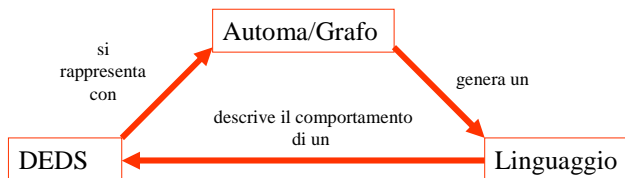
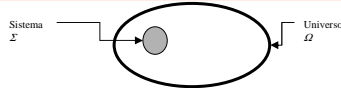


## 1.1 DEDS/Automi/Linguaggi

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

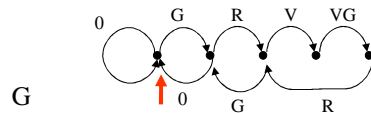


## 1.2 DEDS/Automi/Linguaggi

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

$\Sigma$        $E = \{\text{insieme degli eventi}\}$        $\{\text{insieme dei comportamenti}\}$

$\Sigma_{\text{semaforo}}$        $\{0, V, G, VG, R\}$        $\{(0, VG, R, V, VG, R, \dots); (0, VG, 0); \dots\}$



$\{\text{linguaggio generato da G}\}$

$E = \text{alfabeto} = \{0, V, G, VG, R\}$

## 1.3 DEDS/Automi/Linguaggi

Rappresentando un DEDS  $\Sigma \subseteq \Omega = E^T$  mediante un grafo  $G$ , l'insieme dei comportamenti di  $\Sigma$  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  $\Sigma$  parla il linguaggio  $L$ .

Lo studio di un DEDS  $\Sigma$  può essere condotto attraverso lo studio di un opportuno linguaggio.

Obiettivi :

- analisi del linguaggio parlato da un dato DEDS (analisi).
- costruzione di un DEDS  $\Sigma$  che esibisca un dato insieme di comportamenti, cioè parli un dato linguaggio (modellazione, sintesi);

## LINGUAGGI/DEFINIZIONI E NOTAZIONI

- 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**  $\epsilon$  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:  $E = \{a, b, g\}$

$L_1 = \{\varepsilon, g, abb\}$

$L_2 = \{\text{tutte le possibili stringhe di lunghezza 3, che iniziano con l'evento a; } aaa, aab, aba, abb, aag, aga, agg, abg, agb\}$  (9 stringhe)

$L_3 = \{\text{tutte le possibili stringhe di lunghezza finita che iniziano con a}\}$  ( $\infty$  stringhe)

- 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** finite di elementi di  $E$  compresa la stringa vuota
- $*$  := chiusura di **Kleene**  
p.e.  $E = \{a, b, c\}$ ,  $E^* = \{\varepsilon, a, ab, ac, b, ba, bc, c, ca, cb, aa, bb, \dots\}$
- $E^*$  insieme composto da un'**infinità numerabile di elementi**
- Un qualunque linguaggio su  $E$  è un sottoinsieme di  $E^*$ .  
 $\emptyset, E, E^*$  sono linguaggi.

## TERMINOLOGIA RELATIVA ALLE STRINGHE

- Sia  $s = t u v$ ;  $t, u, v \in E^*$ 
  - $t$  := **prefisso** di  $s$
  - $u$  := **sottostringa** di  $s$
  - $v$  := **suffisso** di  $s$
  
- $\varepsilon$  è prefisso, suffisso, sottostringa di  $s$ , per ogni  $s \in E^*$
  
- $s$  è prefisso, suffisso, sottostringa di se stessa

## OPERAZIONI SUI LINGUAGGI

### CONCATENAZIONE

Sia  $L_a, L_b \subseteq E^*$ , la concatenazione  $L_a L_b$  di  $L_a$  con  $L_b$  è data da

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

(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  $L \subseteq E^*$ , la chiusura  $\bar{L}$  di  $L$  rispetto al prefisso è data da

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

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

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

### CHIUSURA DI KLEENE

Sia  $L \subseteq E^*$ , la chiusura  $L^*$  di Kleene di  $L$  è data da

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

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

## ESEMPI

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

## 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  $\Gamma: X \rightarrow 2^E$ , 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  $G = (X, E, f, \Gamma, x_0, X_m)$  per indicare un automa.

## AUTOMI

Nella rappresentazione di un AUTOMA  $G = (X, E, f, \Gamma, x_0, X_m)$  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

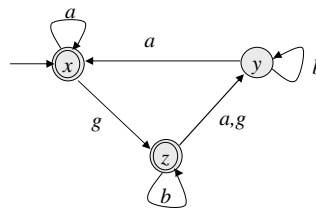
$X = \{x, y, z\}$

$E = \{a, b, g\}$

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

$f(z, b) = z; f(z, a) = f(z, g) = y$

Si noti che  $\Gamma$  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'automata  $G$  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.

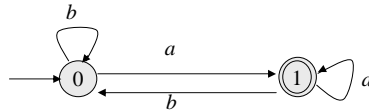
Osserviamo che:

- $\Lambda(G)$  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'automata. Diremo anche che l'automata 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

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

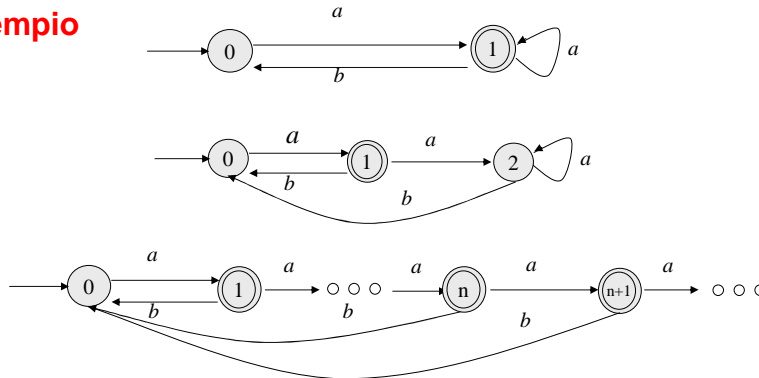
## AUTOMI EQUIVALENTI

### 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  $\Sigma$  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  $\Sigma$  (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 raggiungere 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  $\overline{\Lambda_m(G)} \neq \Lambda(G)$ .

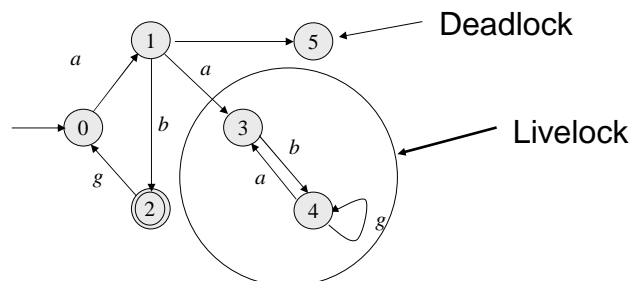
### Esercizio

- 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:

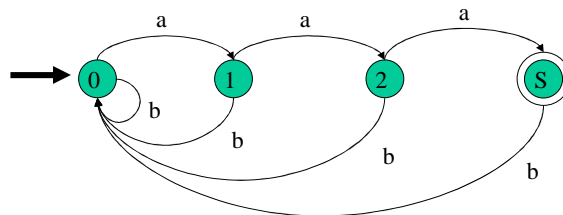
- legge in sequenza le cifre (comprese nell'insieme 0, ..., 9) che gli vengono fornite;
- segnala se, nel corso della lettura, ha rilevato la terna 666.

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)

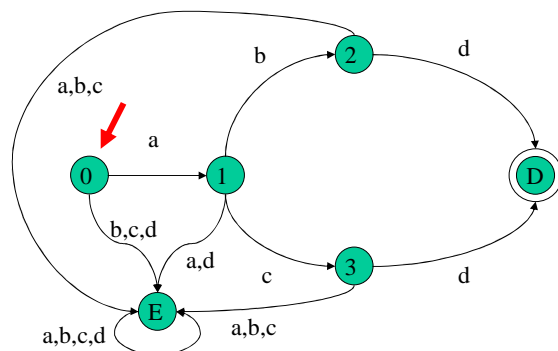
d = indicazione del completamento del compito (display ON / ... / STATUS REPORT DONE)

Una esecuzione del compito in un ordine non corretto (ad esempio una indicazione di compito espletato prima di aver comunicato l'esito 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  $\text{COAC}(\text{AC}(G)) = \text{AC}(\text{COAC}(G)) = T(G)$ .

## OPERAZIONE COMPLEMENTO

Sia  $G = (X, E, f, \Gamma, x_0, X_m)$  un automa trim che marca il linguaggio  $L \subseteq E^*$ . Allora  $G$  genera il linguaggio  $L$  e possiamo costruire un nuovo automa  $C(G)$ , detto complemento di  $G$ , che marchi il linguaggio  $E^* \setminus L$  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  $f_{ex}(x_d,s) = x_d$  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  $L(G_{ex}) = E^*$ ,  $L_m(G_{ex}) = L$ .

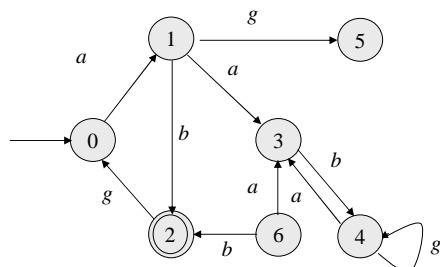
2. Si cambia la marcatura di  $G_{ex}$ , marcando tutti gli stati originariamente non marcati, e togliendo la marcatura a quelli marcati e si chiama  $C(G)$  il nuovo automa.

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

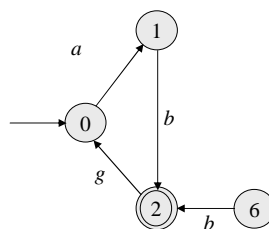
➤ Sia  $G$



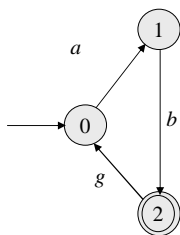
➤ Lo stato 6 non è accessibile

➤  $AC(G)$  non contiene 6 e le due transizioni (a, b) relative ad esso.

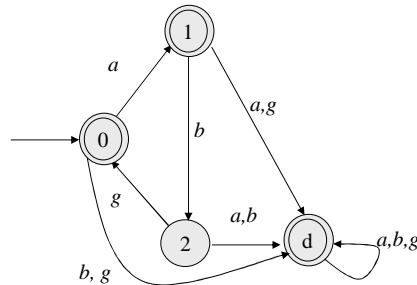
➤  $COAC(G)$ : si ottiene eliminando gli stati di  $G$  che non sono coaccessibili: cioè gli stati 3, 4, 5.



➤  $T(G)$  è dato da:



➤ Complemento di Trim G:



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

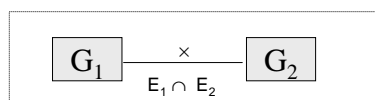
## OPERAZIONI BINARIE SU AUTOMI

Le operazioni binarie che consideriamo sono:

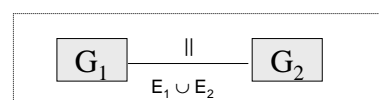
- il prodotto, indicato con  $\times$  (o composizione in serie o composizione completamente sincrona)
- la composizione in parallelo, indicata con  $\parallel$  (o composizione sincrona)

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

$G_1 \times G_2$



$G_1 \parallel G_2$



## PRODOTTO

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

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

$$\Gamma_{1 \times 2}(x_1, x_2) = \Gamma_1(x_1) \cap \Gamma_2(x_2).$$

## ESEMPIO

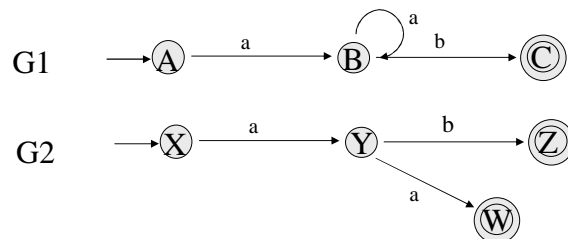
Supponiamo che l' automa G1 rappresenti il funzionamento di un dispositivo che, scorrendo su una faccia di un pezzo in lavorazione, è in grado di compiere le seguenti operazioni

a = eseguire una operazione di pulitura

b = eseguire una operazione di verniciatura su una faccia pulita

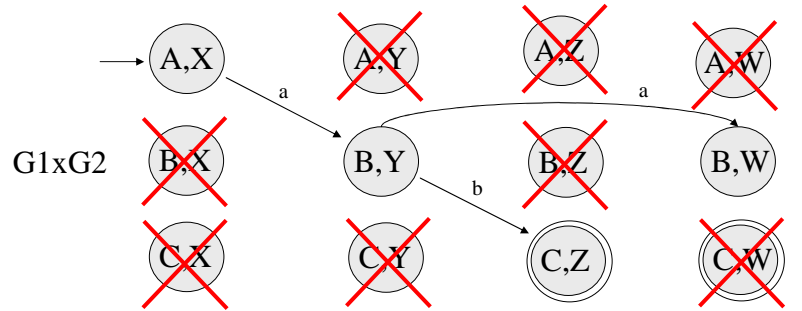
e si arresta, segnalandolo, dopo ogni operazione di verniciatura.

Analogamente, supponiamo che l' automa G2 rappresenti il funzionamento di un dispositivo simile, 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 descritto dall'automata  $G_1 \times G_2$ .



**NB** Prendiamo in considerazione solo la **parte accessibile** poiché solo questa è di interesse ai fini della generazione di 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  $G_2$

## ESEMPIO

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

G = carica il serbatoio di colore GIALLO

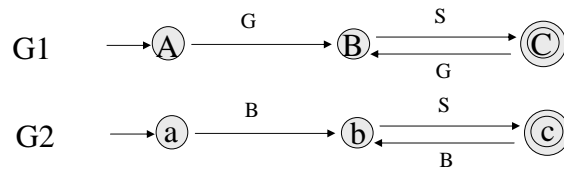
S = spruzza il colore sul pezzo in lavorazione

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

B = carica il serbatoio di colore BLU

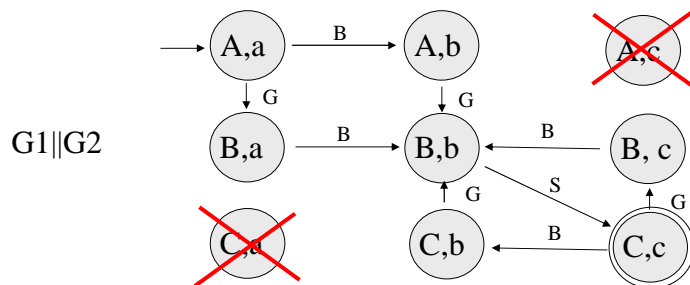
S = spruzza il colore sul pezzo in lavorazione

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 serbatoi 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  $G1 \parallel G2$ .



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

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

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

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

$$\Rightarrow \Lambda_m(G_1 \times G_2) = \begin{cases} \emptyset \\ \{\epsilon\} \end{cases}$$

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

### COMPOSIZIONE PARALLELA

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

dove

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

e quindi

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

**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 riguarda 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:

$$\begin{aligned}G_1 \parallel G_2 &= G_2 \parallel G_1 \\(G_1 \parallel G_2) \parallel G_3 &= G_1 \parallel (G_2 \parallel G_3)\end{aligned}$$

### **Esercizio**

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

## PROIEZIONI

Si definiscono le **proiezioni**:

$$P_i = (E_1 \cup E_2)^* \rightarrow E_i^*, i = 1, 2$$

come segue:

$$\begin{aligned} P_i(\varepsilon) &:= \varepsilon \\ P_i(e) &:= \begin{cases} e & \text{se } e \in E_i \\ \varepsilon & \text{se } e \notin E_i \end{cases} \\ P_i(se) &:= P_i(s) \cdot P_i(e), s \in (E_1 \cup E_2)^*, e \in (E_1 \cup E_2) \end{aligned}$$

In pratica, la proiezione  $P_i$  cancella in ogni stringa di eventi in  $(E_1 \cup E_2)^*$  tutti gli eventi che non appartengono a  $E_i$ .

Considerando la controimmagine della proiezione, otteniamo una mappa, detta **proiezione inversa**,

$$P_i^{-1} : E_i^* \rightarrow 2^{(E_1 \cup E_2)^*}$$

Insieme di tutti i sottoinsiemi di  $(E_1 \cup E_2)^*$

per la quale si ha:

$$P_i^{-1}(t) := \{s \in (E_1 \cup E_2)^* : P_i(s) = t\}$$

In pratica, data una stringa  $t$  in  $E_i$ , la **proiezione inversa**  $P_i^{-1}$  fornisce l'insieme di tutte le stringhe  $s$  in  $E_1 \cup E_2$  che si proiettano, tramite  $P_i$ , sulla stringa data.

Le **proiezioni** e le loro **proiezioni inverse** si estendono ai **linguaggi**, definendole in modo naturale su tutte le **stringhe** che ne fanno parte.

$$L \subseteq (E_1 \cup E_2)^*, P_i(L) := \{t \in E_i^* : \exists s \in L \text{ con } P_i(s) = t\}$$

$$L_i \subseteq (E_i)^*, P_i^{-1}(L_i) := \{s \in (E_1 \cup E_2)^* E_i^* : \exists t \in L_i \text{ con } P_i(s) = t\}$$

Si osservi che:

$$P_i(P_i^{-1}(L)) = L$$

$$L \subseteq P_i^{-1}(L)$$

### ESEMPIO

$$E_1 = \{a, b\}, E_2 = \{b, c\}, L = \{c, ccb, abc, cacb, cabcbba\}$$

$$P_1(L) = \{\varepsilon, b, ab, abbba\},$$

$$P_2(L) = \{c, ccb, bc, cbcbbc\},$$

$$P_1^{-1}(\varepsilon) = \{c\}^*, P_1^{-1}(b) = \{c\}^* \{b\} \{c\}^*,$$

$$P_1^{-1}(ab) = \{c\}^* \{a\} \{c\}^* \{b\} \{c\}^*.$$

**Nota bene:**

$$P_1^{-1}[P_1(abc)] = P_1^{-1}[\{ab\}] \supset \{abc\}$$

Osservando le espressioni delle **proiezioni inverse** nell'esempio, possiamo trovare una semplice implementazione di questa operazione sugli **automi** che rappresentano i **linguaggi**:

$$\text{Se } K_s = \mathcal{L}_m(G) \subseteq E_s^* \subseteq E_l^*$$

e  $P$  è la **proiezione** di  $E_l$  su  $E_s$ , allora un **automa** che marca  $P^{-1}(K_s)$  su può essere ottenuto aggiungendo a tutti gli stati di  $G$  **auto-anelli** per tutti gli eventi in  $E_l \setminus E_s$ .

I **linguaggi** che risultano da una **composizione parallela** di automi possono essere caratterizzati come segue:

$$\mathcal{L}(G_1 \parallel G_2) = P_1^{-1}[\mathcal{L}(G_1)] \cap P_2^{-1}[\mathcal{L}(G_2)]$$

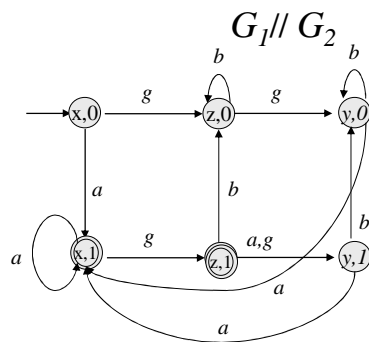
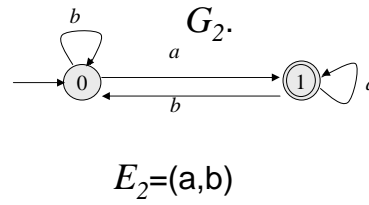
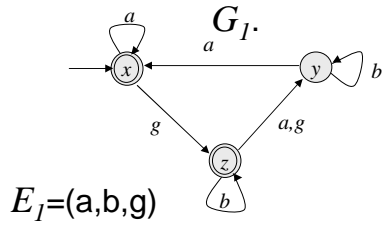
$$\mathcal{L}_m(G_1 \parallel G_2) = P_1^{-1}[\mathcal{L}_m(G_1)] \cap P_2^{-1}[\mathcal{L}_m(G_2)]$$

Si può allora definire una **composizione parallela di linguaggi**:

$$L_i \subseteq E_i^*, \quad i = 1, 2, \quad P_i \text{ come sopra}$$

$$L_1 // L_2 := P_1^{-1}(L_1) \cap P_2^{-1}(L_2)$$

**ESEMPIO**



*Tutti e sei gli stati in  $X_1 \times X_2$  sono raggiungibili da  $(x,0)$*

$$P_i[\mathcal{L}(G_1 // G_2)] \subseteq \mathcal{L}(G_i) \quad i = 1, 2$$