# Automi

L’approccio alla verifica in cui le specifiche sono codificate da formule LTL usa lo stesso tipo di rappresentazione per descrivere programmi e specifiche:

- il **sistema** è modellato da un sistema di transizioni, cioè un **automa** a stati finiti (con etichette sugli stati, ed eventualmente sugli archi) le cui esecuzioni sono infinite.
- la **specifica** è rappresentata da una formula LTL, che a sua volta rappresenta un insieme di interpretazioni. Un insieme di interpretazioni di LTL si può rappresentare in modo finito mediante un **automa** a stati finiti (con etichette sugli stati) che legge parole infinite.

**$\omega$-automi:** automi a stati finiti che leggono parole infinite (di $\Sigma^\omega$).
Sono un modo finito di rappresentare esecuzioni infinite.

Linguaggi accettati da $\omega$-automi: **$\omega$-regolari**.
Si possono descrivere con espressioni che contengono simboli dell’alfabeto $\Sigma$ e gli operatori:

| | | | |
|---|---|---|---|
| $+$ | unione | $*$ | zero o più ripetizioni |
| $\cdot$ | concatenazione | $\omega$ | infinite ripetizioni |

# Automi di Büchi (BA)

Costituiscono la classe più semplice di $\omega$-automi.

Variante con etichette sugli stati: un BA è:

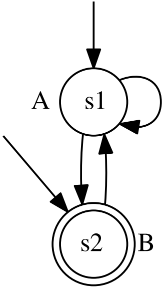
$$\mathcal{A} = ( \underbrace{\underbrace{S, \Delta, I}_{automa}, L, \Sigma}_{automa\ etichettato} , \boldsymbol{F})$$

| | |
|---|---|
| $\boldsymbol{S}$ | : insieme finito di stati |
| $\boldsymbol{\Delta \subseteq S \times S}$ | : relazione di transizione |
| $\boldsymbol{I \subseteq S}$ | : stati iniziali |
| $\boldsymbol{\Sigma}$ | : alfabeto finito |
| $\boldsymbol{L : S \to \Sigma}$ | : etichettatura degli stati |
| $\boldsymbol{F \subseteq S}$ | : **stati di accettazione** (indicati con doppio cerchio) |

**Attenzione:** **non** ci sono stati **finali**

# Esempio: rappresentazione grafica

$$
\begin{array}{lcl}
S & = & \{s1, s2\} \\
\Delta & = & \{(s1, s1), (s1, s2), (s2, s1)) \\
I & = & \{s1, s2\} \\
\Sigma & = & \{A, B\} \\
L & = & s1 \mapsto A, s2 \mapsto B \\
F & = & \{s2\}
\end{array}
$$



# Esecuzioni

$$\mathcal{A} = (S, \Delta, I, L, \Sigma, F)$$

**Esecuzione** $\rho$ di $\mathcal{A}$: cammino infinito nel grafo che inizia in uno stato iniziale.

Un'esecuzione è un mapping $\boldsymbol{\rho : \mathbb{N} \rightarrow S}$:
$\rho(i)$ è lo stato in posizione $i$ nell'esecuzione: $\rho = \rho(0)\ \rho(1)\ \rho(2)\ \ldots$

Un'esecuzione $\rho$ è dunque una funzione $\mathbb{N} \rightarrow S$ tale che:

- $\rho(0) \in I$
- $\langle \rho(i), \rho(i+1) \rangle \in \Delta$, per ogni $i \geq 0$

Sia $v \in \Sigma^\omega$ una parola infinita sull'alfabeto $\Sigma$. Anche una parola infinita è una funzione, $v : \mathbb{N} \rightarrow \Sigma$: $v = v(0)\ v(1)\ v(2)\ \ldots$

Un'esecuzione $\rho$ **legge** la parola $v$ se gli stati dei $\rho$ sono etichettati in accordo con $v$:

- $L(\rho(i)) = v(i)$ per ogni $i \geq 0$

**Nota:** "$\mathcal{A}$ legge $v$" non è la stessa cosa di "$\mathcal{A}$ accetta $v$"

# Esecuzioni di accettazione

$$\mathcal{A} = (S, \Delta, I, L, \Sigma, F)$$

Un'esecuzione di accettazione contiene infinite occorrenze di almeno uno stato di accettazione in $F$:

Sia inf($\rho$) l'insieme degli stati che compaiono infinite volte in $\rho$
(N.B. inf($\rho$) è un insieme finito).

Un'esecuzione $\rho$ è di accettazione se

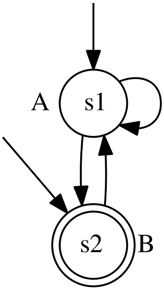
$$\text{inf}(\rho) \cap F \neq \emptyset$$

Un'esecuzione $\rho$ **accetta** la parola $v$ se $\rho$ è un'esecuzione d'accettazione che legge $v$.

L'automa $\mathcal{A}$ accetta una parola $v$ se esiste un'esecuzione di $\mathcal{A}$ che accetta $v$.

Il **linguaggio** $\mathcal{L}(\mathcal{A}) \subseteq \Sigma^\omega$ di $\mathcal{A}$ è l'insieme delle parole accettate da $\mathcal{A}$

# Esempio




Esecuzioni d'accettazione: infinite occorrenze di s2

Parole accettate: $(A^* BA)^\omega$

Non accettate: tutte le parole che terminano con $A^\omega$

# Modellazione di sistemi mediante Automi di Büchi

Un sistema i cui stati sono etichettati da insiemi di variabili proposizionali in $P$ si può rappresentare mediante un BA

$$\mathcal{A} = (S, \Delta, I, L, 2^P, S)$$

dove:

- $L : S \rightarrow 2^P$ etichetta ciascuno stato $s \in S$ con un insieme di lettere proposizionali, quelle vere in $s$;
- $(s, r) \in \Delta$ sse c'è una transizione atomica da $s$ a $r$;
- $2^P$: alfabeto, i cui simboli sono insiemi di lettere proposizionali (sottoinsiemi di $P$);
- Stati di accettazione: tutti ($S$).

**Se sul sistema non sono imposti vincoli di fairness,
nell'automa di Büchi corrispondente
tutti gli stati sono stati di accettazione**

# Stati di accettazione e vincoli di fairness

Se sul modello di un sistema si impone un vincolo di fairness, identificato da un insieme $F$ di stati, allora il modello è un automa di Büchi della forma

$$\mathcal{A} = (S, \Delta, I, L, 2^P, F)$$

L'insieme degli stati di accettazione dell'automa coincide con l'insieme che rappresenta il vincolo di fairness

- un'esecuzione fair passa infinite volte per almeno uno stato di $F$
- un'esecuzione fair è un'esecuzione di accettazione dell'automa di Büchi $\mathcal{A}$

# Esecuzioni di un automa e interpretazioni temporali

Consideriamo un automa con etichette in $2^P$:

$$\mathcal{A} = (S, \Delta, I, L, 2^P, F)$$

Le parole lette da $\mathcal{A}$ sono **sequenze infinite di sottoinsiemi di $P$**:
$X_1, X_2, X_3, \ldots$ con $X_i \subseteq P$

Ogni insieme $X_i \subseteq P$ è un'interpretazione proposizionale classica

Quindi le parole lette da $\mathcal{A}$ sono interpretazioni del linguaggio di LTL su $P$:
un'esecuzione $\rho$ di $\mathcal{A}$ legge $\mathcal{M}$ tale che, per ogni $i \in \mathbb{N}$:

$$\mathcal{M}(i) = L(\rho(i))$$

# Specifica di proprietà di sistemi mediante BA

La specifica di una proprietà di un sistema $\mathcal{A}$ si può dare mediante un automa $\mathcal{B}$ sullo stesso alfabeto di $\mathcal{A}$, in modo tale che

**$\mathcal{A}$ soddisfa la specifica $\mathcal{B}$ sse $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$**

Nella specifica di proprietà mediante automi è utile considerare automi in cui le **etichette sono formule**.

Un automa in cui le etichette sono formule costituisce una rappresentazione compatta di un automa con etichette in $2^P$.

Uno stato etichettato da una formula proposizionale $F$ è la rappresentazione compatta dell'insieme degli stati etichettati dai modelli di $F$.

**Esempio:** se il linguaggio è $P = \{p, q, r\}$, uno stato etichettato da $p \vee (q \wedge \neg r)$ è la rappresentazione compatta degli stati:

$$\begin{array}{ll} \{p\}, & \text{(modello di } p\text{)} \\ \{p, q\}, & \text{(modello di } p \text{ e di } q \wedge \neg r\text{)} \\ \{p, r\}, & \text{(modello di } p\text{)} \\ \{p, q, r\}, & \text{(modello di } p\text{)} \\ \{q\} & \text{(modello di } q \wedge \neg r\text{)} \end{array}$$

# Rappresentazione compatta di automi con etichette in $2^P$: stati etichettati da formule

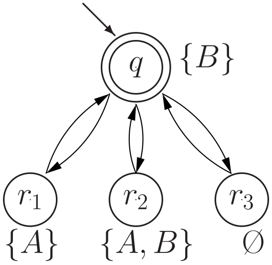
La specifica di una proprietà di un sistema può essere rappresentata da un automa di Büchi.

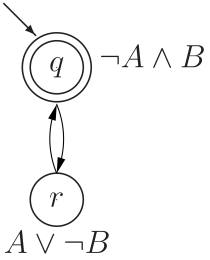
Ma in pratica è spesso inefficiente usare per le specifiche di proprietà automi dove ciascuno stato corrisponde a una singola assegnazione proposizionale.

Quando più stati hanno successori e predecessori in comune, si possono combinare in un unico stato, etichettato da una formula soddisfatta esattamente dalle assegnazioni degli stati combinati.

# Esempio

Sia $P = \{A, B\}$:




Le interpretazioni lette dall'automa sono sempre interpretazioni temporali: tutte le interpretazioni $\mathcal{M}$ tali che

$$\begin{array}{ll} \mathcal{M}_i \models \neg A \land B & \text{se } i \text{ è pari} \\ \mathcal{M}_i \models A \lor \neg B & \text{se } i \text{ è dispari} \end{array}$$

# Automi etichettati da formule

$$L : S \rightarrow 2^{2^P}$$

$L(s)$ è un insieme di assegnazioni proposizionali, rappresentato da una formula proposizionale classica con atomi in $P$

N.B. Il linguaggio accettato da un tale automa non cambia. Si tratta solo di una rappresentazione più compatta

Una parola $v : \mathbb{N} \rightarrow 2^P$ letta dall'automa corrisponde all'interpretazione temporale $\mathcal{M}$ tale che $\mathcal{M}(i) = v(i)$.

**Definizione di esecuzione:** una run $\rho$ su $v$ è un mapping $\mathbb{N} \rightarrow S$ tale che:

- $\rho(0) \in I$
- $\langle \rho(i), \rho(i+1) \rangle \in \Delta$, per ogni $i \geq 0$
- $v(i) \models L(\rho(i))$ per ogni $i \geq 0$ (cioè $\mathcal{M}_i \models L(\rho(i))$)

**Nota:** se uno stato $\rho(i)$ è etichettato da $\top$, allora per ogni "simbolo" $v(i)$:

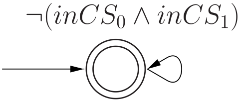
$$v(i) \models L(\rho(i))$$

# Semplificazione di un automa etichettato da formule

- eliminazione di stati etichettati da $\perp$ (non saranno mai in alcuna run di accettazione)
- eliminazione di nodi che non sono raggiungibili da alcuno stato iniziale

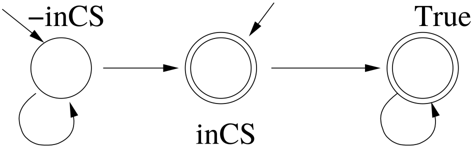
# Specifica di proprietà mediante BA etichettati da formule: esempi

**Mutual Exclusion:** i processi 1 e 2 non si trovano mai contemporaneamente nella propria "sezione critica": $\Box\neg(inCS_0 \wedge inCS_1)$




Ogni esecuzione $s_0, s_1, \ldots$ in $\mathcal{L}(\mathcal{A})$ deve essere accettata dall'automa $\mathcal{B}$ della specifica: per ogni $i$, $L_{\mathcal{A}}(s_i) \models \neg(inCS_0 \wedge inCS_1)$.

**Liveness** (qualcosa accade prima o poi): un processo prima o poi entrerà nella sua sezione critica: $\Diamond inCS$



# Verifica basata sulla teoria degli automi

La specifica di una proprietà di un sistema $\mathcal{A}$ si può dare mediante un automa $\mathcal{B}$ sullo stesso alfabeto di $\mathcal{A}$, in modo tale che

il modello $\mathcal{A}$ del sistema soddisfa la specifica rappresentata dall'automa $\mathcal{B}$ sse

$$\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$$

che equivale a

$$\mathcal{L}(\mathcal{A}) \cap \overline{\mathcal{L}(\mathcal{B})} = \emptyset$$

dove $\overline{\mathcal{L}(\mathcal{B})} = \Sigma^\omega - \mathcal{L}(\mathcal{B})$ è il complemento di $\mathcal{L}(\mathcal{B})$

(L'algoritmo per controllare l'inclusione tra BA è più complesso di quelli per costruire l'intersezione e controllare se il linguaggio di un BA è vuoto)

Gli automi di Büchi sono chiusi rispetto a unione, intersezione, complemento.

# Checking emptyness

$\mathcal{L}(\mathcal{A}) \neq \emptyset$ se e solo se $\mathcal{A}$ ha almeno un'esecuzione di accettazione: esiste $\rho$ tale che $\text{inf}(\rho) \cap F \neq \emptyset$

Questo vale se e solo se in $\mathcal{A}$ **esiste una componente fortemente connessa (SCC) raggiungibile da uno stato iniziale e contenente almeno uno stato di accettazione**.

# Operazioni sugli automi

Gli automi di Büchi sono chiusi rispetto a unione, intersezione, complemento: se $\mathcal{A}$ e $\mathcal{B}$ sono BA, esistono automi:

$$\begin{array}{c}\mathcal{A} \cup \mathcal{B} \text{ tale che } \mathcal{L}(\mathcal{A} \cup \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cup \mathcal{L}(\mathcal{B}) \\ \mathcal{A} \cap \mathcal{B} \text{ tale che } \mathcal{L}(\mathcal{A} \cap \mathcal{B}) = \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{B}) \\ \overline{\mathcal{A}} \text{ tale che } \mathcal{L}(\overline{\mathcal{A}}) = \overline{\mathcal{L}(\mathcal{A})}\end{array}$$

**Unione**

Siano: $\begin{array}{l}\mathcal{A}_1 = (\Sigma, S_1, \Delta_1, I_1, L_1, F_1) \\ \mathcal{A}_2 = (\Sigma, S_2, \Delta_2, I_2, L_2, F_2)\end{array}$

automi sullo stesso alfabeto $\Sigma$, con $S_1 \cap S_2 = \emptyset$ (altrimenti si rinominano gli stati).

$$\mathcal{A}_1 \cup \mathcal{A}_2 = (\Sigma, S_1 \cup S_2, \Delta_1 \cup \Delta_2, I_1 \cup I_2, L, F_1 \cup F_2)$$

con $L$ tale che $L(s) = \begin{cases} L_1(s) & \text{se } s \in S_1 \\ L_2(s) & \text{se } s \in S_2 \end{cases}$

# Automi di Büchi generalizzati (GBA)

Per costruire l'**intersezione** di due BA, abbiamo bisogno di generalizzare la nozione di BA.

In un GBA ci sono più insiemi di stati di accettazione

$$F = \{f_1, f_2, ..., f_m\} \text{ con } f_i \subseteq S$$

Un'esecuzione di accettazione $\rho$ passa infinite volte per almeno uno stato di ciascuno degli insiemi in $F$:

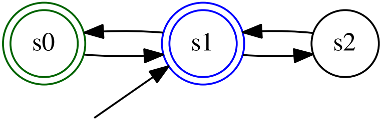
$$\mathbf{inf}(\rho) \cap f_i \neq \emptyset \text{ per ogni } f_i \in F$$

Ciascun elemento di $F$ rappresenta un vincolo di fairness

**Un GBA può dunque rappresentare un sistema su cui si impongono diversi vincoli di fairness**

Gli automi di Büchi generalizzati **non aggiungono potere espressivo**: è possibile tradurre un GBA in un BA che accetta lo stesso linguaggio

# Esempio



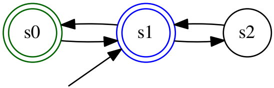

$F = \{f_1, f_2\}$ dove $f_1 = \{s_0\}$ e $f_2 = \{s_1\}$

Le esecuzioni di accettazione passano infinite volte sia per $s_0$ che per $s_1$.

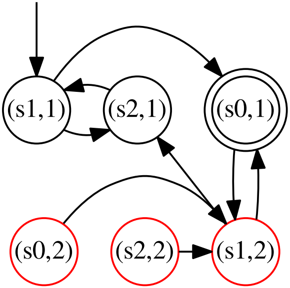
L'esecuzione $s_1 s_0 (s_1 s_2)^\omega$ non è di accettazione: non contiene infinite volte nessuno stato di $f_1$.

Il linguaggio accettato è $(s_1 (s_2 s_1)^* s_0)^\omega$.

# Traduzione di GBA in BA: esempio




$F = \{f_1, f_2\}$ con $f_1 = \{s_0\},\ f_2 = \{s_1\}$



# Traduzione di GBA in BA (I)

Sia $\mathcal{A} = (S, \Delta, I, L, \Sigma, \{f_1, ..., f_m\})$ un GBA.

La traduzione $\mathcal{A}^* = (S^*, \Delta^*, I^*, L^*, \Sigma, F)$ di $\mathcal{A}$ è il BA così definito:

**Stati.** $S^*$ contiene $m$ copie distinte di $S$:

$$S^* = S \times \{1, ..., m\} = \{(s, i) \mid s \in S,\ 1 \leq i \leq m\}$$

**Stati iniziali.** Quelli iniziali della prima copia: $I^* = I \times \{1\} = \{(s, 1) \mid s \in I\}$

**Alfabeto.** L'alfabeto è lo stesso di $\mathcal{A}$.

**Etichette.** $L^*((s, i)) = L(s)$

# Traduzione di GBA in BA (II)

**Relazione di transizione.** Si definisce l'operazione $i \oplus_m 1$ sull'insieme $\{1, ..., m\}$:

$$i \oplus_m 1 = \begin{cases} i+1 & \text{se } i < m \\ 1 & \text{se } i = m \end{cases} \qquad \text{cioè: } i \oplus_m 1 = (i \ mod \ m) + 1$$

Transizioni $\Delta^*$ nell'automa $\mathcal{A}^*$: per ogni transizione $(s, s')$ in $\Delta$, $\Delta^*$ contiene

$$\begin{array}{ll} ((s,i),(s', i \oplus_m 1)) & \text{se } s \in f_i \\ ((s,i),(s',i)) & \text{altrimenti} \end{array}$$

Cioè:

- se $s$ è nell'$i$-esimo insieme di accettazione, allora da $(s, i)$ si passa a $(s', i \oplus_m 1)$ – cioè si passa alla "copia" successiva
- altrimenti da $(s, i)$ si passa a $(s', i)$ – si resta nella stessa "copia".

In questo modo l'automa cicla sulle $m$ copie di $S$.

# Traduzione di GBA in BA (III)

**Stati di accettazione:** quelli della prima copia

$$F = f_1 \times \{1\} = \{(s, 1) \mid s \in f_1\}$$

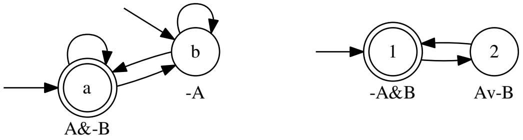
Se si passa per uno stato di $F$ infinite volte, vuol dire che ogni volta l'esecuzione è passata per tutte le copie di $S$, quindi passa per uno stato di accettazione di ogni copia infinite volte (per passare da una copia $S_i$ alla copia $S_{i \oplus_m 1}$ deve passare per uno stato di accettazione di $S_i$).

# Traduzione di GBA in BA: casi particolari

Consideriamo anche la traduzione di GBA in BA in due casi particolari:

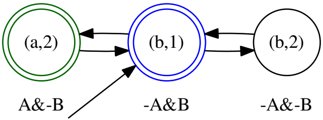
- un caso banale: se
$$\mathcal{A} = (S, \Delta, I, L, \Sigma, \{f_1\})$$
è un GBA con un solo insieme in $F$, allora la sua traduzione è il BA:
$$(S, \Delta, I, L, \Sigma, f_1)$$
- se uno degli insiemi di accettazione coincide con $S$, il GBA può essere prima semplificato: se
$$\mathcal{A} = (S, \Delta, I, L, \Sigma, \{S, f_1, ..., f_m\})$$
allora la sua traduzione è la traduzione di
$$(S, \Delta, I, L, \Sigma, \{f_1, ..., f_m\})$$

# Intersezione di automi etichettati da formule: esempio




Stati: prodotto cartesiano $\{a, b\} \times \{1, 2\}$

$$
\begin{aligned}
L(\langle a, 1\rangle) &= A \wedge \neg B \wedge \neg A \wedge B = \bot && \text{si elimina} \\
L(\langle a, 2\rangle) &= A \wedge \neg B \wedge (A \vee \neg B) = A \wedge \neg B \\
L(\langle b, 1\rangle) &= \neg A \wedge \neg A \wedge B = \neg A \wedge B \\
L(\langle b, 2\rangle) &= \neg A \wedge (A \vee \neg B) = \neg A \wedge \neg B
\end{aligned}
$$



# Intersezione di automi etichettati da formule (I)

Se $\mathcal{A}_1$ e $\mathcal{A}_2$ sono BA etichettati da formule (sullo stesso alfabeto), si definisce un GBA $A_1 \cap \mathcal{A}_2$, che accetta il linguaggio $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{B})$. Il GBA può essere poi trasformato in un BA che accetta lo stesso linguaggio.

Un'esecuzione su $A_1 \cap \mathcal{A}_2$ simula due esecuzioni simultanee, una su $A_1$ e una su $\mathcal{A}_2$

**Stati:** uno stato di $A_1 \cap \mathcal{A}_2$ contiene uno stato di $A_1$ e uno di $\mathcal{A}_2$:

$$S = S_1 \times S_2 = \{\langle s_1, s_2 \rangle \mid s_1 \in S_1 \text{ e } s_2 \in S_2\}$$

**Stati iniziali:** "contengono" stati iniziali di entrambi gli automi

$$\langle s_1, s_2 \rangle \in I \text{ sse } s_1 \in I_1 \text{ e } s_2 \in I_2$$

**Alfabeto:** l'alfabeto è lo stesso di $\mathcal{A}_1$ e $\mathcal{A}_2$.

**Etichette:** congiunzione delle etichette

$$L(\langle s_1, s_2 \rangle) = L_1(s_1) \wedge L_2(s_2)$$

# Intersezione di automi etichettati da formule (II)

**Relazione di transizione:**

$$(\langle q, r\rangle, \langle q', r'\rangle) \in \Delta \text{ sse } (q, q') \in \Delta_1 \text{ e } (r, r') \in \Delta_2$$

**Due insiemi di stati di accettazione:** uno contiene tutti gli stati $\langle s, s'\rangle$ con $s \in F_1$, l'altro gli stati $\langle s, s'\rangle$ con $s' \in F_2$:

$$F = \{f_1, f_2\} \text{ con } f_1 = F_1 \times S_2,\ f_2 = S_1 \times F_2$$

Vengono visitati infinitamente spesso sia stati di $F_1$ che stati di $F_2$ (ma non necessariamente contemporaneamente)

# Operazioni su BA e operatori logici

Unione, intersezione, complemento nei BA corrispondono a $\wedge, \vee, \neg$.

Una formula $F$ di LTL si può rappresentare mediante un BA $\mathcal{A}_F$ etichettato da formule (proposizionali classiche), tale che:

$$\mathcal{L}(\mathcal{A}_F) = \{\mathcal{M} \mid \mathcal{M} \models F\}$$

Si ha allora che:

$$\mathcal{L}(\mathcal{A}_{F\wedge G}) = \{\mathcal{M} \mid \mathcal{M} \models F \text{ e } \mathcal{M} \models G\} = \mathcal{L}(\mathcal{A}_F) \cap \mathcal{L}(\mathcal{A}_G)$$

$$\mathcal{L}(\mathcal{A}_{F\vee G}) = \{\mathcal{M} \mid \mathcal{M} \models F \text{ o } \mathcal{M} \models G\} = \mathcal{L}(\mathcal{A}_F) \cup \mathcal{L}(\mathcal{A}_G)$$

$$\mathcal{L}(\mathcal{A}_{\neg F}) = \{\mathcal{M} \mid \mathcal{M} \not\models F\} = \overline{\mathcal{L}(\mathcal{A}_F)}$$

# Verifica in teoria degli automi (I)

1. Il **sistema** viene modellato mediante un BA $\mathcal{A}$ **etichettato da formule**: se

$$\mathcal{A}' = (S, \Delta, I, L', 2^P, F)$$

è l'automa etichettato da insiemi di atomi che rappresenta il sistema, allora

$$\mathcal{A} = (S, \Delta, I, L, 2^{2^P}, F)$$

dove

$$L(s) = \bigwedge_{p \in L'(s)} p \wedge \bigwedge_{p \in P - L'(s)} \neg p$$

Ad esempio, se $P = \{p, q, r\}$ e $L'(s) = \{p, r\}$, allora $L(s) = p \wedge r \wedge \neg q$.

2. Sia $F$ la specifica che si vuole verificare per $\mathcal{A}$. $F$ può essere rappresentata da un automa $\mathcal{B}_F$ etichettato da formule, e se ne può poi costruire il complemento.
Ma la costruzione del complemento di un automa è difficile e costosa.
Si costruisce allora direttamente l'**automa $\mathcal{B}_{\neg F} = \overline{\mathcal{B}_F}$, che rappresenta la negazione della specifica** $\neg F$.

# Verifica in teoria degli automi (II)

**3.** Si costruisce il GBA intersezione

$$\mathcal{G} = \mathcal{A} \cap \mathcal{B}_{\neg F}$$

**4.** Si controlla se il linguaggio di $\mathcal{G}$ è vuoto oppure no:

**Il sistema soddisfa la specifica sse $\mathcal{L}(\mathcal{G}) = \emptyset$**

# Checking emptiness + identificazione di un controesempio

Come controllare se $\mathcal{L}(\mathcal{G}) = \emptyset$ e fornire un controesempio in caso di risposta negativa.

$$\mathcal{G} = \mathcal{A} \cap \mathcal{B}_{\neg F} = (\Sigma, S, \Delta, I, L, F)$$

Assumiamo che tutte le etichette in $\mathcal{G}$ siano diverse da $\perp$ (gli stati etichettati da $\perp$ sono stati eliminati)

**Se $\mathcal{G}$ è un GBA, con insiemi di accettazione $F = \{f_1, ..., f_m\}$,**
**allora $\mathcal{L}(\mathcal{G}) \neq \emptyset$ se e solo se**

> **esiste un ciclo $s_1, s_2, ..., s_k, s_1$ (una SSC $\{s_1, s_2, ..., s_k\}$),**
> **raggiungibile da uno stato iniziale,**
> **che contiene uno stato di ciascun insieme $f_i$:**

$$\{s_1, s_2, ..., s_k\} \cap f_i \neq \emptyset \text{ per ogni } i = 1, ..., m$$

# Osservazione

In generale, se $A$ è una formula proposizionale, controllare se $A \leftrightarrow \perp$ è NP-completo.

Tuttavia:

- $\mathcal{A}$ è etichettato da congiunzioni di letterali;
- per qualsiasi formula temporale $F$, è possibile costruire un BA $\mathcal{B}_{\neg F}$, che rappresenta $\neg F$, in cui tutte le etichette sono congiunzioni di letterali;
- l'intersezione di automi conserva questa proprietà.

Se $A$ è una congiunzione di letterali, allora controllare se $A \leftrightarrow \perp$ richiede tempo lineare.

# La ricerca di componenti fortemente connesse in un grafo

Per trovare le componenti fortemente connesse si può utilizzare una visita in profondità.

Se $\mathcal{L}(\mathcal{G}) \neq \emptyset$ la visita produce un'esecuzione $\rho$ in $\mathcal{L}(\mathcal{G})$ che si può rappresentare in modo finito:

$\rho$ è costituita da un prefisso $\sigma_1$ finito, seguito da una sequenza periodica di stati:

$$\rho = \sigma_1 \, (\sigma_2)^\omega \text{ con } \sigma_1, \sigma_2 \text{ finite}$$

($\rho$ è una sequenza **ultimately periodic**).

# Formule LTL e automi

- Ogni formula $F$ di LTL può essere "tradotta" in un automa $\mathcal{B}_F$ (etichettato da congiunzioni di letterali),
  che accetta esattamente i modelli di $F$:

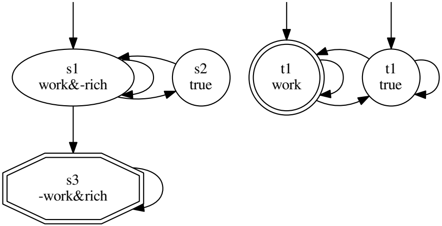
$$\mathcal{L}(\mathcal{B}_F) = \{\mathcal{M} \mid \mathcal{M} \models F\}$$

- Se $\mathcal{L}(\mathcal{B}_F) \neq \emptyset$ allora $\mathcal{B}_F$ ha un'esecuzione di accettazione *ultimately periodic*.

**Quindi:** se una formula LTL è soddisfacibile, allora ha un modello **ultimately periodic**.

# Esercizio

Si considerino i due automi $\mathcal{A}_1$ e $\mathcal{A}_2$ sotto rappresentati:




1. Costruire l'intersezione $\mathcal{A} = \mathcal{A}_1 \cap \mathcal{A}_2$ dei due automi;
2. Quale formula temporale rappresenta l'automa $\mathcal{A}_2$ (di destra)?
3. Il linguaggio di $\mathcal{A}$ è vuoto? Quale specifica si può dunque dire che sia soddisfatta o non soddisfatta dal sistema rappresentato dall'automa $\mathcal{A}_1$?
4. Trasformare l'automa generalizzato $\mathcal{A}$ in un automa semplice equivalente $\mathcal{A}'$. Il linguaggio di $\mathcal{A}'$ è vuoto?
5. Considerare l'automa $\mathcal{A}'_1$ che è come $\mathcal{A}_1$ per tutte le componenti, tranne che non vi sono vincoli di fairness ($F = \{s1, s2, s3\}$). Costruire $\mathcal{A} = \mathcal{A}'_1 \cap \mathcal{A}_2$ e verificare se il suo linguaggio è vuoto o no.