è sottolineato anche dal ciclo indeterminato. Tale ciclo, rappresentato dal coppia nello stato  $\{4Y1, 4Y1Y2\}$ , è indeterminato in quanto il guasto appartenente alla classe  $E_{f2}$  compare una volta con l'etichetta  $N$  (infatti  $4Y1$  equivale a scrivere:  $4Y1N2N3$ ) e una volta con l'etichetta  $Y$  (in  $4Y1Y2$ ).

Vediamo altri due esempi per chiarire ulteriormente i concetti visti.

**Esempio 4.3.** Consideriamo l'automa e il corrispondente diagnosticatore mostrati in Figura 7, in cui l'unico evento non osservabile e di guasto è  $e_d$ . Il diagnosticatore ha un ciclo di stati incerti, corrispondente alla sequenza di eventi  $bgd$ . Inoltre i diagnosticatori di Figura 5 e di Figura 7 sono identici. In questo caso però il ciclo di stati incerti è anche indeterminato, infatti corrispondente a questo ciclo nel diagnosticatore, troviamo due cicli nell'automa  $G$ : il primo coinvolge gli stati 3, 4, e 5 che compaiono con l'etichetta  $Y$  nel ciclo del diagnosticatore, mentre il secondo coinvolge gli stati 7, 11, e 12 che portano l'etichetta  $N$  nel ciclo del diagnosticatore.

**Esempio 4.4.** Consideriamo il sistema  $G$  rappresentato in Figura 8. Gli eventi  $\alpha, \beta, \gamma, \delta$  e  $\sigma$

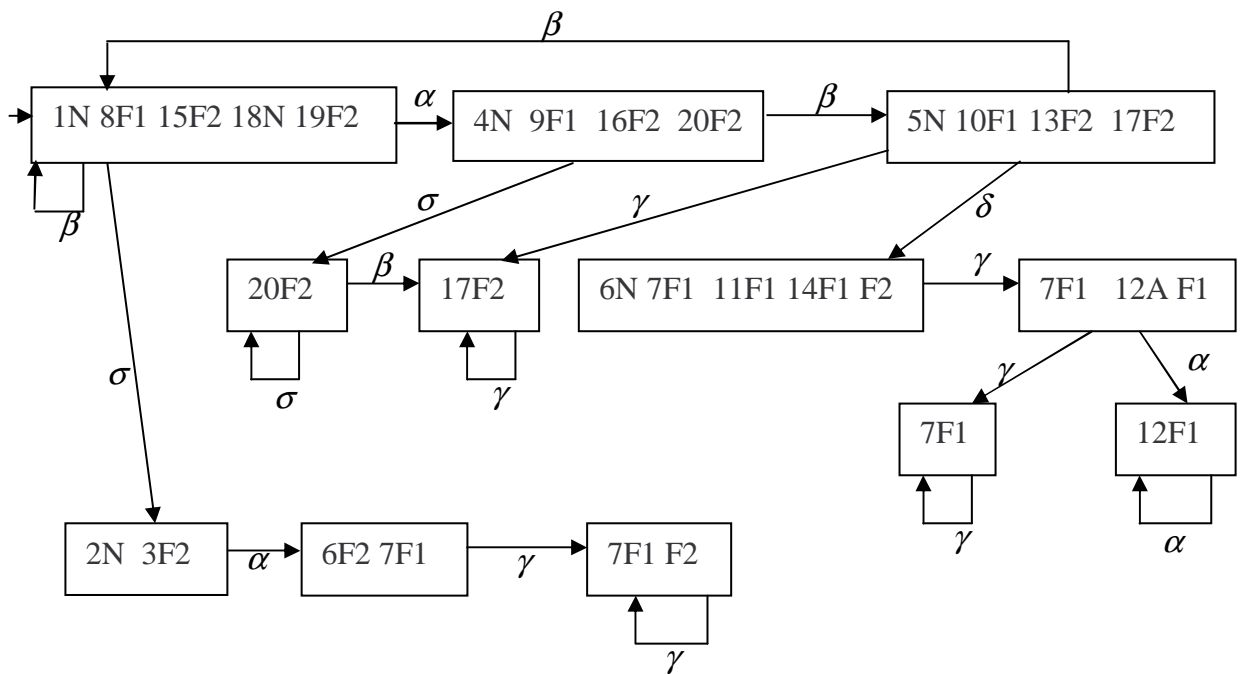
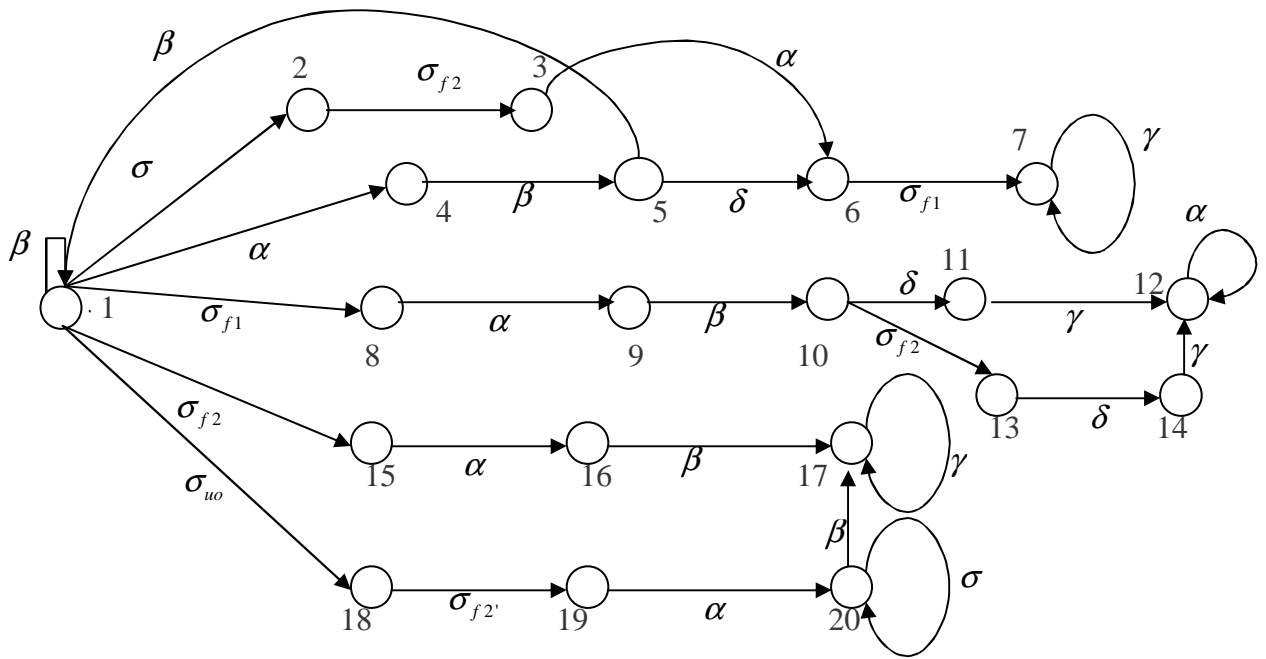


Figura 8: Sistema e corrispondente diagnosticatore per l'evento non osservabile  $e_d$ .

sono osservabili, mentre  $\sigma_u, \sigma_{f1}, \sigma_{f2}$  e  $\sigma_{f2'}$  sono eventi non osservabili. Consideriamo le classi di guasto  $E_{f1} = \{\sigma_{f1}\}$  e  $E_{f2} = \{\sigma_{f2}, \sigma_{f2'}\}$ . In basso nella stessa figura è rappresentato il diagnosticatore. Come si può notare da un'accurata analisi del diagnosticatore tale sistema è diagnosticabile, in quanto i cicli incerti del diagnosticatore non sono anche indeterminati.

## Riferimenti bibliografici

- [1] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, D. Teneketzis. *Diagnosability of Discrete-Event Systems*. In *Proc. IEEE Transaction on Automatic Control* , 9 settembre 1995.
- [2] C. G. Cassandras, S. Lafortune. *Introduction to Discrete Event Systems*. In *Kluwer Academic Publisher* 1999.