Fondamenti di Automatica

Docente: Prof. Giuseppe CONTE

Dispense e materiale didattico

http://www.diiga.univpm.it/C3I085A → Materiale didattico

0. Introduzione

L'Automatica è una disciplina del settore dell'Ingegneria dell'Informazione.

Come tale, essa si occupa di sviluppare metodi e strumenti per la realizzazione di macchine e sistemi artificiali che siano capaci di regolare il proprio funzionamento senza bisogno di ricorrere all'intervento dell'uomo (macchine e sistemi automatici).

Nelle fasi di studio, l'Automatica si avvale di strumenti prevalentemente matematici.

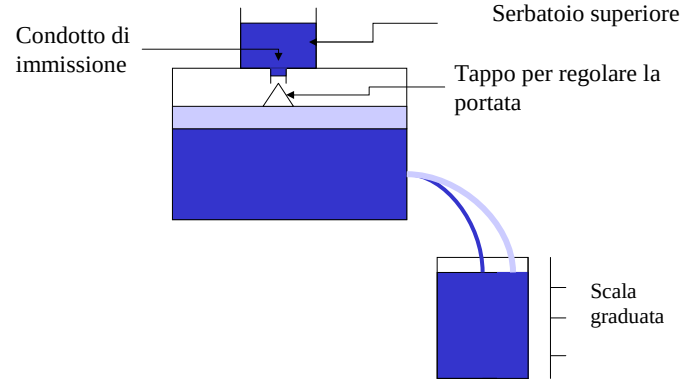
Nelle fasi applicative/realizzative l'Automatica si avvale di strumenti matematici, informatici e tipici dell'elettronica e delle telecomunicazioni.

Gli ambiti di applicazione coprono la totalità dei settori ingegneristici e si estendono a quelli dell'economia, della medicina ed anche delle scienze sociali.

0.1 Esempi

Esempio 1

L'orologio ad acqua di età alessandrina (3° secolo a.c.):



0.2 Esempio/Regolatore di Watt

Esempio 2: Regolatore di Watt

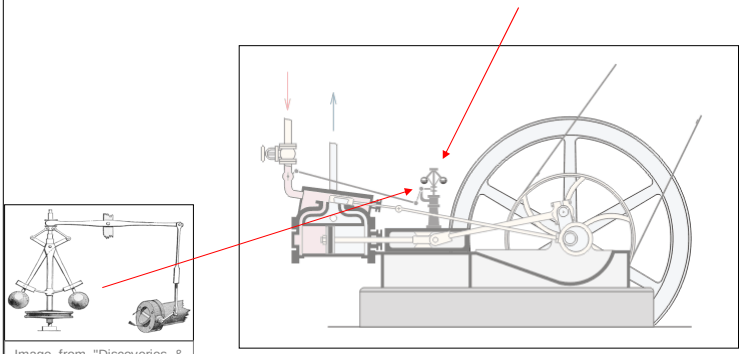
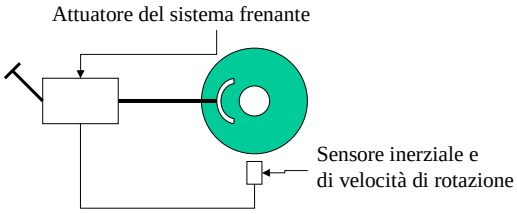


Image from "Discoveries & Inventions of the Nineteenth Century" by R. Routledge, 13th edition, published 1900

0.3 Esempi

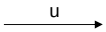
Esempio 3

Sistema ABS per la regolazione della frenata:

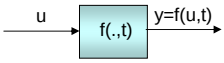


0. 4 Diagrammi a blocchi

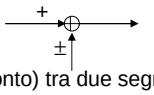
I Diagrammi a Blocchi forniscono un linguaggio semplice ed efficace per descrivere il funzionamento di sistemi automatici di controllo.
Gli elementi che costituiscono i Diagrammi a Blocchi sono 4:

Freccia 

indica il trasferimento di un segnale u, rappresentato mediante una variabile fisica, nel senso indicato.

Blocco 

descrive la relazione tra due (o più) segnali, uno dei quali è visto come causa e l'altro è visto come effetto (.....); la funzione che esprime tale relazione viene scritta nel blocco.

Nodo Sommatore 

indica l'effettuazione di una operazione di somma o sottrazione (confronto) tra due segnali.

Biforcazione 

indica lo sdoppiamento di un segnale che viene ad agire simultaneamente come causa in due distinti blocchi.

0. 5 Diagramma del Regolatore di Watt

Descrizione di funzionamento del Regolatore di Watt mediante diagramma a blocchi

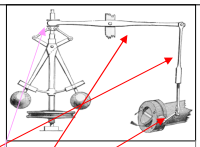
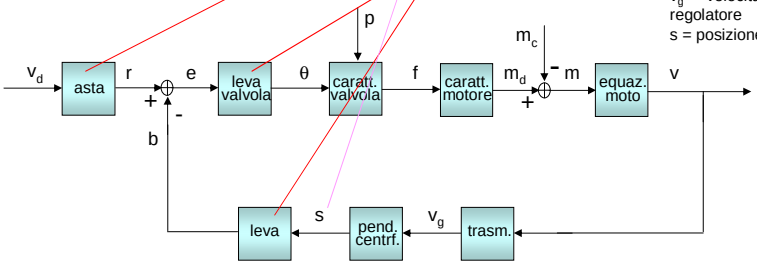


Image from "Discoveries & Inventions of the Nineteenth Century" by R. Routledge, 13th edition, published 1900

v_d = velocità desiderata
r = posizione di riferimento asta
b = posizione leva
θ = posizione angolare valvola
p = pressione vapore
f = flusso vapore
m_d = coppia
m_c = carico
m = coppia effettiva
v = velocità effettiva asse
v_g = velocità angolare regolatore
s = posizione del collare



1.1 Oggetti fisici e modelli

Mediante i metodi e gli strumenti dell'Automatica vogliamo intervenire sugli oggetti e sui fenomeni fisici che riscontriamo nel mondo che ci circonda, che interagiscono con noi in vario modo e che esercitano su di noi effetti di vario tipo.

Per poter governare e gestire, secondo determinati obiettivi, il nostro rapporto con questa realtà fisica ci occorrono innanzitutto degli strumenti che ci permettano, in tutta generalità, di rappresentare in modo agevole gli oggetti e i fenomeni fisici in relazione al rapporto che essi hanno tra di loro e con noi.

Tale rappresentazione si avvale di modelli che, in pratica, concentrano ed esplicitano (parte del)l'informazione disponibile su un determinato oggetto o fenomeno.

1.2 Modelli/Esempi

1.1 Esempio

Numerose attività umane su larga scala (nell' agricoltura, commercio, industria, …) sono fortemente influenzate dall'avvicendarsi delle stagioni. Questa osservazione stimolò, agli inizi della storia umana, l'interesse per la realizzazione di un calendario, che non è altro che un modello del ciclo annuale delle stagioni. Sapere quanti giorni mancano all'arrivo dell'inverno o all'inizio della stagione delle piogge consente di organizzare al meglio le attività menzionate sopra ed è anzi un fattore chiave di successo in epoche pre-tecnologiche. Ciò spiega il motivo per il quale le società primitive impegnarono cospicue risorse (in una società pre-tecnologica occorre che numerosi individui lavorino alla produzione di cibo e mezzi di sostentamento per consentire ad un altro individuo di dedicarsi soltanto ad un lavoro intellettuale) nell'osservazione dei fenomeni astronomici e atmosferici, cioè nella raccolta di dati sul comportamento del ciclo stagionale, al fine di elaborare sotto forma di calendario un preciso modello dell'avvicendarsi delle stagioni.

L'uso che si fa del modello, in questo caso, consiste nel formulare previsioni sull'andamento futuro del fenomeno in oggetto sulla base della situazione attuale.

(Altri esempi simili?)

1.3 Modelli/Esempi

1.2 Esempio

Quando siamo sotto una doccia abbiamo necessità di regolare il getto d'acqua sia come intensità che come temperatura e lo facciamo agendo sui rubinetti dell'acqua calda e fredda o sul miscelatore. Il sistema fisico costituito dalla caldaia o dal serbatoio di acqua calda e dall'impianto idraulico risponde alla nostra azione variando la portata e la temperatura in maniera caratteristica. L'esperianza ci mostra che prima di ottenere il risultato desiderato dobbiamo "imparare" come reagisce tale sistema (accade, ad esempio, che se diminuiamo troppo la portata, non sia possibile mantenere una temperatura sufficientemente alta, oppure che intercorra un ritardo significativo tra la nostra azione e il manifestarsi di una risposta apprezzabile). Imparare come reagisce il sistema equivale a farsi un modello mentale dello stesso (in relazione a quanto esemplificato prima, il nostro modello della doccia conterrà l'informazione relativa alla portata minima per mantenere una temperatura sufficientemente alta o all'entità del ritardo con il quale il sistema reagisce in maniera apprezzabile ai nostri comandi). La nostra doccia sarà tanto più piacevole quanto più è accurato il modello che ci siamo fatti. L'uso che si fa del modello, in questo caso, consiste nell'utilizzarlo per formulare una strategia di controllo, da attuare azionando i rubinetti o il miscelatore, per ottenere un risultato desiderato (si confronti col caso visto in precedenza).

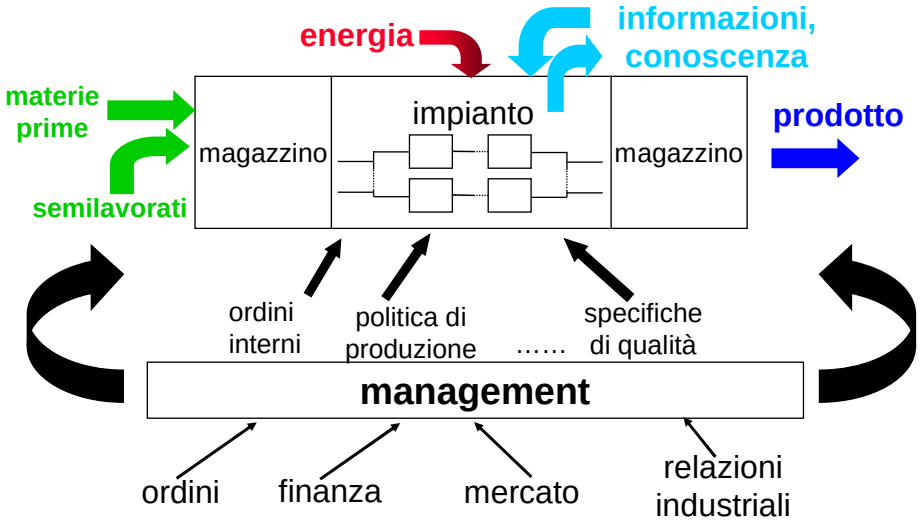
(Altri esempi simili?)

1.4 Modelli/Esempi

Esempio:
Sistema di produzione
•Insieme organizzato di persone e macchine operanti
- entro spazi appositi
- con tempi e ritmi opportuni
- secondo procedure formalizzate
- per produrre beni o utilities
- a partire da materie prime e/o semilavorati

•La finalità è rendere disponibili ai clienti prodotti
- nella quantità stabilita ed ai prezzi fissati
- con le caratteristiche fissate
- nei tempi stabiliti
- ottimizzando i costi

Sistema di Produzione



•Risorse finite da impegnare in modo efficiente
•Comportamento da realizzare in modo preciso

1.4 Modelli/Esempi

•Produzione di ciclomotori.

•Produzione di energia elettrica.

•Gestione del traffico su una rete telefonica.

•Gestione del traffico automobilistico/ferroviario.

•Produzione di un pasto.

- Occorre disporre delle materia prime o degli eventuali semilavorati nelle quantità e con la tempistica richiesta.
- Diverse lavorazioni (lavaggio, taglio, cottura, eccetera) avvengono in uno specifico impianto (la cucina)., dotato di diversi macchinari (forno, fornelli, lavapiatti, eccetera). Ciascuna lavorazione avviene secondo procedure specifiche.
- La produzione deve rispettare un certo numero di macro specifiche(composizione del pasto, tempistica, qualità, eccetera).

2.1 Sistemi dinamici

I modelli che utilizziamo nell'ambito dell'Automatica prendono il nome di SISTEMI o SISTEMI DINAMICI. Il nostro primo obiettivo sarà dunque quello di comprendere appieno cosa sia un SISTEMA e cosa significhi usarlo come modello di un oggetto o di un fenomeno fisico.

Per dare una definizione operativa di SISTEMA occorre introdurre preliminarmente alcuni concetti.

Consideriamo un insieme T i cui elementi verranno usati come indici. Possiamo scegliere, ad esempio, $T = \{1,2,3,\ldots\}$, oppure $T = \mathbf{Z}$ (insieme dei numeri relativi) o $T = \mathfrak{R}$ (insieme dei numeri reali). La cosa importante è che T sia un insieme *ordinato.* Nel caso della prima scelta che abbiamo esemplificato, assegnare gli indici 1,2,3,…. agli oggetti di un insieme significa disporre tali oggetti in un determinato ordine logico. Nel caso delle altre scelte esemplificate, possiamo pensare che ciascun indice in T specifichi un determinato istante di tempo: parleremo di *tempo discreto* quando utilizziamo i numeri relativi per indicare i vari istanti e parleremo di *tempo continuo* quando utilizziamo i numeri reali.

2.2 Sistemi dinamici

EVENTO: tutto ciò che, ad un certo istante, colpisce i nostri sensi; indicheremo con E l'insieme di (tutti i) possibili eventi e con $e \in E$ un singolo evento.

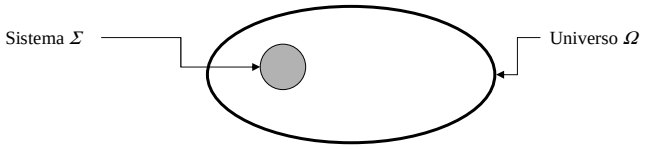
COMPORTAMENTO: successione di eventi ordinata rispetto a T; indicheremo un singolo comportamento con $c = \{e_i\}_{i \in I \subseteq T}$ o anche con $c = e(t)_{t \in I \subseteq T}$, dove I è un intervallo in T.

UNIVERSO DEI COMPORTAMENTI: l'insieme di (tutti i) possibili comportamenti indiciati in un qualsiasi intervallo $I \subseteq T$ fissato (se T rappresenta il tempo, questi saranno i comportamenti possibili nel corso dell'intervallo temporale $I \subseteq T$): indicheremo l'universo con $\Omega = E^I$.

2.3 Sistemi dinamici

2.1 Definizione

Un *sistema dinamico* Σ è definito come un sottoinsieme dell'universo di comportamenti Ω : $\Sigma \subseteq \Omega$.



Dato un sistema $\Sigma \subseteq \Omega$, ogni comportamento $c \in \Sigma$ rappresenta un possibile comportamento del sistema relativamente nell'intervallo $I \subseteq T$ al quale ci si riferisce. Da questo punto di vista, la Definizione 2.1 asserisce che, per quanto riguarda i nostri scopi, un sistema dinamico è un oggetto astratto caratterizzato da un insieme di comportamenti: esattamente quelli, tra tutti i comportamenti presenti nell'universo Ω, che gli sono propri o che, in altri termini, esso è capace di esibire.

2.4 Sistemi dinamici

Notazioni

Quando rappresentiamo un comportamento c come $c = \{e_i\}_{i \in I \subseteq T}$ o come $c = e(t)_{t \in I \subseteq T}$, dove I è un intervallo in T, interpretiamo e come una variabile dipendente (dall'indice i o dall'istante di tempo t), definita sull'intervallo $I \subseteq T$, a valori in E. In questa accezione, e_i o $e(t)$ rappresenta il valore assunto da tale variabile dipendente in corrispondenza dell'indice i o dell'istante t. Diremo che il comportamento c è descritto dall'andamento di e su $I \subseteq T$.

Può accadere che sia utile rappresentare ogni singolo evento $e \in E$ come costituito da un certo numero, ad esempio n, di eventi più semplici. Ciò equivale a pensare la variabile e come una variabile a più dimensioni, che rappresenteremo in forma vettoriale come

$$e = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix} = (e_1, e_2, \ldots, e_n)^T$$ o come un vettore di più variabili. In corrispondenza dell'indice i o dell'istante t si avrà l'evento $e_i = (e_1(i), e_2(i),\ldots, e_n(i))$ o, rispettivamente, $e(t) = (e_1(t), e_2(t),\ldots, e_n(t))$.

2.5 Sistemi e modelli

Possiamo pensare che ogni sistema dinamico $\Sigma \subseteq \Omega$, che come abbiamo detto è un oggetto astratto, non sia altro che il modo nel quale un oggetto concreto o un fenomeno (sistema fisico) S si manifesta ai nostri sensi. Così facendo, associamo in pratica il sistema Σ ad S.

Ogni volta che associamo ad un oggetto concreto o a un fenomeno S un sistema dinamico astratto Σ, cioè un insieme di comportamenti scelti in un universo Ω, diciamo che utilizziamo il sistema Σ come un modello per descrivere l'oggetto S o un modello di S.

Per uno stesso oggetto S si possono avere diversi modelli, ciascuno caratterizzata da un diverso contenuto di informazione. Diremo, in questo caso, che ciascun modello è relativo ad un diverso *livello di rappresentazione* e, di norma, intenderemo come più alto il livello di rappresentazione associato al modello che contiene la quantità maggiore di informazione.

2.6 Sistemi dinamici/Esempi

Per illustrare il concetto di sistema dinamico tramite esempi, è opportuno fare riferimento a situazioni concrete, nelle quali una o più entità fisiche concorrono a generare i comportamenti di nostro interesse. Ciò che occorre comprendere è che il punto di vista che vogliamo sviluppare pone l'accento sul sistema visto come insieme astratto di comportamenti e non come oggetto fisico.

1.2 Esempio

Consideriamo un semaforo (o un insieme di semafori sincronizzati tra loro). Dal punto di vista di un automobilista, ciò che interessa nel comportamento del semaforo è l'alternarsi di luci di colore verde (V), giallo (G) e rosso (R) su una faccia del semaforo. Indichiamo rispettivamente con V, G, R, VG, VR, GR, VGR e 0 (= spento) gli eventi relativi ad ogni possibile combinazione di luci.

E = {V, G, R, VG, VR, GR, VGR, 0}

Ω = {insieme di tutte le possibili sequenze di elementi di E }

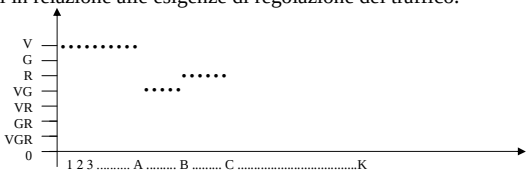
Nell'ipotesi che il semaforo funzioni correttamente, un suo modello $\Sigma_{semaforo} \subseteq \Omega$ è facilmente caratterizzabile come un sottoinsieme di sequenze (comportamenti) di U, (quali esattamente ?).

2.7 Sistemi dinamici/Esempi

1.3 Esempio

Consideriamo ancora il semaforo dell'esempio precedente, questa volta dal punto di vista di un tecnico incaricato di regolare i flussi di traffico all'incrocio dove è sistemato il semaforo.

A differenza di quanto supposto per l'automobilista del precedente Esempio, è da presumere che al tecnico interessi, oltre al semplice alternarsi di luci di vario colore, anche la tempistica con la quale tutto ciò avviene. Indichiamo ancora con E = {V, G, R, VG, VR, GR, VGR, 0} l'insieme delle possibili combinazioni di luce su una faccia (eventi) e supponiamo che, per quanto interessa al tecnico in questa situazione, il tempo possa essere scandito in secondi. Quindi, scelto $T = \mathbf{Z}$ (insieme dei numeri relativi) per rappresentare il tempo discreto, l'universo Ω dei possibili comportamenti relativamente ad un intervallo $I = [0, K] \subseteq T$ risulta costituito dalle funzioni f(t): $I \rightarrow E$. Sarà compito del tecnico predisporre il funzionamento del semaforo in modo tale che $\Sigma_{semaforo}$ contenga solo i comportamenti che risultano desiderabili in relazione alle esigenze di regolazione del traffico.



(Quale dei due modelli del semaforo è relativo al livello di rapresentazione più alto? Quale potrebbere essere un modello del semaforo in tempo continuo?

3.1 Rappresentazione di un sistema dinamico

Osserviamo che ci sono vari modi per specificare quali elementi dell'universo Ω facciano parte di Σ, cioè di un dato sistema o modello:

• mediante un elenco o una tabella;

• mediante un insieme di regole linguistiche;

• mediante un insieme di regole matematiche.

In generale, utilizzeremo un formalismo del tipo $\Sigma = \{c \in \Omega$, tali che $P(c) = VERO\} \subseteq \Omega$, dove con $P(c) = VERO$ indichiamo il fatto che il comportamento c soddisfa una certa condizione P. Il modo utilizzato per specificare quali elementi dell'universo Ω facciano parte di Σ da luogo a quella che viene chiamata una rappresentazione di Σ.

3.2 Rappresentazione/Esempi

1.4 Esempio
Si descriva $\Sigma_{semaforo}$ dell'Esempio 1.2 mediante
a) una tabella,
b) un insieme di regole linguistiche.

1.5 Esempio

Si consideri una certa quantità di un gas perfetto contenuto in un volume costante V e se ne indichino con P(t) e T(t) rispettivamente la pressione e la temperatura all'istante t. La Legge dei Gas Perfetti afferma che vale la relazione PV = RT, dove R è una costante. Si espliciti, mediante la legge matematica ricordata, un modello che descriva in tempo continuo i possibili comportamenti, in termini di pressione e temperatura, di tale quantità di gas.

3.3 Rappresentazione/Esempi

1.6 Esempio

Consideriamo il sistema fisico costituito dal Sole e dalla Terra che ruota attorno ad esso. Keplero fornì un modello del comportamento di tale sistema affermando che il moto della Terra rispetta le seguenti tre leggi

a) la Terra si muove su un'orbita ellittica avente il Sole in uno dei fuochi;

b) la congiungente Terra-Sole spazza aree uguali in tempi uguali;

c) il quadrato del periodo di rivoluzione è proporzionale al cubo dell'asse maggiore dell'ellisse.

Come possiamo ricondurre la descrizione del comportamento data da Keplero ad un modello del tipo introdotto nella Definizione 1.1?

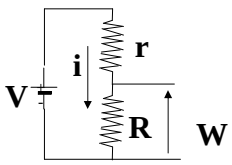
1.7 Osservazione

Si osservi che il modello sviluppato da Keplero illustra in maniera esplicita come la Terra si muove introno al Sole, ma non fornisce alcuna spiegazione del perché ciò si verifichi. Dovette passare molto tempo dopo l'introduzione di tale modello prima che Newton, scoprendo la Legge di Attrazione Gravitazionale, fornisse una valida spiegazione al comportamento della terra nel suo moto attorno al Sole. Questo esempio illustra il fatto che il contenuto informativo di un modello descrive le modalità con le quali si attua un determinato fenomeno, ma non necessariamente le cause che generano il fenomeno.

3.4 Rappresentazione/Esempi

1.8 Esempio

Si consideri il dispositivo elettrico denominato Partitore di Tensione illustrato nello schema sottostante:



Supponendo costanti i valori delle resistenze **R** e **r**, le grandezze fisiche (variabili) che caratterizzano il comportamento del dispositivo, cioè le tensioni **V** e **W** e la corrente **i**, sono legate dalle relazioni **W = VR/(R+r)** e **i = V/(R+r)**.
Un modello $\sum_{\text{partitore}}$ sarà dato nell'universo Ω = {f(t): $\Re \rightarrow \Re^3$; f(t) continua} da $\sum_{\text{partitore}}$ = {f(t) $\in \Omega$, f(t) = (V(t), V(t)**R/(R+r)**, V(t)**/(R+r)**)}.

(Qual'è l'insieme degli eventi e cosa significano?)

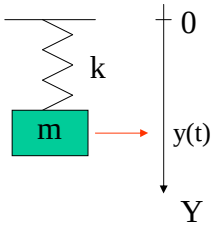
(Quale sarà un modello se immaginiamo di poter variare, sempre in modo continuo, il valore della resistenza r ?)

3.5 Rappresentazione/Esempi

1.9 Esempio

Si consideri il sistema meccanico costituito dalla massa m sospesa ad una molla avente costante elastica k.

Indicando con y(t) la posizione della massa e trascurando la presenza della forza di gravità o di eventuali attriti, è facile mostrare che y(t) verifica l'equazione differenziale lineare del secondo ordine



$$\ddot{y}(t) = -\frac{k}{m} y(t)$$

(Descrivere in dettaglio l'insieme degli eventi e l'universo.)

4.1 Struttura di un sistema

Come detto a suo tempo, i comportamenti di un dato sistema $\sum$ sono di solito descritti dall'andamento di un certo numero di variabili e_i , che sono le componenti di una variabile a più dimensioni $e = (e_1, e_2, \ldots, e_n)^T$.

Tali variabili possono, in molti casi, venire classificate in base al ruolo che assegnamo loro nella rappresentazione di $\sum$. La presenza di una tale classificazione conferisce struttura al sistema $\sum$.

4.2 Struttura/Ingressi e uscite

Supponiamo di prendere in considerazione un sistema fisico S con il quale sia possibile interagire attraverso uno scambio di stimoli (cioè di segnali di qualche tipo inviati dall'ambiente esterno al sistema) e di risposte (cioè di segnali inviati dal sistema all'ambiente esterno, in seguito all'elaborazione degli stimoli ricevuti) (esempi?).

In un qualsiasi modello Σ di S, i comportamenti saranno rappresentati mediante l'andamento di un certo numero di variabili, alcune delle quali descriveranno i valori assunti dai segnali di stimolo e altre descriveranno i valori assunti dai segnali di risposta (oltre ad eventuali variabili che non appartengono a nessuno di questi due gruppi). Le prime verranno denominate variabili di ingresso, o ingressi o input al sistema $\sum$ e le seconde verranno denominate uscite, o risposte o output del sistema $\sum$.

4.3 Struttura/Ingressi e uscite

Notazione

Se è possibile decomporre ciascun evento e di un insieme E in una parte interpretabile come uno stimolo esterno (ingresso) e una parte interpretabile come una risposta (uscita), indicheremo il singolo evento con $e = (u, y)$ e porremo $E = U \times Y$. L'universo dei comportamenti relativo ad T, in questo caso, sarà denotato con $\Omega = (U \times Y)^T = U^T \times Y^T$ ed il singolo comportamento con $c = \{u_i, y_i\}_{i \in I \subseteq T}$ o anche con $c = (u(t), y(t))_{t \in I \subseteq T}$.

2.1 Definizione

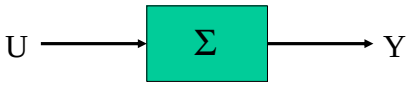
Un sistema $\Sigma \subseteq U^T \times Y^T$, nella cui rappresentazione si possono individuare ingressi e di uscite, si dice *sistema ingresso/uscita* o *sistema orientato*.

2.2 Esempio

Un esempio è dato dal partitore di tensione dell'Esempio 1.8, interpretando la tensione V come l'ingresso al sistema e la corrente i e la tensione W come le risposte.

4.4 Struttura/Sistema orientato

Un sistema orientato $\Sigma \subseteq U^I \times Y^I$ si rappresenta schematicamente nel modo seguente



Osservazione

Si noti che nella descrizione di un evento *e* possono essere presenti variabili che non sono interpretabili come ingressi o uscite (si rivedano gli Esempi 1.8 e 1.9). Tali variabili, il cui ruolo è molto spesso quello di rendere quanto più semplice e maneggevole possibile la rappresentazione del sistema in oggetto, verranno genericamente dette variabili ausiliarie.

4.5 Struttura/Esempi

Esempio

Consideriamo una popolazione composta al tempo t da $x_1(t)$ individui, una parte dei quali (diciamo 1/k) giungono a maturità e si riproducono generando una prole di un individuo ciascuno all'istante t+1. Supponiamo che ad ogni istante t si possano immettere (o prelevare) u(t) individui e indichiamo con m la massa (media) di ciascun individuo. Tale popolazione costituitsce un sistema che evolve a tempo discreto in cui u(t) e $y(t) = m\, x_1(t)$ possono essere pensate come ingresso e uscita.

Per dare una semplice descrizione del comportamento del sistema, conviene introdurre una variabile $x_2(t)$ che rappresenti il numero di individui che nascono al tempo t. Le regole enunciate sopra, quindi, si esplicitano come segue:

$$\begin{cases} x_2(t+1) = \frac{1}{k} x_1(t) \\ x_1(t+1) = x_1(t) + x_2(t) + u(t) \\ y(t) = m x_1(t) \end{cases}$$

Le variabili x_i costituiscono un esempio di variabili ausiliarie.

4.7 Struttura/Stato

Come abbiamo già osservato, è possibile che nella descrizione del comportamento di un sistema orientato Σ compaiano variabili ausiliarie che non sono interpretabili come rappresentative di ingressi o di uscite.

In alcuni casi significativi, alcune di queste variabili assumono un ruolo importante, descritto dalla definizione seguente:

2.1 Definizione

Dato un sistema orientato

$$\Sigma = \{(u(t),y(t),x(t)),\ P(u(t),y(t),x(t)) = VERO\} \subseteq \Omega = U^I \times Y^I \times X^I$$

con u variabile di ingresso e y variabile di uscita, la variabile ausiliaria x è detta *stato* di Σ se, dati due comportamenti

$$c(t) = (u(t),y(t),x(t)) \in \Sigma \qquad c'(t) = (u'(t),y'(t),x'(t)) \in \Sigma$$

per ogni $t_0 \in T$, si ha che

a) $x(t_0) = x'(t_0)$ implica che il comportamento $c''(t)$ definito da $c''(t) = c(t)$ per $t < t_0$ e $c''(t) = c'(t)$ per $t \geq t_0$ appartiene a Σ;

b) $x(t_0) = x'(t_0)$ e $u(t) = u'(t)$ per $t \geq t_0$, implica $c(t) = c'(t)$ per $t > t_0$.

4.7bis Struttura/Stato

Osservazione

Dati due comportamenti $c(t) = (u(t),y(t),x(t))$ e $c'(t) = (u'(t),y'(t),x'(t))$, il comportamento $c''(t)$ definito da

$$c''(t) = c(t) \text{ per } t < t_0$$
$$c''(t) = c'(t) \text{ per } t \geq t_0$$

viene detto concatenazione in t_0 di $c(t)$ e $c'(t)$.

La definizione di stato che abbiamo dato nella diapositiva precedente dice, in sostanza, che la variabile $x(t)$ è uno stato se la concatenazione in t_0 di due comportamenti $c(t)$ e $c'(t)$ di Σ, per i quali si abbia $x(t_0) = x'(t_0)$, da luogo ad un ulteriore comportamento di Σ.

4.8 Struttura/Stato

La nozione di stato che abbiamo introdotto si caratterizza in particolare per la seguente importante proprietà:

4.1 Proposizione (Proprietà dello stato)

Dato un sistema orientato

$$\Sigma = \{(u(t),y(t),x(t)),\ P(u(t),y(t),x(t)) = VERO\} \subseteq \Omega = U^I \times Y^I \times X^I$$

con ingresso u, uscita y e stato x, si ha che, in ogni comportamento $c(t) = (u(t),y(t),x(t) \in \Sigma$ e per ogni $t_0 \in T$, l'uscita $y(t)$ per $t > t_0$ è determinata da $x(t_0)$ e $u(t)$ per $t \geq t_0$.

Dimostrazione. Viene lasciata per esercizio.

Se, nella rappresentazione di Σ considerata sopra, T rappresenta il tempo, la Proposizione 4.1 asserisce che, posto t_0 = *istante presente*, l'uscita $y(t)$ nel futuro (cioè per $t > t_0$) è determinata dal valore dello stato all'istante presente $x(t_0)$ e dal valore dell'ingresso $u(t)$ a partire dall'istante presente.

4.9 Esempio

Si consideri il sistema meccanico costituito dalla massa m sospesa ad una molla avente costante elastica k. Come già visto, indicando con y(t) la posizione della massa e trascurando la presenza della forza di gravità o di eventuali attriti, si ha che y(t) verifica l'equazione differenziale lineare del secondo ordine

$$\ddot{y}(t) = -\frac{k}{m} y(t)$$

Per risolvere l'equazione, cioè predire il movimento, occorre conoscere la posizione e la velocità della massa all'istante 0 (condizioni iniziali).

Introducendo le variabili ausiliarie x_1= y e x_2= y, possiamo riscrivere l'equazione che descrive il moto come un sistema di due equazioni differenziali lineari del primo ordine

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = -\frac{k}{m} x_1(t) \end{cases}$$

Si verifichi che le variabili ausiliarie x_1= y e x_2 rappresentano lo stato del sistema.

4.10 Struttura/Stato

Lo stato rappresenta una misura dell'energia (o dell'informazione) contenuta nel sistema.

Lo stato può essere visto come rappresentativo della memoria del sistema.

5.1 Classi di modelli

Da quanto abbiamo detto fino ad ora si comprende che per assegnare un modello di un sistemo fisico S occorre fissare, attraverso una scelta opportuna dell'insieme degli eventi E e dell'insieme degli indici T, un universo Ω, e quindi specificare i possibili comportamenti di S in Ω.

A seconda dei casi, l'insieme degli eventi E può essere scelto come

• un insieme *discreto* (intuitivamente rappresentabile come un insieme di punti isolati gli uni dagli altri in uno spazio $\Re^N$),

• un insieme *continuo* (intuitivamente rappresentabile come l' insieme dei punti che riempiono un aperto di $\Re^N$).

5.2 Classi di modelli

L'insieme degli indici verrà scelto in base all'interesse, alla possibilità e/o all'esigenza di descrivere i comportamenti di S in relazione allo scorrere del tempo o meno. In particolare, se scegliamo *T* come insieme dei tempi, possiamo specificare cosa accade in ogni istante. In alternativa, cioè se *T* non è visto come un insieme dei tempi, possiamo specificare solamente l'ordine in cui accadono gli eventi.

La prima soluzione è adatta al caso di sistemi per i quali la successione di eventi che descrive un comportamento può essere temporizzata, cioè vista come una funzione del tempo: parleremo in tal caso di sistemi *time-driven*.

La seconda soluzione è adatta al caso di sistemi per i quali la successione di eventi che descrive un comportamento non può essere temporizzata, in quanto il verificarsi di determinati eventi è indipendente dal tempo: parleremo in tal caso di sistemi *event-driven*.

5.3 Classi di modelli

Una volta fissato l'universo Ω, occorre, come abbiamo detto, assegnare un modo per determinare i possibili comportamenti di S in Ω.

A questo scopo, ci si avvale, in generale:

• di informazione raccolta osservando il comportamento del sistema fisico in varie (o in tutte le) situazioni possibili (si pensi all'esempio del semaforo o del calendario),

• di informazione dedotta dalla conoscenza di parametri strutturali e ipotesi di leggi fisiche o regole di altro tipo che determinano il comportamento del sistema stesso (si pensi all'esempio del sistema meccanico Massa/Molla).

5.4 Classi di modelli

L'informazione del primo tipo può rivestire un ruolo preminente nei casi in cui i comportamenti del sistema presentino una limitata variabilità (gli eventi di interesse sono pochi o il comportamento è ripetitivo) e si possa ricorrere ad un insieme degli eventi di tipo discreto. Spesso, in queste situazioni, si può giungere a costruire modelli la cui rappresentazione è data da un elenco o una tabella che enumerano (eventualmente mediante convenzioni stabilite) tutti i comportamenti possibili.

In casi più complessi, sarà l'informazione del secondo tipo a risultare più significativa. In queste situazioni, è utile di solito ricorrere a regole matematiche di vario tipo per specificare quali siano i comportamenti possibili.

5.5 Classi di modelli

Nel seguito, focalizzeremo la nostra attenzione su sistemi dinamici che, in base alla loro rappresentazione, possono essere raggruppati in due classi, rispondenti, rispettivamente, alle esigenze di modellazione espresse sopra.

Si tratta dei cosiddetti Sistemi Dinamici ad Eventi Discreti, o DEDS, che sono particolarmente efficaci per modellare i comportamenti nel caso di insiemi di eventi di tipo discreto, e dei cosiddetti SISTEMI DINAMICI LINEARI, che sono utilizzabili per modellare un gran numero di fenomeni interessanti in vari settori.

6.1 DEDS e Automi

Definizione

Sia E un insieme di eventi di tipo discreto e si consideri l'insieme ordinato $T = \{1,2,3,\ldots\}$. Un sistema dinamico $\Sigma \subseteq \Omega = E^T$ viene detto *sistema dinamico ad eventi discreti* o *DEDS (Discrete Events Dynamical System).*

Osserviamo che nella Definizione precedente T può essere visto come insieme dei tempi, misurati in maniera discreta a partire dall'istante 1, o, in alternativa, come un insieme di indici che marcano una sequenza logica. Nel primo caso il DEDS sarà visto come un sistema time-driven a tempo discreto, mentre nel secondo caso sarà visto come un sistema event-driven.

6.2 DEDS e Automi

La rappresentazione più efficace di un DEDS (ricordiamo che una rappresentazione di un sistema dinamico Σ non è altro che un modo per specificare quali comportamenti dell'universo Ω appartengano a Σ), in particolare nel caso in cui E sia un insieme finito, si ottiene impiegando strutture matematiche che vengono dette *automi.*

In termini generali (daremo più avanti una definizione formale), un *automa* G è un oggetto rappresentato da un *grafo*, cioè da

- un insieme di *punti* (detti *nodi* del grafo), indicati da lettere maiuscole,
- un insieme di terne ordinate (X,a,Y), il cui primo e il terzo elemento sono nodi, dette *archi* del grafo,

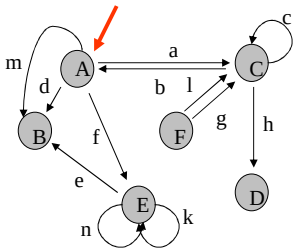
nel quale è stato scelto un nodo, detto *nodo iniziale*.

6.3 DEDS e Automi/Esempio

Esempio

Un automa a sei nodi G.

Si osservi che, nella rappresentazione grafica, si sono indicati gli archi con il solo elemento centrale della terna (es. (A,a,C) con a). Il punto iniziale A viene evidenziato con una freccia.



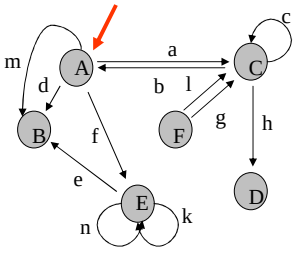
Definizione

In un automa G, due archi (X,a,Y) e (U,b,V) si dicono consecutivi se Y = U.

Un insieme di archi $\{(X_1,a_1,Y_1), (X_2, a_2,Y_2), \ldots, (X_n, a_n,Y_n)\}$ consecutivi si dice cammino; X_1 è detto punto iniziale del cammino e Y_n è detto punto finale.

6.4 DEDS e Automi

Se in un automa G pensiamo alle lettere che etichettano gli archi come alle *lettere* di un *alfabeto*, possiamo interpretare l'automa come un dispositivo per formare le *parole* di un *linguaggio*, associando ad ogni possibile cammino che parte dal punto iniziale la parola formata dalla sequenza ordinata di lettere che marcano gli archi costituenti il cammino.



Ad esempio, per l'automa G illustrato a fianco, avremo, nel linguaggio, le parole a; accb; fkne; ……..

Definizione

L'insieme L delle parole costruito a partire da un automa G come descritto sopra si chiama *linguaggio generato da G.*

6.5 DEDS e Automi

Dato un automa G, associamo ad ognuna delle lettere utilizzate per indicare gli archi di G un evento dell'insieme di eventi *E*. Questo ci permette di interpretare ogni parola del linguaggio L generato da G come una successione di eventi, cioè come un comportamento nell'universo $\Omega = E^T$, con $T = \{1,2,3,\ldots\}$.

Diremo che l'automa G è una rappresentazione del DEDS Σ se $L = \Sigma \subseteq \Omega$.

Nel seguito, sulla base di quanto convenuto sopra, utilizzeremo automi e linguaggi generati da essi per maneggiare i DEDS e studiarne le proprietà.

6.6 DEDS e Automi/Modelli

In particolare, per costruire il modello Σ di un dato sistema S utilizzando un opportuno DEDS, detto *E* l'insieme degli eventi di interesse, occorrerà individuare tutti i possibili comportamenti di S e, successivamente, costruire un automa G tale che il linguaggio L da esso generato mediante le lettere dell'alfabeto E contenga tutti i comportamenti di S.

7.1 Sistemi lineari ARX

Definizione

Sia E l'insieme di eventi costituito dallo spazio vettoriale $\Re \times \Re$ e sia $T = \mathbf{Z}$ (insieme dei numeri relativi). Un sistema dinamico orientato $\Sigma \subseteq \Omega = E^T$ del tipo

$\Sigma = \{(y(t), u(t)) \in \Re \times \Re$ *tali che*

$$y(t) = a_1 y(t-1) + a_2 y(t-2) + \ldots + a_{na} y(t-na) + b_1 u(t-1) + b_2 u(t-2) + \ldots + b_{nb} u(t-nb) \quad (6.1)$$

per $t \in I \subseteq T$ *}*

viene detto *sistema dinamico lineare (di tipo) ARX*.

Nota

Il nome ARX origina dalla contrazione di "Auto Regressivo con componente Esogena (Exogenous, in inglese)": nell'equazione 6.1 si chiama componente Auto Regressiva quella costituita dai termini in y e componente Esogena quella costituita dai termini in u.

7.2 Sistemi lineari ARX

Ricordiamo che schematicamente il sistema orientato Σ si rappresenta come

In termini di funzionamento, cioè di azione svolta, l'interpretazione che diamo di Σ è quella di un dispositivo che, stimolato, dal segnale di ingresso $u(t)$ fornisce in risposta il segnale $y(t)$.

L'equazione 6.1

$$y(t) = a_1 y(t-1) + a_2 y(t-2) + \ldots + a_{na} y(t-na) + b_1 u(t-1) + b_2 u(t-2) + \ldots + b_{nb} u(t-nb) \quad (6.1)$$

asserisce che, in ogni comportamento di Σ, il valore dell'uscita y ad ogni istante t, cioè $y(t)$, è una combinazione lineare dei valori dell'uscita stessa negli na istanti che precedono t e dei valori dell'ingresso negli nb istanti che precedono t.

7.3 Sistemi lineari ARX

Se consideriamo il funzionamento del sistema ARX Σ sull'intervallo $I = [0,+\infty) \subseteq T$, l'utilizzo dell'equazione 6.1

$$y(t) = a_1y(t-1)+a_2y(t-2)+\ldots+a_{na}y(t-na)+b_1u(t-1)+b_2u(t-2)+\ldots+b_{nb}u(t-nb) \quad (6.1)$$

richiede di specificare alcuni valori di y e di u in istanti precedenti 0.

In tali situazioni sarà sempre sottointeso che

$$u(t) = 0 \text{ per ogni } t < 0 \qquad (6.2)$$

(ciò significa soltanto che non è stato fornito alcuno stimolo significativo al sistema prima dell'istante di inizio del funzionamento).

I valori y(-1), y(-2), …, y(-na) verranno detti condizioni iniziali all'istante 0 e rappresentano le condizioni nelle quali si trova il sistema in conseguenza di ciò che è accaduto prima dell'istante 0.

7.4 Sistemi lineari ARX

Nel seguito supporremo sempre che sia soddisfatta la seguente

Assunzione

Nell'equazione 6.1 si ha na ≥ nb.

L'indice na misura la memoria, e quindi la complessità, del sistema Σ descritto da 6.1.

Definizione

L'indice na viene detto ordine del sistema Σ descritto da 6.1.

Osserviamo che il dato y(-1), y(-2), …, y(-na) ha proprietà analoghe a quelle indicate per lo stato nel paragrafo 4.7. In particolare osserviamo che scelti arbitrariamente y(-1), y(-2), …, y(-na) e $u(t)$ per $t \geq 0$, si ha in corrispondenza di questi un unico comportamento di Σ definito da 6.1.

Diremo pertanto che il dato y(-1), y(-2), …, y(-na) descrive lo stato del sistema Σ all'istante $t = 0$.

7.5 Sistemi lineari ARX

Evoluzione libera e forzata

Osserviamo che, dato un segnale di ingresso u(t) = f(t), $t \in [0,+\infty)$, e un insieme di condizioni iniziali $y(-1) = y_1, \ldots y(-na) = y_{na}$, grazie alla struttura dell'equazione 6.1, possiamo suddividere il comportamento in uscita y(t), $t \in [0,+\infty)$, di Σ in due parti:

- una parte indicata con $y_L(t)$ e detta *risposta libera* calcolata mediante 6.1 ponendo

$$y(-1) = y_1, \ldots y(-na) = y_{na} \text{ e } u(t) = 0 \text{ , } t \in [0,+\infty),$$

($y_L(t)$ descrive il comportamento in uscita di Σ quando l'ingresso è nullo)

- una parte indicata con $y_F(t)$ e detta *risposta forzata* calcolata mediante 6.1 ponendo

$$y(-1) = 0, \ldots y(-na) = 0 \text{ e } u(t) = f(t) \text{ , } t \in [0,+\infty).$$

($y_F(t)$ descrive il comportamento in uscita di Σ quando la condizione iniziale è nulla)

Infatti, non è difficile verificare (come si fa?) che si ha

$$y(t) = y_L(t) + y_F(t)$$

7.6 Sistemi lineari ARX

Proposizione

Dato un sistema Σ definito da 6.1 e fissata la condizione iniziale y(-1) = y(-2) = … = y(-na) = 0, dette rispettivamente y'(t) e y"(t) le risposte relative a due qualsiasi ingressi u'(t) e u"(t), si ha per ogni $a,b \in \Re$ che la risposta y(t) relativa all'ingresso u(t) = au'(t) + bu"(t) verifica y(t) = ay'(t) + by"(t)

Dimostrazione. Lasciata per esercizio.

Quanto asserito dalla proposizione precedente rende ragione del termine "lineare" nella caratterizzazione dei sistemi che stiamo considerando: la risposta relativa alla condizione iniziale nulla e ad una combinazione lineare di ingressi è data dalla combinazione lineare, con gli stessi coefficienti, delle risposte relative ai singoli ingressi. In altri termini, la risposta forzata è lineare rispetto all'ingresso u(t).

(Si provi che la risposta libera è lineare rispetto alla condizione iniziale.)

7.7 Sistemi lineari ARX/Modelli

Per costruire un sistema dinamico Σ di tipo ARX che modelli un dato sistema fisico S, si può operare come segue:

a) si esegue un esperimento di funzionamento sul sistema S nell'intervallo [0,N-1], consistente nel fornire ad esso un ingresso $u(t)$ e misurare la corrispondente risposta $y(t)$, in modo da ricavare il comportamento $c(t) = (u(t), y(t))$;

b) si fissa il grado di complessità del modello che si intende costuire, cioè si sceglie il valore dell'indice na che misura la lunghezza della parte AR, e si sceglie nb ≤ na;

c) si considera per ogni t la quantità $e(t) = y(t) - a_1y(t-1) - \dots - a_{na}y(t-na) - b_1u(t-1) - \dots - b_{nb}u(t-nb)$, nella quale i valori di y e di u sono quelli che costituiscono il comportamento $c(t)$ e $\theta = (a_1, a_2, \dots, a_{na}, b_1, b_2, \dots, b_{nb})$ è un vettore di incognite;

d) si considera la funzione $J(\theta)$ definita da $J(\theta) = (1/N)\Sigma_t e^2(t)$ e si cerca il suo punto di minimo θmin .

7.8 Sistemi lineari ARX/Modelli

Osserviamo che $J(\theta min) = 0$ significa che il comportamento $c(t)$ appartiene al sistema ARX Σ definito dai valori $\theta min = (a_1, a_2, \dots, a_{na}, b_1, b_2, \dots, b_{nb})$. In altre parole, se $J(\theta min) = 0$, Σ è un modello di S, almeno per quanto riguarda ciò che possiamo dedurre su S dall'esito del nostro esperimento, cioè dall'informazione che $c(t)$ è un suo comportamento. Questa informazione può essere sufficiente a caratterizzare S se l'esperimento che ci ha condotto a determinare $c(t)$ è stato sufficientemente completo e esaustivo (in teoria, dovremmo dare ad S tutti i possibili ingressi e condurre l'osservazione delle uscite su $[0, +\infty)$).

Nella realtà, non possiamo aspettarci che $J(\theta min) = 0$, ma si avrà $J(\theta min) = e > 0$. La quantità e prende il nome di errore (quadratico, medio) di modello e indica quanto fedelmente il sistema Σ modella le caratteristiche di S, almeno in riferimento all'esito del nostro esperimento.

8.1 Problematiche di controllo

Come abbiamo già accennato, disporre di un modello per un dato fenomeno, ci consente di predire il l'evoluzione futura del fenomeno o, più in generale, di analizzarne le caratteristiche.

Oltre a predire il comportamento di un fenomeno, siamo spesso interessati a governarlo, cioè a far si che in una data situazione esso esibisca, tra tutti i comportamenti possibili, solo quel comportamento (o eventualmente quei comportamenti) che soddisfano a determinate specifiche.

L'insieme delle azioni che ci consentono di governare un fenomeno prende il nome di azione di controllo. Se disponiamo di un modello del fenomeno, possiamo utilizzarlo per progettare e definire una azione di controllo adeguata allo scopo prefisso, cioè all'ottenimento di un comportamento che soddisfi le specifiche.

8.2 Problematiche di controllo

Esempio

Nel caso di un DEDS Σ, le specifiche riguardano di solito l'esclusione di determinati comportamenti che, per qualche ragione, vengono ritenuti indesiderabili.

L'azione di controllo, in tal caso, può esplicarsi attraverso l'inibizione dei comportamenti indesiderati.

Esempio

Nel caso di un sistema ARX Σ, le specifiche riguardano di solito le caratteristiche dell'uscita (ad esempio si vuole un ben preciso segnale in uscita, oppure si vuole che tutte le possibile uscite mostrino un andamento particolare).

L'azione di controllo, in tal caso, può esplicarsi attraverso la scelta di opportuni ingressi, che forzano il sistema a rispondere nel modo desiderato.

9.1 Architetture di controllo

L'azione di controllo può venire esercitata da un operatore umano o da un dispositivo che prende il nome di controllore.

Quest'ultima è la situazione di interesse fondamentale per l'Automatica: quando l'azione di controllo viene esercitata da un dispositivo che non richiede l'intervento di un operatore umano, parliamo in effetti di controllo automatico.

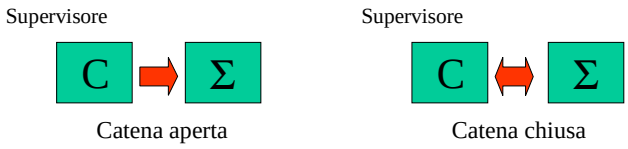
Nell'architettura di una struttura costituita da un sistema (controllato) Σ e un controllore C, possiamo distinguere due situazioni di interesse:

• nella prima situazione il controllore svolge la sua azione di controllo sulla base di informazioni e conoscenze possedute a priori e non è in grado di monitorare il suo effetto sul sistema controllato; parleremo in questo caso di controllo in catena aperta;

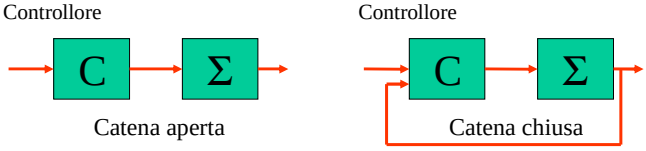
• nella seconda situazione il controllore svolge la sua azione di controllo sulla base di informazioni relative al contemporaneo funzionamento del sistema ed è quindi in grado di monitorare il suo effetto sul sistema controllato; parleremo in questo caso di controllo in catena chiusa.

9.2 Architetture di controllo

Architetture di controllo: DEDS



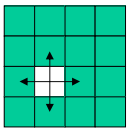
Architetture di controllo: sistemi lineari ARX



9.3 Architetture di controllo/Esempi

Esempio

Il gioco del 15:



si descriva con un DEDS (rappresentandolo con un automa) il comportamento di un giocatore.

(Suggerimento: indichiamo con A, B, D, S i possibili eventi costituiti dallo spostamento della casella vuota in Alto, in Basso, a Destra, a Sinistra e cominciamo a descrivere uno dei possibili comportamenti come cammino di un grafo che costruianmo via via. E' facilmente comprensibile che i nodi del grafo rappresenteranno la posizione della casella vuota, e pertanto possono essere indicati con (i,j), i = 1,2,3,4, j = 1,2,3,4.)

Si costruisca un supervisore che, tramite l'inibizione di alcuni eventi, lavorando in catena aperta, forzi il giocatore a portare la casella vuota in (1,1).

Si costruisca un supervisore che, tramite l'inibizione di alcuni eventi, lavorando in catena chiusa, forzi il giocatore a portare la casella vuota in (2,2).

9.2 Architetture di controllo/Esempi

Esempio

Dato il sistema lineare ARX di ordina 1 descritto da

$\Sigma 1 \equiv y(t) = 1/2y(t-1) + u(t-1)$

si descriva l'azione di un controllore che, lavorando in catena aperta, forza la risposta y(t) del sistema, qualunque sia la condizione iniziale, a tendere asintoticamente al valore 2.

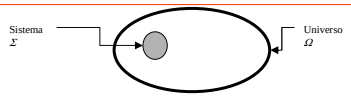
Si provi ad ottenere lo stesso risultato per il sistema

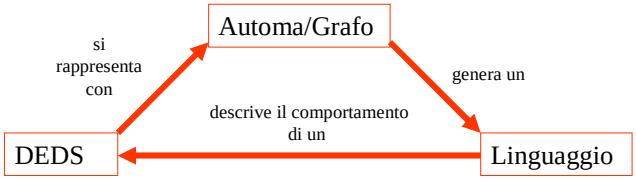
$\Sigma 2 \equiv y(t) = y(t-1) + u(t-1)$.

Si costruisca un controllore che lavorando in catena chiusa forza la risposta ad assumere il valore costante 2 nel caso di entrambi i sistemi.

1.1 DEDS/Automi/Linguaggi

2.1 Definizione Un *sistema dinamico* Σ è definito come un sottoinsieme dell'universo di comportamenti Ω: $\Sigma \subseteq \Omega$.





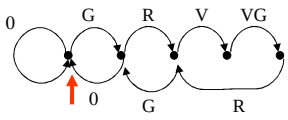
1.2 DEDS/Automi/Linguaggi

Definizione Sia *E* un insieme di eventi di tipo discreto e si consideri l'insieme ordinato $T = \{1,2,3,\ldots\}$. Un sistema dinamico $\Sigma \subseteq \Omega = E^T$ viene detto *sistema dinamico ad eventi discreti* o *DEDS (Discrete Events Dynamical System).*

Σ E = {insieme degli eventi} {insieme dei comportamenti}

$\Sigma_{semaforo}$ {0,V,G, VG, R} {(0,VG,R,V,VG,R,...);(0,VG,0);…}

G



{linguaggio generato da G}

E = alfabeto = {0,V,G, VG, R}

1.3 DEDS/Automi/Linguaggi

Rappresentando un DEDS $\Sigma \subseteq \Omega = E^T$ mediante un grafo G, l'insieme dei comportamenti di Σ viene a coincidere con il linguaggio L generato da G su E, visto come alfabeto, cioè con l'insieme delle parole corrispondenti ai cammini percorribili in G.

Diremo che Σ parla il linguaggio L.

Lo studio di un DEDS Σ può essere condotto attraverso lo studio di un opportuno linguaggio.

Obiettivi :

• analisi del linguaggio parlato da un dato DEDS (analisi).

•costruzione di un DEDS Σ che esibisca un dato insieme di comportamenti, cioè parli un dato linguaggio (modellazione, sintesi);

1.4 Linguaggio

L'insieme di eventi E, che assumiamo finito, è visto come un alfabeto.
Una sequenza ordinata di eventi presi da questo alfabeto è una parola o una stringa (stringa di eventi).
Stringa vuota ε è la stringa che non contiene nessun evento.
Lunghezza di una stringa è il numero di eventi contenuti in essa, contati con la rispettiva molteplicità.

Definizione. Un linguaggio definito su E è un insieme di stringhe di lunghezza finita formata di eventi di E.

Esempio: E = {a, b, g}

L1 = {ε, g, abb}

L2 = {tutte le possibili stringhe di lunghezza 3, che iniziano con l'evento a; aaa, aab, aba, abb, aag, aga, agg, abg, agb} (9 stringhe)

L3 = {tutte le possibili stringhe di lunghezza finita che iniziano con a} (∞ stringhe)

1.5 Linguaggio

L'operazione chiave per costruire stringhe è la concatenazione:

se u e v sono due stringhe, la loro concatenazione uv è costituita dalla sequenza di eventi in u seguita dalla sequenza di eventi in v (es.: u = a; v = ab $\rightarrow$ uv = abb).

- $\varepsilon \equiv$ stringa vuota $\equiv$ elemento identità nella concatenazione

- E* = l'insieme di tutte le stringhe di lunghezza finita formate (mediante la concatenazione) da elementi di elementi di E, compresa la stringa vuota

NOTA
- E* è un insieme composto da un'infinità numerabile di elementi
- un qualunque linguaggio su E è un sottoinsieme di E*
- $\varnothing$, E, E* sono linguaggi.

1.6 Terminologia relativa alle stringhe

Definizione. Sia s una stringa appartenente a E*.
Una stringa $u \in E^*$ si dice sottostringa di s se, per qualche stringa v, $w \in E^*$, si ha s = vuw.
Una stringa $u \in E^*$ si dice prefisso di s se, per qualche stringa $v \in E^*$, si ha s = uv.
Una stringa $u \in E^*$ si dice suffisso di s se, per qualche stringa $v \in E^*$, si ha s = vu.

Esempi
- ε è prefisso, suffisso, sottostringa di s, per ogni $s \in E^*$
- s è prefisso, suffisso, sottostringa di se stessa
- a è prefisso di abc, ab è prefisso di abc, c è suffisso di abc

1.7 Operazioni sui linguaggi

CONCATENAZIONE

Siano L_a, $L_b \subseteq E^*$ due linguagi. La concatenazione L_aL_b di L_a con L_b è data da

$$L_a L_b = \{s \in E^* : s = s_a s_b \text{ per quache } s_a \in L_a, s_b \in L_b\}.$$

(Una stringa è in L_a L_b se essa può essere scritta come la concatenazione di una stringa in L_a con una stringa in L_b)

CHIUSURA RISPETTO AL PREFISSO

Sia $L \subseteq E^*$ un linguaggio. La chiusura $\overline{L}$ di L rispetto al prefisso è data da

$$L = \{s \in E^*, \text{ tali che st} \in L \text{ per qualche } t \in E^*)\}$$

($\overline{L}$ è composto dai prefissi di tutte le stringhe in L, ovviamente $L \subseteq \overline{L}$).

Se $L = \overline{L}$, L è detto chiuso rispetto al prefisso.

CHIUSURA DI KLEENE

Sia $L \subseteq E^*$ un linguaggio. La chiusura L^* di Kleene di L è data da

$$L^* = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$$

L'operazione "chiusura di Kleene" (*) è idempotente $(L^*)^* = L^*$

ESEMPI

- E = {a, b, g}; L_4 = {ε, a, abb}; L_5 = {g}
  - Né L_4 né L_5 sono chiusi rispetto al prefisso
  - ✓ L_4L_5 = {g, ag, abbg}; $\overline{L}_4$ = {ε, a, ab. abb}
  - ✓ $\overline{L}_5$ = {ε, g}; $L_4\overline{L}_5$ = {ε, a, , a, abb, g, ag, abbg}
  - ✓ L_5^* = {ε, g, gg, ggg, …}
  - ✓ L_4^* = {ε, a, abb, aa, aabb, abba, abbabb …}
- $\varepsilon \notin \Phi$
- {ε} è un linguaggio non vuoto contenente solo la stringa vuota
- Se L= Φ, $\overline{L}$ = Φ; se $L \neq \Phi \Rightarrow \varepsilon \in \overline{L}$
- $\Phi^* = \{\varepsilon\}$; $\{\varepsilon\}^* = \{\varepsilon\}$

AUTOMI

Come abbiamo già detto a suo tempo, utilizzeremo gli automi per definire e manipolare i linguaggi.

Definizione
Un automa G è un oggetto costituito dai seguenti dati
- un insieme X, detto insieme di stato o degli stati
- un insieme E, detto insieme degli eventi
- una funzione Γ: X $\rightarrow 2^E$, detta funzione (indicatrice) degli eventi attivi
- una funzione f a dominio in X $\times$ E e codominio in X, definita per ogni coppia (x,e) tale che e $\in \Gamma(x)$, detta funzione transizione di stato
- uno stato iniziale $x_0 \in X$
- un sottoinsieme X_m di X, detto sottoinsieme degli stati marcati.

Scriveremo G = (X, E, f, Γ, x_0, X_m) per indicare un automa.

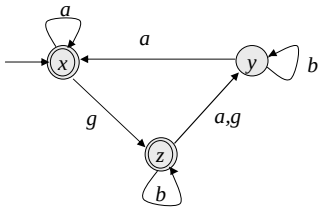
AUTOMI

Nella rappresentazione di un AUTOMA G = (X, E, f, Γ, x_0, X_m) mediante un grafo si assegna:

- un nodo del grafo per ogni elemento di X, cioè per ogni stato dell'automa;
- un arco (x, e, y) per ogni terna (x, e, y) tale che x, y $\in$ X e si abbia f(x,e)=y (in particolare questo implica che e $\in \Gamma(x)$).

Si indica inoltre con una freccia il nodo corrispondente a x_0 e si distinguono gli stati marcati mediante un doppio cerchio.

Esempio
X={x,y,z}
E = {a,b,g}
f(x,a)=x; f(x,g)=z; f(y,a)=x; f(y,b)=y;
f(z,b)=z; f(z,a)= f(z,g)= y
Si noti che Γ è implicitamente definita da f.



ULTERIORI NOTAZIONI

Senza cambiare notazione, indicheremo con f l'estensione della funzione transizione di stato da un sottoinsieme di $X \times E$ ad un sottoinsieme di $X \times E^*$ definita in modo ricorsivo come segue:

- $f(x, \varepsilon) := x$;
- $f(x,se) := f(f(x,s),e)$, per ogni s tale che $f(x,s)$ sia definita per ogni $e \in \Gamma(f(x,s))$.

Esempio

Nel caso dell'esempio precedente si ha, in particolare:

$f(y, \varepsilon) = y$

$f(x, gba) = f(f(x, gb), a) = f(f(f(x, g), b), a) = f(f(z, b), a) = f(z, a)=y$

$f(x, aagb) = z$

$f(z, b^n) = z \quad \forall\, n \geq 0$ (b^n := n consecutivi accadimenti dell'evento b)

LINGUAGGI GENERATI DA AUTOMI

Definizione

Il linguaggio $\Lambda(G)$ generato da un automa $G = (X, E, f, \Gamma, x_0, X_m)$ è definito da

$$\Lambda(G) := \{s \in E^*, \text{ tali che } f(x_0,s) \text{ è definito}\}$$

Definizione

Il linguaggio $\Lambda_m(G)$ marcato da un automa $G = (X, E, f, \Gamma, x_0, X_m)$ è definito da

$$\Lambda_m(G) := \{s \in \Lambda(G) \text{ tali che } f(x_0,s) \in X_m\}$$

OSSERVAZIONI

Il linguaggio $\Lambda(G)$ generato dall'automa G descrive tutti i percorsi orientati percorribili ad iniziare da x_0 sul grafo scelto per rappresentare G e, allo stesso tempo, descrive tutti i comportamenti del DEDS a cui G soggiace.

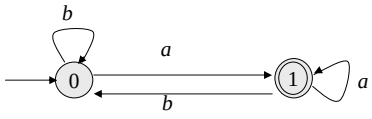
Osserviamo che:

- $\Lambda(G)$ è chiuso rispetto al prefisso;.
- se $f(x_0,s)$ è definita per ogni $s \in E^*$, allora $\Lambda(G) = E^*$;
- $\Lambda_m(G)$ è il sottoinsieme di $\Lambda(G)$ composto solo dalle stringhe s per le quali $f(x_0, s) \in X_m$; in generale, non è chiuso rispetto al prefisso;
- Il linguaggio marcato è detto anche linguaggio riconosciuto dall'automa. Diremo anche che l'automa è un riconoscitore del linguaggio.

ESEMPIO

Dato l'automa $G = (X, E, f, \Gamma, x_0, X_m)$ con

- $X = \{0,1\}$, $x_0 = 0$, $X_m = \{1\}$
- $E = \{a,b\}$
- $f(0,a) = 1$; $f(0,b) = 0$; $f(1,a) = 1$; $f(1,b) = 0$,



si ha

$$\Lambda_m(G) = \{\ a,aa,ba,aaa,aba,baa,bba,...\ \}.$$

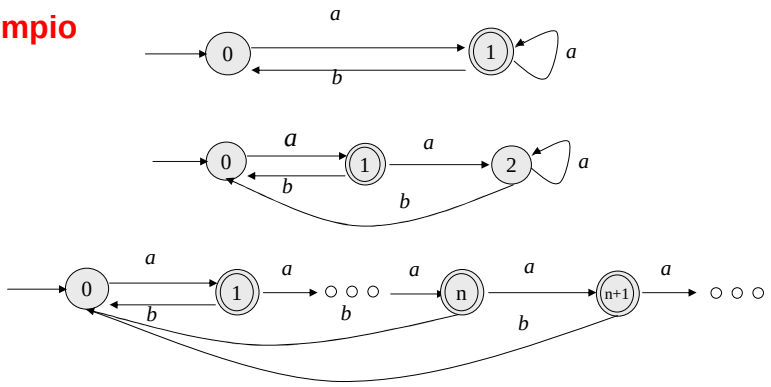
AUTOMI EQUIVALENTI

Definizione

Due automi G_1 e G_2 si dicono equivalenti se generano e marcano gli stessi linguaggi, cioè se

$$\Lambda(G_1) = \Lambda(G_2) \text{ e } \Lambda_m(G_1) = \Lambda_m(G_2).$$

Esempio



ANALISI DELLE PRESTAZIONI DI UN DEDS: PRESENZA DI BLOCCHI

Dopo aver modellato con un DEDS Σ una data macchina, possiamo porci il problema di verificare se tutti i possibili comportamenti della macchina portano all'espletamento dei compiti per i quali essa è impiegata, o se vi sono anche comportamenti (in genere, indesiderati) che non godono di tale proprietà.

Come abbiamo già menzionato, possiamo rappresentare le situazioni di espletamento di un dato compito per mezzo della marcatura di opportuni stati in un automa G che rappresenti Σ (ricordiamo che la scelta degli eventuali stati marcati è una questione relativa alla modellazione dei comportamenti che ci interessano).

Una situazione nella quale sono presenti comportamenti indesiderati, cioè che non portano all'espletamento di alcuno dei compiti previsti, è data dall'esistenza di stati non marcati che, se raggiunti, non possono essere abbandonati o dai quali non è possibile comunque ragiungere alcuno stato marcato.

ANALISI DELLE PRESTAZIONI DI UN DEDS: PRESENZA DI BLOCCHI

Definizione

Uno stato x di un automa $G = (X, E, f, \Gamma, x_0, X_m)$ si dice bloccante (deadlock) se

- $x \notin X_m$
- esiste $s \in E^*$ tale che $f(x_0,s) = x$
- $\Gamma(x) = \emptyset$.

Se esiste uno stato bloccante x, ogni stringa s tale che $f(x_0,s) = x$ descrive un comportamento che da, come esito, il mancato l'espletamento dei compiti previsti.

ANALISI DELLE PRESTAZIONI DI UN DEDS: PRESENZA DI BLOCCHI

Teorema

Se l'automa G possiede stati bloccanti, allora $\overline{\Lambda_m(G)} \neq \Lambda(G)$.

Esercizio

- Si dimostri quanto affermato sopra.
- Si provi con un controesempio che il viceversa di quanto affermato sopra non è vero (suggerimento: due stati sono sufficienti …)

ANALISI DELLE PRESTAZIONI DI UN DEDS: PRESENZA DI BLOCCHI

Un'altra situazione nella quale sono presenti comportamenti indesiderati, cioè che non portano all'espletamento di alcuno dei compiti previsti, è data dall'esistenza di sottoinsiemi di stati non marcati che, se raggiunti, non possono essere abbandonati.

Definizione

Un sottoinsieme di stati X' di un automa $G = (X, E, f, \Gamma, x_0, X_m)$ si dice costrittivo (livelock) se

- $X' \cap X_m = \Phi$;
- X' non contiene stati bloccanti;
- esistono $x \in X'$ ed $s \in E^*$ tali che $f(x_0,s) = x$;
- $f(x,s) \in X'$ per ogni $x \in X'$ ed $s \in E^*$ per i quali $f(x,s)$ sia definito.

ANALISI DELLE PRESTAZIONI DI UN DEDS: PRESENZA DI BLOCCHI

Teorema

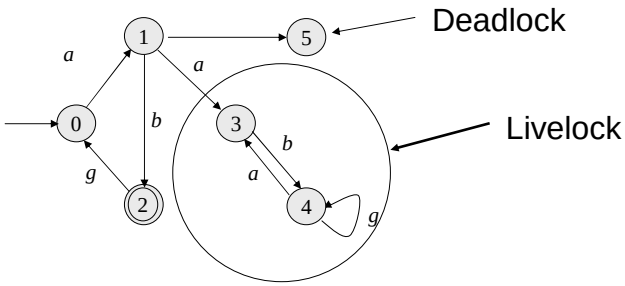
Se l'automa G possiede un sottoinsieme costrittivo di stati, allora $\overline{\Lambda_m}(G) \neq \Lambda(G)$.

Esercizio

- Si dimostri quanto affermato sopra (suggerimento: si consideri un comportamento che porta a raggiungere uno stato dell'insieme costrittivo e si ragioni su quali stringhe ammettono come prefisso la stringa ad esso associata).
- Si provi con un controesempio che il viceversa di quanto affermato sopra non è vero (suggerimento: due stati sono sufficienti …).

Esercizio
Si provi che G non ha stati bloccanti ne sottoinsiemi di stati costrittivi se $\Lambda_m(G) = \Lambda(G)$.

Esempio

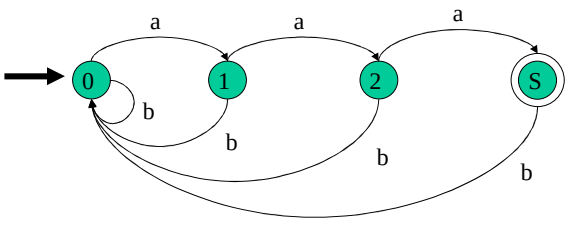


ESEMPIO

Dispositivo che riconosce una terna di cifre

Il dispositivo che consideriamo ha il seguente funzionamento:
- legge in sequenza le cifre (comprese nell'insieme 0, …, 9) che gli vengono fornite;
- segnala se, nel corso della lettura, ha rilevato la terna 666.

In un possibile modello DEDS, possiamo considerare due soli eventi:
a = la cifra letta è 6
b = la cifra letta non è 6.
Una rappresentazione, in tal caso, è la seguente:



Si analizzi il significato degli stati nel modello precedente.

ESEMPIO

Dispositivo di verifica dello stato all'accensione

Il dispositivo che consideriamo ha il compito di verificare lo stato di una macchina al momento dell'accensione di quest'ultima e di segnalare se essa è pronta per lavorare o se vi sono problemi (pensiamo ad esempio ad un dispositivo di questo tipo impiegato su una fotocopiatrice: al momento dell'accensione vengono eseguite un certo numero di verifiche, relative al funzionamento della lampada, alla presenza della carta, del toner, eccetera, e viene mostrato sul display un messaggio relativo all'esito positivo o negativo di tali verifiche).
In un possibile modello DEDS, possiamo considerare quali eventi rappresentativi del funzionamento del dispositivo i seguenti:

a = avvenuta accensione (display ON)

b = esito positivo delle verifiche (display ON / STATUS OK)

c = esito negativo delle verifiche (display ON / ERROR)

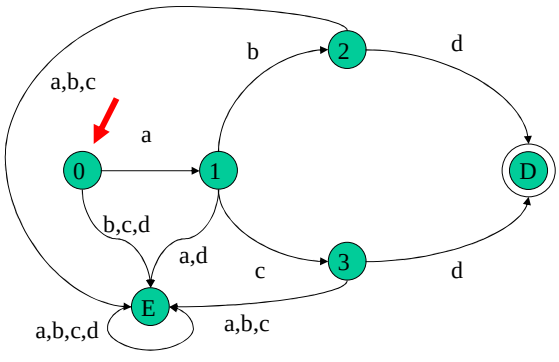
d = indicazione del completamento del compito (display ON / … / STATUS REPORT DONE)

Una esecuzione del compito in un ordine non corretto (ad esempio una indicazione di compito espletato prima di aver comunicato lìesito delle verifiche) deve dar luogo ad una segnalazione di errore (display lampeggiante).

ESEMPIO

Dispositivo di verifica dello stato all'accensione

Una rappresentazione, in tal caso, è la seguente:



Analizzare i linguaggi generato e individuare gli eventuali stati bloccanti.

OPERAZIONI SU AUTOMI

Le operazioni che definiremo servono per sintetizzare nuovi automi a partire da automi dati.

OPERAZIONI UNARIE:
Argomento dell'operazione è un singolo automa

OPERAZIONI BINARIE (O DI COMPOSIZIONE):
Argomento dell'operazione sono due automi

OPERAZIONE PARTE ACCESSIBILE

Definizione

Uno stato x di un automa $G = (X, E, f, \Gamma, x_0, X_m)$ si dice accessibile o raggiungibile se esiste $s \in E^*$ tale che $f(x_0,s) = x$.

Si chiama parte accessibile di G il sottoautoma AC(G) ottenuto eliminando tutti gli stati non accessibili e tutti gli archi che hanno uno di tali stati come estremo.

Definizione

Si chiama Parte Accessibile l'operazione che associa ad un automa G la sua parte accessibile AC(G).

Si provi che $\Lambda(G) = \Lambda(AC(G))$ e $\Lambda_m(G) = \Lambda_m(AC(G))$.

OPERAZIONE PARTE COACCESSIBILE

Definizione

Uno stato x di un automa $G = (X, E, f, \Gamma, x_0, X_m)$ si dice coaccessibile se esiste $s \in E^*$ tale che $f(x,s) \in X_m$.

Si chiama parte coaccessibile di G il sottoautoma COAC(G) ottenuto eliminando tutti gli stati non coaccessibili e tutti gli archi che hanno uno di tali stati come estremo.

Definizione

Si chiama Parte Coaccessibile l'operazione che associa ad un automa G la sua parte coaccessibile COAC(G).

L'operazione può ridurre il linguaggio generato ma non influenza quello marcato. Si verifichi quanto detto provando che $\Lambda_m(G) = \Lambda_m(COAC(G))$ e costruendo un esempio con $\Lambda(G) \neq \Lambda(COAC(G))$.

Si provi che se G = COAC(G), allora $\Lambda(G) = \overline{\Lambda_m(G)}$ e G non ha stati bloccanti.

OPERAZIONE PARTE TRIM

Definizione

Un automa $G = (X, E, f, \Gamma, x_0, X_m)$ si dice trim o ridotto se ogni suo stato è accessibile e coaccessibile.

Si chiama parte trim o ridotta di G il sottoautoma T(G) ottenuto eliminando tutti gli stati non accessibili o non coaccessibili e tutti gli archi che hanno uno di tali stati come estremo.

Definizione

Si chiama Parte Trim l'operazione che associa ad un automa G la sua parte trim T(G).

Si provi che COAC(AC(G)) = AC(COAC(G)) = T(G.

OPERAZIONE COMPLEMENTO

Sia G = (X, E , f , Γ, x_0 , X_m) un automa trim che marca il linguaggio L $\subseteq$ E^*. Allora G genera il linguaggio $\overline{L}$ e possiamo costruire un nuovo automa C(G), detto complemento di G, che marchi il linguaggio E*\L come segue.

1. Si aggiunge uno stato x_d (stato morto o "dump") non marcato ad X e si definisce f_{ex} ponendo
   - $f_{ex}(x,s) = f(x,s)$ se $f(x,s)$ è definita e $f_{ex}(x,s) = x_d$ se $f(x,s)$ non è definita;
   - $f_{ex}(x_d,s) = x_d$ per ogni $s \in E^*$.

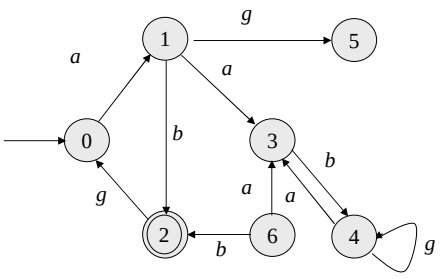
Si noti che detto G_{ex} il nuovo automa così costruito si ha $L(G_{ex}) = E^*$, $L_m(G_{ex}) = L$.

2. Si cambia la marcatura di G_{ex}, marcando tutti gli stati originariamente non marcati, e togliendo la marcatura a quelli marcati e si chiama C(G) il nuovo automa.

Si ha $\Lambda(C(G)) = E^*$, $\Lambda_m(C(G)) = E^* \setminus L$.
L'operazione che associa C(G) a G si dice operazione complemento.

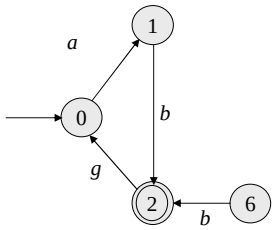
Esempio

- Sia G

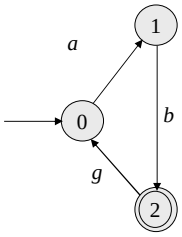


- Lo stato 6 non è accessibile
- AC(G) non contiene 6 e le due transizioni (a, b) relative ad esso.

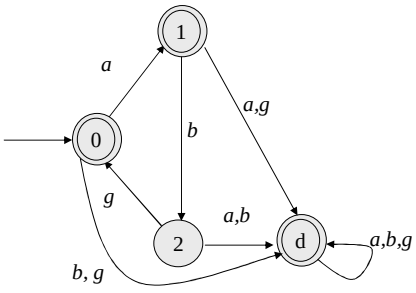
- COAC(G): si ottiene eliminando gli stati di G che non sono coaccessibili: cioè gli stati 3, 4, 5.



- T(G) è dato da:



- Complemento di Trim G:



- Aggiungiamo d e estendiamo f
- Scambiamo la condizione di marcatura su tutti gli stati

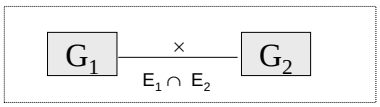
OPERAZIONI BINARIE SU AUTOMI

Le operazioni binarie che consideriamo sono:

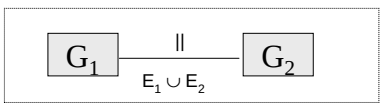
- il prodotto, indicato con $\times$ (o composizione in serie o composizione completamente sincrona)
- la composizione in parallelo, indicata con || (o composizione sincrona)

$G_1 = (X_1, E_1 , f_1 , \Gamma_1, x_{01} , X_{m1})$ $G_2 = (X_2, E_2 , f_2 , \Gamma_2, x_{02} , X_{m2})$

$G_1 \times G_2$



$G_1 \,||\, G_2$



PRODOTTO

$G_1 \times G_2 := AC(X_1 \times X_2, E_1 \cap E_2, f, \Gamma_{1\times2}, (x_{01},x_{02}), X_{m1} \times X_{m2})$

dove

$$f((x_1,x_2), e) := \begin{cases} (f_1(x_1, e), f_2(x_2, e)), & e \in \Gamma_1(x_1) \cap \Gamma_2(x_2) \\ \text{non definita} & \text{altrimenti} \end{cases}$$

e quindi

$$\Gamma_{1\times2},(x_1,x_2) = \Gamma_1(x_1) \cap \Gamma_2(x_2).$$

NB Prendiamo in considerazione solo la parte accessibile poiché solo questa è di interesse ai fini della generazionedi un linguaggio

NB Nel prodotto le transizioni dei due automi devono essere sempre sincronizzate da un evento comune in $E_1 \cap E_2$.

- Il prodotto rappresenta dunque l'interconnessione a passo-bloccato di G_1 e G_2 dove un evento accade se e solo se accade in ambedue gli automi.

- Gli stati di $G_1 \times G_2$ sono indicati a coppie, con il primo elemento stato di G_1 ed il secondo stato di G_2

$$\Lambda (G_1 \times G_2) = \Lambda (G_1) \cap \Lambda (G_2)$$

$$\Lambda_m (G_1 \times G_2) = \Lambda_m (G_1) \cap \Lambda_m (G_2)$$

L'intersezione di due linguaggi può essere realizzata tramite il prodotto delle rispettive rappresentazioni con automi:

- Se $E_1 \cap E_2 = \Phi \Rightarrow \Lambda (G_1 \times G_2) = \{\varepsilon\}$,

$$\Rightarrow \Lambda_m (G_1 \times G_2) = \begin{cases} \varnothing \\ \{\varepsilon\} \end{cases}$$

a seconda dello stato di marcatura dello stato iniziale (x_{01},x_{02}).

ESEMPIO

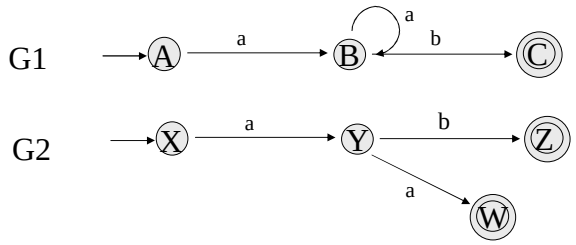
Supponiamo che l' automa G1 rappresenti il funzioanmento di un dispositivo che, scorrendo su una faccia di un pezzo in lavorazione, è in grado di compiere le seguenti operazioni

a = eseguire una operazione di pulitura

b = eseguire una operazione di verniciatura su una faccia pulita

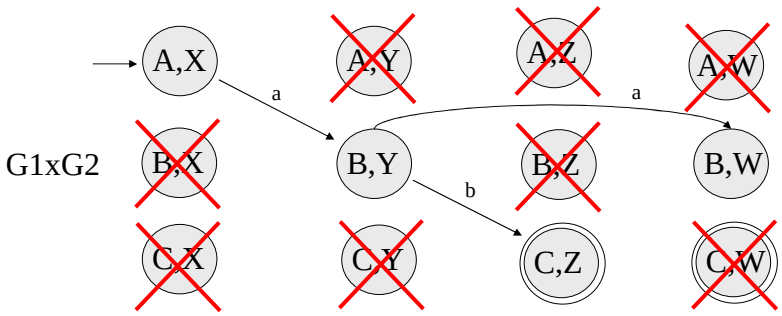
e si arresta, segnalandolo, dopo ogni operazione di verniciatura.

Analogamente, supponiamo che l'automa G2 rappresenti il funzioanmento di un dispositivo similare, che però è regolato in modo tale da arrestarsi dopo aver compiuto due operazioni.



ESEMPIO (cont.)

Per mettere in opera un dispositivo che esegua contemporaneamente pulitura e verniciatura su due facce diverse dello stesso pezzo e si fermi, segnalandolo, al termine di questa operazione, si può pensare di utilizzare i due dispositivi visti. Il comportamento globale che si ottiene può essere descitto dall'automa G1xG2.



COMPOSIZIONE PARALLELA

$G_1 \text{ II } G_2 := AC(X_1 \times X_2, E_1 \cup E_2, f, \Gamma_{1 \text{ II } 2}, (x_{01},x_{02}), X_{m1} \times X_{m2})$

dove

$$f((x1,x2), e) := \begin{cases} (f_1(x_1, e), f_2(x_2, e)), & e \in \Gamma_1(x_1) \cap \Gamma_2(x_2) \\ (f_1(x_1, e), x_2), & e \in \Gamma_1(x_1) \setminus E_2 \\ (x_1, f_2(x_2, e)), & e \in \Gamma_2(x_2) \setminus E_1 \\ \text{non definita} & \text{altrimenti} \end{cases}$$

e quindi

$$\Gamma_{1 \text{ II } 2},(x_1,x_2) = (\Gamma_1(x_1) \cap \Gamma_2(x_2)) \cup \Gamma_1(x_1) \setminus E_2 \cup \Gamma_2(x_2) \setminus E_1.$$

NB Nella composizione parallela un evento comune (cioè appartenente a $E_1 \cap E_2$) può aver luogo solo se ciò avviene simultaneamente per i due:

Ne segue che i due automi risultano sincronizzati per quanto riguarda gli eventi comuni.

In particolare, se $E_1 = E_2$, la composizione parallela coincide col prodotto e tutti gli eventi risultano sincronizzati.

NB Gli eventi che non sono comuni possono aver luogo senza vincoli

Per quanto riguardi questi eventi, gli automi non sono sincronizzati.

In particolare, se $E_1 \cap E_2 = \Phi$, non c'è alcun sincronismo e si definisce il comportamento come concorrente.

La composizione parallela gode delle proprietà commutativa e associativa:

$$G_1 \,\text{II}\, G_2 = G_2 \,\text{II}\, G_1$$
$$(G_1 \,\text{II}\, G_2) \,\text{II}\, G_3 = G_1 \,\text{II}\, (G_2 \,\text{II}\, G_3)$$

Esercizio

Si verifichi se valgono proprietà analoghe per il prodotto e si diano opportuni esempi e/o controesempi.

ESEMPIO

Supponiamo che l' automa G1 rappresenti il funzionanmento di un dispositivo per verniciatura che compie le seguenti operazioni:

G = carica il serbatoio di colore GIALLO

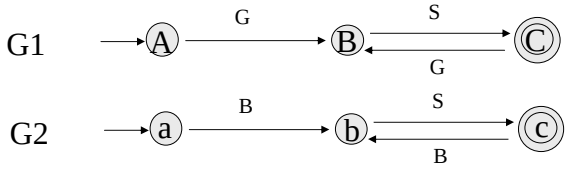
S = spruzza il colore sul pezzo in lavorazione

e supponiamo che, analogamente, l'automa G2 rappresenti il funzioanmento di un altro dispositivo per verniciatura dello stesso tipo che compie le seguenti operazioni:

B = carica il serbatoio di colore BLU

S = spruzza il colore sul pezzo in lavorazione

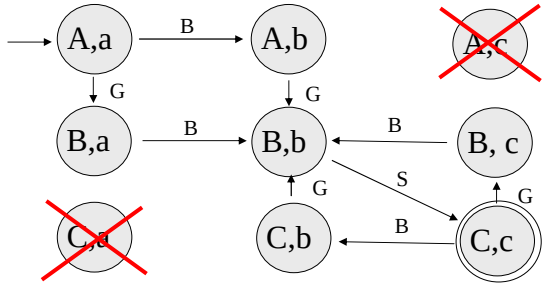
e supponiamo che entrambi I dispositivi segnalino l'avvenuta verniciatura.



ESEMPIO (cont.)

Per ottenere una verniciatura di colore VERDE, spruzzando BLU e GIALLO, si può mettere in opera un dispositivo composto dai due precedenti. Naturalmente, per avere il risultato voluto, occorrerà che l'operazione di spruzzo venga eseguita dai due dispositivi contemporaneamente, mentre le operazioni di carica dei serbatoii possono avvenire in maniera indipendente per ciascun dispositivo. La segnalazione dovrà riguardare anche in questo caso l'avvenuta verniciatura. Il funzionamento complessivo può allora essere descritto dall'automa G1 ‖ G2.

G1‖G2



Confronto tra sistemi

In varie occasioni, dato l'insieme dei comportamenti di due sistemi, possiamo essere interessati a confrontarli tra loro (ad esempio per comprendere quale dei due soddisfa maggiormente ad alcune specifiche). Nel caso di DEDS come quelli che stiamo trattando, ciò equivale a confrontare tra loro due linguaggi L ed L', costruiti su un alfabeto comune E. Se si dispone di due automi $G = (X, E, f, \Gamma, x_0, X_m)$ e $G' = (X', E, f', \Gamma', x_0', X_m')$ che generano L ed L', il confronto può essere fatto utilizzando questi ultimi.

Spesso, ciò che interessa paragonare nel confronto sono insiemi di stringhe aventi il medesimo prefisso. In particolare, data una stringa t in $L \cap L'$, ad esempio, interessa sapere quali delle stringhe te (con $e \in E$) appartenenti ad L appartengono anche ad L' e viceversa.

Per poter fare questo è conveniente identificare lo stato $x = f(x_0, t) \in G$ con lo stato $x' = f'(x_0', t) \in G'$, allo scopo di confrontare agevolmente $\Gamma(x)$ con $\Gamma'(x')$.

Da un punto di vista generale, occorre sviluppare una procedura che consenta di mettere in corrispondenza l'insieme degli stati di G con l'insieme degli stati di G'.

Confronto tra sistemi

Prendiamo in considerazione il caso in cui $L \subseteq L'$. In generale, non è detto che i grafi soggiacenti a G e G' abbiano una struttura simile o compatibile. Può accadere che in G si abbia $f(x_0, t) = f(x_0, s) = x$, mentre in G' $f'(x_0', t) = x' \neq f'(x_0', s) = x''$, e quindi non sia possibile decidere chi tra x' e x"deve essere messo in corrispondenza con x.

Per ovviare a questo tipo di situazioni si può procedere a modificare G, senza alterare il linguaggio L da esso generato, in modo da rendere la struttura del grafo soggiacente compatibile con quella del grafo soggiacente a G'.

Questra operazione prende il nome di raffinamento dello stato di G e può essere attuata ricorrendo all'operazione di prodotto.

Raffinamento tramite prodotto

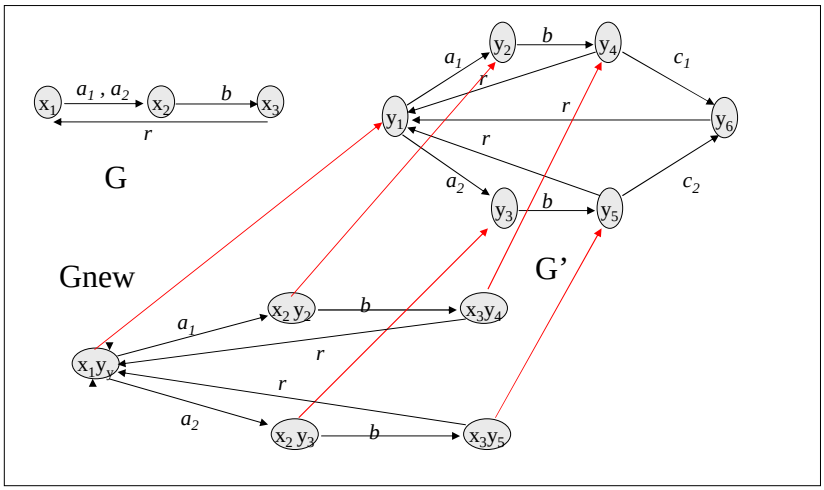
Consideriamo due grafi G e G' con L = L(G) $\subseteq$ L'= L(G') e proponiamoci di effettuare un confronto tra i linguaggi mettendo in corrispondenza gli stati di G, o quelli di un grafo equivalente ottenuto per raffinamento, con quelli di G'.

A tale scopo costruiamo G new= G $\times$ G'.

Gli stati di Gnew sono coppie (x,x'), x $\in$X ed x' $\in$X' ù, e Gnew risulta equivalente a G, cioè genera lo stesso linguaggio L, poichè L(Gnew) = L(G) $\cap$L(G')=L.

La corrispondenza tra stati di Gnew e stati di G' si ottiene associando allo stato (x,x') lo stato x'.

Raffinamento tramite prodotto - Esempio



Linguaggi regolari

Quanto precede mostra l'utilità di rappresentare i linguaggi tramite automi, purchè gli automi stessi siano facilmente maneggiabili. E' questo il caso degli automi aventi uno spazio degli stati finito, o automi a stati finiti, che possono, in particolare, essere trattati mediante calcolatori a memoria finita.

Il problema che ci si pone, dunque, è quello di determinare quali linguaggi (eventualmente composti da infinite parole) si possano rappresentare tramite automi a stati finiti.

Definizione

Dato un alfabeto E, diremo linguaggio regolare ogni linguaggio L costruito su E ($L \subseteq E^*$) che possa essere rappresentato da un automa a stati finiti.

Nota

L'insieme $\Re$ dei linguaggi regolari su E è un sottoinsieme proprio di 2^{E^*}.

Linguaggi regolari

Se L ed L' sono linguaggi regolari, allora sono linguaggi regolari anche

- $\overline{L}$
- $\overline{L}^*$
- $C(L) = E^* \backslash L$
- LL'
- $L \cup L'$
- $L \cap L'$

Linguaggi regolari

Esercizio

Si dimostri quanto asserito nella precedente trasparenza.

(Suggerimenti. Si rifletta su cosa occorre dimostrare e, successivamente, detti G e G' due automi che generno L ed L',

•Si consideri Trim(G) e si marchino tutti i suoi stati …..

•Si aggiunga un nuovo stato iniziale a G, lo si marchi e lo si connetta al vecchio stato iniziale con $\in$. Poi si aggiunga una transizione ε da ciascuno stato marcato di G al vecchio stato iniziale …..

•S consideri l'operazione di complemento …..

•Si crei un nuovo stato iniziale e lo si connetta con due $\in$ agli stati iniziali di G e G', …..

•Si connettano gli stati di G allo stato iniziale di G' mediante $\in$ e si tolga la marcatura a tutti gli stati di G …..

•Si consideri il prodotto $G \times G'$ .....

Linguaggi regolari

I linguaggi regolari possono essere descritti tramite espressioni dette anch'esse regolari.

Nelle espressioni regolari si usa

• il segno + anziché il segno $\cup$ ad indicare l'aggragazione di due o più insiemi di stringhe a formarne un altro: $A+B = A \cup B$

• il segno * ad indicare la chiusura di Kleene di un insieme A (es. $A= \{a\} \Rightarrow A^* = \{\varepsilon, a, aa, aaa, \ldots.\}$)

• la giustapposizione ad indicare l'insieme di stringhe ottenute giustapponendo le stringhe di un insieme a quelle di un altro: $AB = \{ab, a \in A, b \in B\}$.

Linguaggi regolari

Stabiliamo che:

• le espressioni Φ, $\{\varepsilon\}$, $\{e\}$ per ogni $e \in E$ sono regolari

• se A e B consistono di espresioni regolari, anche A*, B*, AB, A+B consistono di espressioni regolari

• non ci sono altre espressioni regolari oltre quelle ottenibili a partire dalle costruzioni implicite nelle due regole precedenti.

Esempio
E={a,b,g}
(a+b)g* indica il linguaggio L={a,b,ag, bg, agg, bgg, aggg, bggg, …}
(ab)*+g indica il linguaggio L={ε, g, ab, abab, ababab, …}

Teorema

Ogni linguaggio che può essere scritto con espressioni regolari è un linguaggio regolare e viceversa

Riconoscitore canonico

Osserviamo che dato un linguaggio L possono esistere più automi diversi che lo marcano (come si fa a costruire un esempio?). Possiamo classificare tali automi in base alla cardinalità del loro spazio degli stati e fissare l'interesse su quelli per i quali si ha cardinalità minima.

Definizione

Un automa che marchi il linguaggio L e abbia insieme degli stasti di cardinalità minima tra tutti gli automi che marcano L è detto riconoscitore canonico di L.

Proposizione

Se L=E*, allora il riconoscitore canonico di L è unico.

(Esercizio: si dimostri quanto detto sopra nel caso E = {a,b})

Riconoscitore canonico

L'interesse nel considerare il riconoscitore canonico è dovuto al fatto che esso rappresenta il modo più economico per descrivere un dato linguaggio e, quindi per studiarne le proprietà.

Dato un automa a stati finiti $G = (X, E, f, \Gamma, x_0, X_m)$, una prima misura della memoria occorrente per rappresentarlo in un calcolatore è data da card(X) $\times$ card(E).

(Esercizio: si valuti l'accuratezza della misura descritta sopra)

Costruzione del riconoscitore canonico

Per costruire il riconoscitore canonico di un dato linguaggio L si opera in genere come segue:

1)si determina un automa G che marchi L;

2)si modifica G, se necessario, riducendo il numero degli stati fino ad ottenere la cardinalità minima, senza modificare il linguaggio marcato.

Si noti che al passo 1) non ci si preoccupa di limitare il numero degli stati.

Il passo 2) consiste essenzialmente in una operazione, detta di aggregazione, che raggruppa più stati in un unico stato. L'aggregazione si basa sulla seguente definizione:

Definizione

Due stati x e x' di un automa G sono detti equivalenti se $\Lambda_m(G(x)) = \Lambda_m(G(x'))$, cioè se l'insieme dei percorsi marcati ottenibili a partire da x e quello dei percorsi marcati ottenibili a partire da x' sono uguali.

L'operazione di aggregazione consiste nell'individuare tutti gli stati equivalenti tra loro e sostituitrli con un unico stato, detto stato aggregato.

(Esercizio: si descriva come compiere il passo 1))

Costruzione del riconoscitore canonico

Dati due stati x e y di G, osserviamo che
- se $x \in X_m$ e $y \notin X_m$, allora x non è equivalente a y;
- se entrambi appartengono o entrambi non appartengono a X_m e
  - f(x, e) = f(y, e) per ogni $e \in E$,

allora x ed y sono equivalenti;
- se entrambi appartengono o entrambi non appartengono a X_m e
  - f(x, e) = f(y, e) per ogni $e \in E' \subseteq E$
  - f(x, e) = y e f(y, e) = x per ogni $e \in E \backslash E'$

allora x ed y sono equivalenti.

Più in generale, se X' è un insieme di stati tali che $X' \subseteq X_m$ oppure $X' \cap X_m = \emptyset$, si ha che R è fatto di stati equivalenti se
  - f(x, e) = $z \notin R$ per $x \in R$ implica f(y, e) = z per ogni $y \in R$.

(Esercizio: si dimostri quanto affermato sopra)

Esempio

•Consideriamo una macchina che legge cifre dall'insieme {1, 2, 3} e marca qualunque stringa che finisca con la sottostringa 123.

•E = {1, 2, 3} ; evento: "la macchina legge n", n = 1, 2, 3.

•L'automa dovrà generare il linguaggio E* e marcare

•Lm = {st $\in$ E* : s $\in$ E* e t = 123}

•Per determinare X e costruire f, cominciamo fissando X_0, che indica la condizione di partenza, prima della quale non è stata letta alcuna cifra.

•Poiché siamo interessati alla stringa 123, definiamo
  - X_1 la prima cifra letta è "1"
  - Xnon1 la prima cifra letta è 2 o 3

Esempio

Consideriamo una macchina che legge cifre dall'insieme {1, 2, 3} e marca qualunque stringa che finisca con la sottostringa 123.

E = {1, 2, 3} ; evento: "la macchina legge n", n = 1, 2, 3.

L'automa dovrà generare il linguaggio E* e marcare

Lm = {st ∈ E* : s ∈ E* e t = 123}

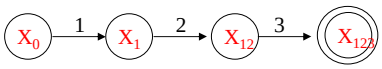
Per determinare X e costruire f, cominciamo fissando X_0, che indica la condizione di partenza, prima della quale non è stata letta alcuna cifra.

Poiché siamo interessati alla stringa 123, definiamo

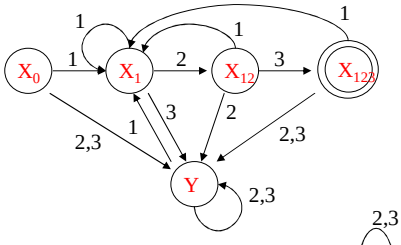
- X_1 l'ultima cifra letta è "1"
- X_{12} le ultime due cifre lette formano la stringa "12"
- X_{123} le ultime tre cifre lette formano la stringa "123", da marcare

Esempio

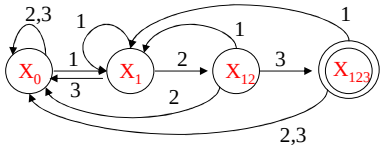
Si ottiene il grafo



al quale occorre aggiungere tutti gli archi necessari a rappresentare, oltre a quello già raffigurato, tutti gli altri possibili comportamenti. Ciò può essere fatto inserendo un ulteriore stato Y e gli archi restanti come segue



Infine, aggregando tra loro gli stati X_0 e Y si ottiene il grafo

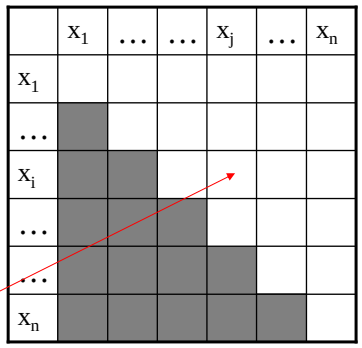


Procedura per ridurre il numero di stati 1

Dato un automa a stati finiti $G = (X, E, f, \Gamma, x_0, X_m)$ con $X=\{x_1,\ldots,x_n\}$

con $\Gamma(x)=E$ per ogni $x \in X$

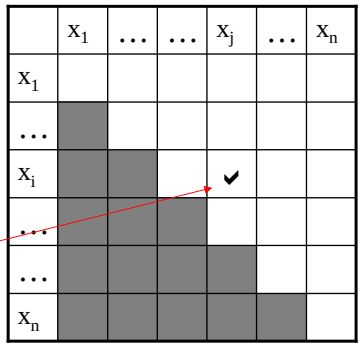
1) si costruisce una tabella avente tante righe e tante colonne quanti sono gli stati di G e si considera la porzione di tabella situata al di sopra della diagonale, diagonale compresa, le cui caselle corrispondono a tutte le posibili coppie di stati di G.



Casella corrispondente alla coppia (x_i,x_j).

Procedura per ridurre il numero di stati 2

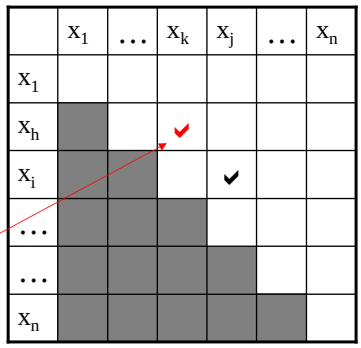
2) si marca con un "flag" ogni casella (x_i,x_j) per la quale $x_i \in X_m$ e $x_j \notin X_m$ o viceversa (il flag, o bandierina, sta a significare che i due stati che compongono la coppia non sono equivalenti)



Flag sulla casella (x_i,x_j) se i due stati che compongono la coppia (x_i,x_j) non sono entrambi marcati o entrambi non marcati

Procedura per ridurre il numero di stati 3

3) si considerano una ad una tutte le coppie di stati (x_h,x_k).che corrispondono a caselle non marcate in precedenza e per ciascuna di esse si considerano tutte le possibili coppie $(f(x_h,e),f(x_k,e))$ con $e \in E$: se, per qualche $e \in E$, la coppia $(f(x_h,e),f(x_k,e))$ (oppure la coppia $(f(x_k,e),f(x_h,e))$ risulta essere flagged, si mette un flag anche sulla casella (x_h,x_k).



Nuovo flag a questo passo sulla casella (x_h,x_k) se, ad esempio, $(f(x_h,e),f(x_k,e))= (x_i,x_j)$, già flagged al passo precedente.

Procedura per ridurre il numero di stati 4

Se al passo 3) è stato aggiunto qualche nuovo flag rispetto a quelli che si avevano al termine del passo 2), si ripete la procedura illustrata al passo 3), cioè si itera il passo 3), considerando, ad ogni iterazione, solo le coppie corispondenti a caselle che non sono ancora flagged.

Il procedimento termina quando, nel corso di una iterazione del passo 3), non vengono aggiunti nuovi flag rispetto a quelli che si avevano al termine dell'iterazione precedente.

In pratica, ad ogni iterazione del passo 3) si marcano con nuovi flag, cioè si indicano come composte da stati non equivalenti, tutte le coppie di stati (x_h,x_k) che mediante uno stesso evento $e \in E$ danno luogo ad una coppia di stati già marcata in precedenza da un flag, cioè una coppia di stati che abbiamo già stabilito non essere equivalenti.

Procedura per ridurre il numero di stati 5

Al termine della procedura illustrata avremo una tabella con caselle flagged a caselle non flagged: le coppie corrispondenti a caselle non flagged sono formate da stati equivalenti tra loro, quindi:

4) procede a raggruppare tali coppie, mettendo in un unico insieme coppie che hanno un elemento in comune. Si ottiene in tal modo una suddivisione di X in insiemi disgiunti di stati equivalenti. Infine, si sostituisce un singolo stato ad ogni insieme di stati equivalenti.

Analisi e sintesi DEDS

La rappresentazione di un DEDS mediante un automa a stati finiti ci consente di

- analizzare le proprieta di funzionamento del DEDS (**analisi**),
- definire una politica di controllo che ci consenta di far funzionare il DEDS rispettando determinate specifiche (**sintesi**).

In generale, lo svolgimento delle procedure di analisi da parte di un calcolatore che abbia memorizzato un modello del DEDS sotto forma di automa a stati finiti, nell'ipotesi frequente che cardE sia molto minore di cardX, ha una complessità computazionale proporzionale alla cardX o a $(cardX)^2$.

In genere, nei problemi di analisi, si suppone che tutti gli eventi siano osservabili per chi compie l'analisi. In alcune situazioni, tuttavia, può accadere che il verificarsi di particolari eventi non possa essere rilevato, si parla in tal caso di eventi non osservabili e, nella probleatica di analisi, rientra anche quella di dedurre informazione su quanto non è osservabile per mezzo di quanto lo è.

Analisi e sintesi DEDS

Problemi di analisi:

- Proprietà e problemi di sicurezza
- Proprietà e problemi di blocco
- Problemi di diagnostica e/o di stima dello stato

Proprietà e problemi di sicurezza

I cosiddetti problemi di sicurezza relativi ad un dato DEDS riguardano la possibilità del verificarsi di singoli eventi o di comportamenti indesiderati.

Trasferendo questo al livello di automa che rappresenta il DEDS il problema diviene quello di verificare la presenza di stringhe o sottostringhe indesiderate nel linguaggio generato.

In alternativa, si può considerare il problema dell'inclusione nel linguaggio generato di un linguaggio (detto legale o ammissibile), che non contenga stringhe o sottostringhe indesiderate.

Estendendo tali problematiche, si può considerare il problema di pervenire ad uno stato indesiderato.

Proprietà e problemi di sicurezza

I problemi di sicurezza vengono affrontati in genere medianti procedure di calcolo dirette.

- La presenza di stringhe indesiderate può venire studiata analizzando l'insieme delle stringhe che si possono comporre a partire da ciascuno stato accessibile.
- La relazione $L \subseteq \Lambda(G)$ può essere studiata sfruttando la sua equivalenza con la relazione $L \cap \Lambda(C(G)) = \Phi$ (si ricordi la costruzione di un automa che genera C(G) e quella del prodotto …).
- La raggiungibilità di un dato stato x a partire da x' si studia costruendo la parte accessibile dopo aver fissato x come stato iniziale.

Proprietà e problemi di blocco

Problemi e proprietà di blocco sono legati al verificarsi dlla condizione $\overline{\Lambda_m(G)} \neq \Lambda(G)$: se si ha un blocco, allora $\overline{\Lambda_m(G)} \neq \Lambda(G)$.

Se, in un automa bloccante, occorre determinare tutti gli stati di blocco o deadlocks e tutti i livelocks, si può partire col determinare la parte CoAc e poi esaminare l'insieme degli eventi possibili a partire da ciascuno degli stati non CoAc. In questo modo, se ce ne sono, si trovano i deadlocks.

Per quanto riguarda i livelock, essi vanno ricercati, utilizzando opportune procedure di calcolo, tra i gruppi di stati connessi che sono fuori da CoAc.

(Esercizio: Si dimostri quanto affermato).

Diagnostica e stima dello stato

I problemi di diagnostica e stima dello stato si pongono quando non tutti gli eventi si verificano in modo osservabile per chi compie l'analisi.

Non trattermo qui in dettaglio tali problemi, limitandoci a segnalare che essi possono venire affrontati limitandosi a considerare un sottoautoma che non presenti eventi inosservabili. I comportamenti descritti mediante il sottoautoma permetono di avere una stima dei comportamenti del DEDS rappresentato dall'automa di partenza.

Analisi e sintesi DEDS

Problemi di sintesi:

- Definire una politica di controllo (insieme di regole) che applicata ad un dato DEDS ne modifichi in maniera desiderata il comportamento
- Definire una procedura di implementazione della politica di controllo (controllore)

Problemi di sintesi

Si consideri un DEDS Σ rappresentato da un automa $G = (X, E, f, \Gamma, x_0, X_m)$ e si supponga che l'insieme dei comportamenti di Σ contenga degli elementi indesiderat, o, in altri termini, che $\Lambda(G)$ contenga delle stringhe indesiderate (può trattarsi, ad esempio di comportamenti o stringhe che violano certe condizioni di funzionamento, come "esegui l'operazione B solo dopo aver eseguito l'operazione A", o di comportamenti che portano a situazioni, cioè a stati, di blocco o a condizioni di livelock).

Il problema che ci poniamo è quello di modificare l'insieme dei comportamenti di Σ allo scopo di evitare quelli indesiderati.

Il paradigma di controllo, che descrive la nostra possibilità di intervenire su Σ, prevede che

- si conosca lo stato attuale x di Σ
- sia possibile inibire tutti o parte degli eventi appartenenti a $\Gamma(x)$

Problemi di sintesi

La politica di controllo consisterà nell'inibire specifici eventi in modo tale che non si verifichino comportamenti indesiderati.

L'implementazione di tale politica avviene

– valutando la condizione attuale del sistema, cioè lo stato x in cui esso si trova,

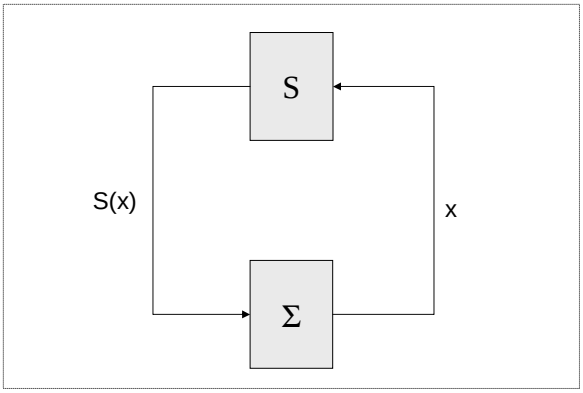
questo ci permettere di individuare l'insieme $\Gamma(x)$, al quale appartengono gli eventi da inibire

– inibendo (oppure: abilitando solo) specifici eventi di $\Gamma(x)$.

Poiché la scelta di quali eventi inibire si base sulla conoscenza dello stato attuale, parleremo di implementazione della politica di controllo in retroazione dallo stato. La modalità di implementazione descrive la struttura del controllore, che indicheremo con S. S(x) indicherà l'azione di controllo (quali eventi inibire) nella situazione (stato) x.

Problemi di sintesi

Schema del controllo in controreazione dallo stato



Problemi di sintesi

In termini più generali (e formali) un problema di sintesi può essere formulato prendendo in considerazione, per un sistema Σ rappresentato dall'automa $G = (X, E, f, \Gamma, x_0, X_m)$, opportuni sottolinguaggi di $\Lambda(G)$, che descrivono il comportamento ammissibile o legale del sistema controllato.

In particolare, ci si può ricondurre a situazioni del tipo

$$\Lambda_r \subset \Lambda_a \subseteq \Lambda(G)$$

dove, essendo $\Lambda(G)$ una rappresentazione del comportamento del sistema senza controllo,

Λ_r rappresenta il comportamento (minimo) desiderato, da ottenersi mediante l'azione di controllo,

Λ_a rappresenta il comportamento ammissibile (massimo) che si è disposti ad accettare

Definizione della politica di controllo

Dato il sistema Σ rappresentato dall'automa $G = (X, E, f, \Gamma, x_0, X_m)$, eventuali eventi di E che non siano disabilitabili mediante politiche di controllo verranno detti eventi non controllabili e si porrà $E = E_C \cup E_{NC}$ (la presenza di eventi non disabilitabili può essere dovuta all'esistenza di eventi intrinsecamente non prevedibili, come i guasti, o a limiti della capacità di intervento sul sistema, come limiti dovuti agli attuatori, o a scelte di modellazione).

Definizione

Una politica di controllo è una funzione $S: X \rightarrow 2^E$ tale che $S(x) \subseteq \Gamma(x)$ (in pratica, S(x) rappresenta l'insieme degli eventi abilitati quando il sistema è nello stato x).

Una politica di controllo S è detta ammissibile se $E_{NC} \cap \Gamma(x) \subseteq S(x)$ per ogni $x \in X$.

Effetti della politica di controllo

L'attuazione della politica di controllo S su G da luogo ad un sistema indicato con S/G, che viene detto controllato in controreazione dallo stato o in catena chiusa. Il comportamento di S/G viene indicato con $\Lambda(S/G)$, cioè come linguaggio generato da S/G.

In pratica, $\Lambda(S/G)$ è costituito dall'insieme delle stringhe di $\Lambda(G)$ (cioè $\Lambda(S/G) \subseteq \Lambda(G)$) che si possono formare mediante gli eventi abilitati da S. Analogamente, $\Lambda_m(S/G) = \Lambda(S/G) \cap \Lambda_m(G)$.

Da quanto abbiamo premesso, segue che, dato un sistema Σ rappresentato dall'automa $G = (X, E, f, \Gamma, x_0, X_m)$ e indicato con Λ_r ($\subset \Lambda_a \subseteq \Lambda(G)$) un comportamento desiderato, il problema di sintesi che ci si pone è innanzitutto quello di definire una politica di controllo S tale che $\Lambda_r \subseteq \Lambda(S/G) \subseteq \Lambda_a \subseteq \Lambda(G)$, o anche $\Lambda_r \subseteq \Lambda_m(S/G) \subseteq \Lambda_a \subseteq \Lambda(G)$. Inoltre, occorrerà determinare il modo per implementare S.

Note

Possiamo definire $\Lambda(S/G)$ ricorsivamente nel modo seguente:

1) $\varepsilon \in \Lambda(S/G)$

2) $[(s \in \Lambda(S/G))$ e $(s\sigma \in \Lambda(G))$ e $(\sigma \in S(f(s,x_0)] \Leftrightarrow [s\sigma \in \Lambda(S/G)]$

Osserviamo che:

- $\Lambda(S/G) \subseteq \Lambda(G)$ è per definizione chiuso rispetto al prefisso
- $\Lambda_m(S/G) \subseteq \overline{\Lambda_m(S/G)} \subseteq \Lambda(S/G) \subseteq \Lambda(G)$
- ε (la stringa vuota) è contenuta in $\Lambda(S/G)$ poichè è contenuta in $\Lambda(G)$

Note

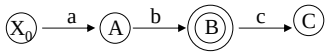
Possiamo avere un comportamente $\Lambda(S/G)$ che prevede situazioni di blocco. Ciò si verifica se

$$\Lambda(S/G) \neq \overline{\Lambda_m(S/G)}$$

In tal caso diremo che la politica di controllo o di supervisione S è di blocco. Ricordiamo che, poiché le stringhe marcate rappresentano compiti ultimati o la registrazione del completamento di una qualche operazione (a seconda di come si è deciso di modellare), una S di blocco da luogo ad un sistema controllato che può non terminare l'esecuzione del compito.

Esempio

Consideriamo il seguente automa G



per il quale si ha L(G) = {ε,a,ab,abcc} , Lm(G) = {ab}.
G è blocking poiché la stringa abc ∈ L(G) non è prefisso di ab.
Se definiamo la politica di controllo S come

S{ε} = {a},
S(A) = {b},
S(B) = ∅,
S(C) = ∅

allora S/G è non blocking poichè

$$L(S/G) = \overline{Lm(S/G)} = \overline{\{ab\}}$$

La politica S definita è ammissibile solo se l'evento c è controllabile. Se l'evento c fosse incontrollabile, non potremmo eliminare le stringhe il comportamento bloccante tramite il controllo.

Teorema di controllabilità

Teorema

Dato un sistema Σ rappresentato da un automa G = (X, E, f, Γ, x_0, X_m), un linguaggio K ⊆ Λ(G) ed un insieme di eventi incontrollabili E_{uc}, se K verifica la seguente condizione

per ogni s ∈ K ed ogni e ∈ E, se ∈ K implica s'e ∈K per ogni s' ∈ K con f(x_0,s) = f(x_0,s') (condizione di completezza)

esiste una politica di controllo S tale che Λ(S/G) = $\overline{K}$, se e solo se $\overline{K}$ E_{uc} ∩ Λ(G) ⊆ $\overline{K}$.

La condizione $\overline{K}E_{uc}$ ∩ Λ(G) ⊆ $\overline{K}$ è detta condizione di controllabilità di K rispetto a Λ(G). Essa è equivalente ad affermare che, per ogni s ∈ K e per ogni e ∈ E_{uc}, se ∈ Λ(G) implica se ∈ K.

Test di controllabilità

Per verificare se K è controllabile rispetto a $\Lambda(G)$, si può operare nel seguente modo.

1) Si costruisce un automa H tale che $\Lambda(H) = \overline{K}$.

2) Si costruisce il prodotto $H \times G$.

3) Per ogni stato $(h,g) \in H \times G$ si confronta $E_{uc} \cap \Gamma(h,g)$ con $E_{uc} \cap \Gamma(g)$.

4) Se esiste un evento e tale che $e \in E_{uc} \cap \Gamma(g)$ ed $e \notin E_{uc} \cap \Gamma(h,g)$, allora K non è controllabile.

Implementazione di S

Dato un sistema Σ rappresentato dall'automa $G = (X, E, f, \Gamma, x_0, X_m)$, un linguaggio $K \subseteq \Lambda(G)$ che verifichi la condizione di completezza ed un insieme di eventi incontrollabili E_{uc}, assumiamo che K sia controllabile (escludiamo i casi non interessanti $K = \Lambda(G)$ e $K = \Phi$).

La politica di controllo S definita da

$S(x) = [E_{uc} \cap \Gamma(x)] \cup \{\sigma \in E_c$ tali che, per ogni $s \in K$ con $f(x_0,s) = x$, si ha $s\sigma \in K\}$

è tale che $\Lambda(S/G) = \overline{K}$ (lo si verifichi).

Per implementare S si considera un automa trim $R = (Y, E, g, \Gamma_R, y_0, Y_m)$ che marca il linguaggio $\overline{K}$, tale cioè che $\Lambda_m(R) = \Lambda(R) = \overline{K}$.

Diremo che R è una realizzazione di S.

L'automa prodotto $R \times G$ descrive il comportamento di di Σ controllato in retroazione mediante S.

Implementazione di S

Quanto affermato segue dal fatto che

$\Lambda(R \times G) = \Lambda(R) \cap \Lambda(G) = \overline{K} \cap \Lambda(G)$

$\Lambda_m(R \times G) = \Lambda_m(R) \cap \Lambda_m(G) = K \cap \Lambda_m(G) = \Lambda(S/G) \cap \Lambda_m(G) = \Lambda_m(S/G)$

Osservando che, per ogni $s \in K$, è possibile identificare $g(y_0,s)$ con $f(x_0, s)$, possiamo dire che

$S(x) = S(f(x ,s)) = [E_{uc} \cap \Gamma(f(x_0, s))] \cup \{\sigma \in E_c : s\sigma \in K\} = \Gamma_R (g(y_0, s)) = \Gamma_{R\times G}(g \times f((y_0, x_0), s))$

dove la prima eguaglianza segue dalla controllabilità di K e la seconda eguaglianza segue dall'essere $K \subset \Lambda(G)$.

L'automa R, in pratica, ci consente di memorizzare in maniera appropriata la politica di controllo S.

Implementazione di S

L'implementazione effettiva di S attraverso il prodotto $R \times G$ avviene come segue:

1) se G è nello stato $f(x_0,s)$ ed R nello stato corrispondente $f(y_0,s)$, l'evento $e \in \Gamma(f(x_0,s))$ è abilitato solo se esso è presente anche in $\Gamma(f(y_0,s))$, cioè solo se $e \in \Gamma(f(y_0,s))$.

2) l'accadimenti dell'evento e, in tal caso, porta negli stati $f(x_0,se)$ ed $f(y_0,se)$ e il processo si ripete.

Un tale modo di operare prende, a volte, il nome di table lookup

Note

In genere, la modifica del comportamento di un sistema tramite un controllore o supervisore è motivata dall'esigenza di rispettare certe specifiche.

Di solito, tali specifiche sono espresse in linguaggio naturale, come, per esempio, le seguenti:

- evitare alcuni stati di G;
- eseguire un insieme di eventi in un dato ordine di priorità,
- non permettere che un evento accada più di n volte fra due accadimenti di un altro evento,
- Permettere l'accadimento di un evento solo a seguito di un determinato altro;
- eccetera.

Note

In pratica, il percorso che occorre compiere per giungere ad ottenere un comportamento desiderato è in sisntesi il seguente:

dati il sistema Σ rappresentato da $G = (X, E, f, \Gamma, x_0, X_m)$, un insieme di eventi incontrollabili E_{uc} e un insieme di specifiche $s_1, \ldots, s_n$ espresse in termini linguistici,

1) si costruisce, se possibile, un sottolinguaggio $K \subseteq \Lambda(G)$ le cui parole rispettano le specifiche (in pratica, si traducono le specifiche espresse in termini linguistici in un modello di linguaggio) e che verifica la condizione di completezza, ;

2) si verifica la controllabilità di K;

3) si costruisce la politica di controllo S tale che L(S/G) = K;

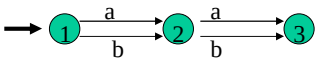
4) si implementa S mediante una sua realizzazione R.

La costruzione di K è in genere molto complessa e richiede esperienza. Può anche accadere che non esista un sottolinguaggio $K \subseteq \Lambda(G)$ le cui parole rispettano le specifiche. In tal caso, occorre modificare in maniera opportuna i dati Σ e G.

Note

Una delle difficoltà più frequenti nella costruzione di K riguarda la necessità di soddisfare specifiche che richiedono di ricordare come si è giunti ad un determinato stato.

Per esempio si consideri



con la specifica aa e ba ammissibili, ab non ammissibile.

Per poter trovare K, in questo caso, è necessario modificare G suddividendo lo stato 2 in due stati distinti, così da aggiungere memoria al sistema