L’algoritmo di Safra

Angelo Montanari (e Alberto Molinari)

Department of Mathematics, Computer Science, and Physics
Università degli Studi di Udine

Riferimento bibliografico principale

Dominique Perrin and Jean-Eric Pin.
Infinite words: automata, semigroups, logic and games (Chapter 1).
Pure and Applied Mathematics Series. Elsevier, 2004.

Introduzione

Büchi DET
$\{\alpha \in \{a, b\}^\omega \mid \cancel{\exists}^{<\omega} n \; \alpha(n) = b\}$
non chiusi per $\neg$

Büchi ND
chiusi per $\neg$

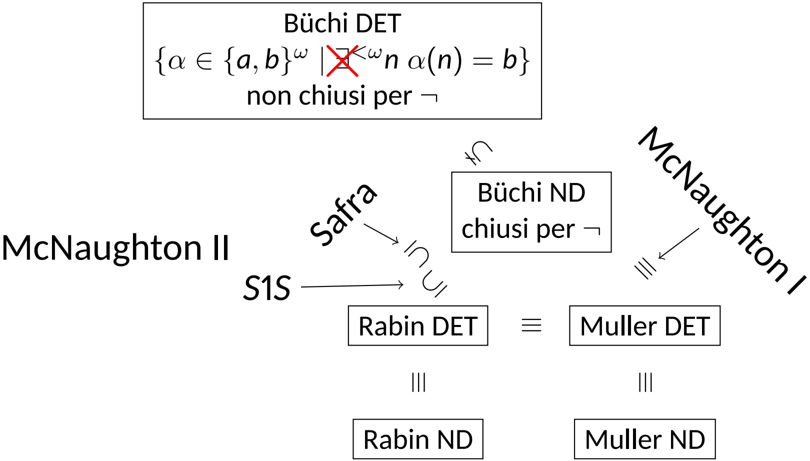
Rabin DET

Muller DET

Rabin ND

Muller ND

Introduzione



Definizioni di Base

- Automa di Büchi

  è una quintupla $\mathcal{A} = (Q, A, \Delta, I, F)$ dove:
  - Q è un insieme di stati
  - A è un alfabeto
  - $\Delta \subseteq Q \times A \times Q$ è la relazione di transizione
  - $I \subseteq Q$ è l'insieme degli stati iniziali
  - $F \subseteq Q$ è l'insieme degli stati finali

  Una computazione σ si dice *di successo* se (e solo se):

$$In(\sigma) \cap F \neq \emptyset$$

Definizioni di Base

- Automa di Rabin

  è una quintupla $\mathcal{D} = (Q, A, \delta, i, \Omega)$ dove Q e A sono definiti come negli automi di Büchi, δ è la funzione di transizione, i lo stato iniziale e

  $$\Omega = \{(L_1, U_1), \ldots, (L_n, U_n)\}$$

  con $L_i, U_i \subseteq Q \ \forall i = 1, \ldots n$.

  Una computazione σ si dice *di successo* se esiste j tale che:
  - $In(\sigma) \cap L_j = \emptyset$
  - $In(\sigma) \cap U_j \neq \emptyset$

Teorema di McNaughton

Teorema (McNaughton)

Un ω-linguaggio $L \subseteq A^\omega$ è riconosciuto da un automa di Büchi (i.e., è ω-regolare) se e soltanto se è riconosciuto da un automa di Rabin.

Per dimostrare $\Rightarrow$ si utilizza l'algoritmo di Safra: un algoritmo che, a partire da un automa di Büchi (non deterministico), costruisce un automa di Rabin (deterministico) equivalente.

L'algoritmo di determinizzazione classico non funziona - I

L'algoritmo di determinizzazione per gli automi a stati finiti su parole finite non funziona con le parole infinite. Ciò segue immediatamente dall'inclusione stretta degli automi di Büchi deterministici negli automi di Büchi non deterministici. È, però, utile capire dove la costruzione viene meno.

L'algoritmo di determinizzazione classico non funziona - I

L'algoritmo di determinizzazione per gli automi a stati finiti su parole finite non funziona con le parole infinite. Ciò segue immediatamente dall'inclusione stretta degli automi di Büchi deterministici negli automi di Büchi non deterministici. È, però, utile capire dove la costruzione viene meno.

Sia $\alpha \in A^\omega$ una ω-parola riconosciuta dall'automa di Büchi $\mathcal{A}$.
Dalla condizione di accettazione degli automi di Büchi segue l'esistenza di uno stato $f \in F$, raggiungibile da uno stato $i \in I$, tale che

$$i \longrightarrow_{\alpha_0} f \longrightarrow_{\alpha_1} f \longrightarrow_{\alpha_2} f \cdots$$

dove $\alpha = \alpha_0 \cdot \alpha_1 \cdot \alpha_2 \cdots$

L'algoritmo di determinizzazione classico non funziona - II

Sia $\mathcal{B}$ l'automa (deterministico) di Büchi $(\mathcal{P}(Q), A, \delta, \{I\}, \mathcal{F})$, dove $\delta(S, a) = \Delta(S, a)(\in \mathcal{P}(Q))$, con $S \subseteq Q$, $a \in A$ e $\mathcal{F} = \{P : P \cap F \neq \emptyset\}$.

L'algoritmo di determinizzazione classico non funziona - II

Sia $\mathcal{B}$ l'automa (deterministico) di Büchi $(\mathcal{P}(Q), A, \delta, \{I\}, \mathcal{F})$, dove $\delta(S, a) = \Delta(S, a)(\in \mathcal{P}(Q))$, con $S \subseteq Q$, $a \in A$ e $\mathcal{F} = \{P : P \cap F \neq \emptyset\}$.

La computazione dell'automa di Büchi (deterministico) $\mathcal{B}$ su α può essere espressa nel modo seguente:

$$I \longrightarrow_{\alpha_0} S_0 \longrightarrow_{\alpha_1} S_1 \longrightarrow_{\alpha_2} S_2 \cdots$$

dove $f \in S_i$, per ogni $i \geq 0$, e, pertanto, $S_i \in \mathcal{F}$. La computazione di $\mathcal{B}$ è quindi di successo.

L’algoritmo di determinizzazione classico non funziona - II

Sia $\mathcal{B}$ l’automa (deterministico) di Büchi $(\mathcal{P}(Q), A, \delta, \{I\}, \mathcal{F})$, dove $\delta(S, a) = \Delta(S, a)(\in \mathcal{P}(Q))$, con $S \subseteq Q$, $a \in A$ e $\mathcal{F} = \{P : P \cap F \neq \emptyset\}$.

La computazione dell’automa di Büchi (deterministico) $\mathcal{B}$ su α può essere espressa nel modo seguente:

$$I \longrightarrow_{\alpha_0} S_0 \longrightarrow_{\alpha_1} S_1 \longrightarrow_{\alpha_2} S_2 \cdots$$

dove $f \in S_i$, per ogni $i \geq 0$, e, pertanto, $S_i \in \mathcal{F}$. La computazione di $\mathcal{B}$ è quindi di successo.

Vale anche il viceversa? (Ossia se la computazione di $\mathcal{B}$ è di successo, allora anche $\mathcal{A}$ ammette una computazione di successo?)

L'algoritmo di determinizzazione classico non funziona - II

Sia $\mathcal{B}$ l'automa (deterministico) di Büchi $(\mathcal{P}(Q), A, \delta, \{I\}, \mathcal{F})$, dove $\delta(S, a) = \Delta(S, a)(\in \mathcal{P}(Q))$, con $S \subseteq Q$, $a \in A$ e $\mathcal{F} = \{P : P \cap F \neq \emptyset\}$.

La computazione dell'automa di Büchi (deterministico) $\mathcal{B}$ su α può essere espressa nel modo seguente:

$$I \longrightarrow_{\alpha_0} S_0 \longrightarrow_{\alpha_1} S_1 \longrightarrow_{\alpha_2} S_2 \cdots$$

dove $f \in S_i$, per ogni $i \geq 0$, e, pertanto, $S_i \in \mathcal{F}$. La computazione di $\mathcal{B}$ è quindi di successo.

Vale anche il viceversa? (Ossia se la computazione di $\mathcal{B}$ è di successo, allora anche $\mathcal{A}$ ammette una computazione di successo?)

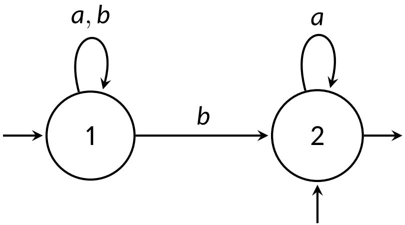
No, perché, per ogni $i \geq 1$, non vi è alcuna garanzia che lo stato f presente in S_i sia raggiungibile dallo stato f presente in S_{i-1}.

Un controesempio

Si consideri il caso dell'automa di Büchi ND che riconosce il linguaggio

$$L = \{\alpha \in A^\omega.\ \exists^{<\omega} n\ \alpha(n) = b\}$$

con $A = \{a, b\}$.

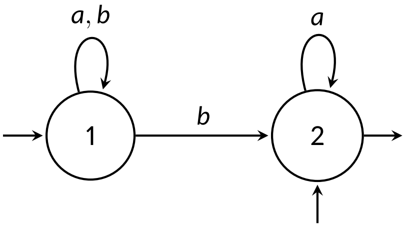


Un controesempio

Si consideri il caso dell'automa di Büchi ND che riconosce il linguaggio

$$L = \{\alpha \in A^\omega .\ \exists^{<\omega} n\ \alpha(n) = b\}$$

con $A = \{a, b\}$.



L'automa non accetta b^ω. Tuttavia la computazione su b^ω della versione determinizzata dell'automa è

$$\{1, 2\} \longrightarrow_b \{1, 2\} \longrightarrow_b \{1, 2\} \longrightarrow_b \cdots$$

ed è quindi di successo.

Una possibile via d'uscita - I

Per rimuovere il problema evidenziato, è necessario identificare una sequenza

$$I \longrightarrow_{\alpha_0} S_0 \longrightarrow_{\alpha_1} S_1 \longrightarrow_{\alpha_2} S_2 \cdots$$

che soddisfi le seguenti due condizioni:

1. $S_0 \subseteq \Delta(I, \alpha_0)$ e, per ogni $n \geq 0$, $S_{n+1} \subseteq \Delta(S_n, \alpha_{n+1})$;
2. per ogni $n \geq 0$ e ogni $q \in S_{n+1}$, esistano uno stato $p \in S_n$ e un cammino $p \longrightarrow_{\alpha_{n+1}} q$ in $\mathcal{A}$ che attraversa uno stato finale.

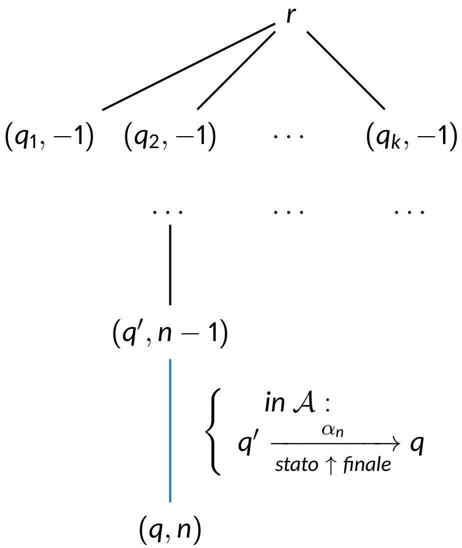
Se le condizioni sono verificate, $\mathcal{A}$ ammette una computazione di successo su $\alpha = \alpha_0\alpha_1\alpha_2\cdots$. Slide 45

Una possibile via d'uscita - II

In modo un po' più formale, al fine di poter applicare il lemma di König, costruiamo un albero (N, r, E), dove:

- $N = \{r\} \cup \{(q, -1) \mid q \in I\} \cup \{(q, n) \mid q \in S_n, n \in \mathbb{N}\}$;
- tutti i nodi della forma $(q, -1)$ hanno come padre r;
- per $n \geq 0$, ogni nodo della forma (q, n) ha come padre un nodo della forma $(q', n-1)$ tale che esiste un percorso in $\mathcal{A}$, etichettato con α_n, che parte in q', termina in q e, se $n \geq 1$, deve visitare almeno uno stato finale.

Una possibile via d'uscita - III



Una possibile via d'uscita - IV

Ogni nodo ha un numero finito di figli, ma l'albero ha infiniti nodi. Ne segue che, per il lemma di König, esiste almeno un ramo infinito

$$\Downarrow$$

Esiste un percorso infinito in $\mathcal{A}$ etichettato con α che parte da *I* e attraversa infinite volte uno stato finale.

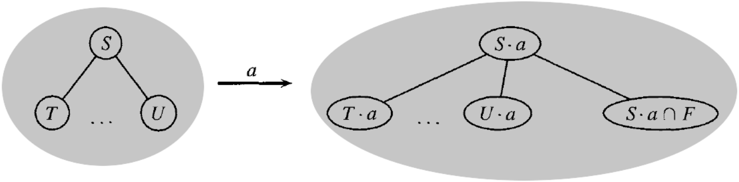
$$\Downarrow$$

α è accettata da $\mathcal{A}$

Una possibile via d'uscita - V

Per trovare tale cammino, costruiremo un automa (di Rabin) che memorizza le occorrenze degli stati finali.

Gli stati di tale automa sono alberi orientati, i cui nodi vengono etichettati con gli insiemi S_i.



Algoritmo di Safra

Descriviamo formalmente l'algoritmo.

Sia $\mathcal{A}$ un generico automa di Büchi (non deterministico):

$$\mathcal{A} = (Q, A, \Delta, I, F)$$

dove $|Q| = n$.

Costruiamo un automa di Rabin (deterministico) equivalente

$$\mathcal{D} = (S, A, \delta, i, \Omega)$$

Stati - I

Gli stati dell'automa di Rabin sono alberi ordinati etichettati, con alcuni nodi marcati.

Sia $V = \{1, 2, \ldots, 2n\}$.

Uno stato (albero) è una quadrupla (T, f, e, M) dove:

- $T \subseteq V$ è l'insieme dei nodi;
- $f : T \to T^*$ è una funzione che mappa ogni nodo nella sequenza ordinata dei suoi figli;
- $e : T \to 2^Q \setminus \emptyset$ è una funzione che associa un'etichetta (un sottoinsieme non vuoto di Q) ad ogni nodo;
- $M \subseteq T$ è l'insieme dei nodi marcati.

Stati - II

Restringiamo la nostra attenzione agli alberi che soddisfano le seguenti condizioni:

1. la radice dell'albero è 1;
2. i nodi marcati devono essere foglie dell'albero;
3. per ogni nodo v, l'unione delle etichette dei figli è un sottoinsieme stretto dell'etichetta di v $(e(v))$;
4. se il nodo v non è un antenato del nodo w e il nodo w non è un antenato del nodo v, allora $e(v) \cap e(w) = \emptyset$.

Indichiamo con $\mathcal{T}_n$ l'insieme di tutti gli alberi definiti in questo modo. Esso costituirà l'insieme S degli stati dell'automa di Rabin $\mathcal{D}$.

Stati - III

Proposizione

Un albero $(T, f, e, M) \in \mathcal{T}_n$ *ha al più n nodi.*

Stati - III

Proposizione

Un albero $(T, f, e, M) \in \mathcal{T}_n$ ha al più n nodi.

Sia:

$$r(v) = e(v) \setminus \bigcup_{w \text{ figlio di } v} e(w)$$

Per la condizione 3, tale insieme non può essere vuoto. Segue che:

$$|T| = \sum_{v \in T} 1 \leq \sum_{v \in T} |r(v)|$$

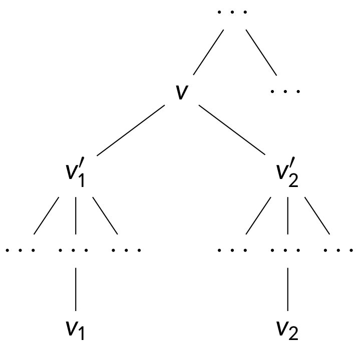
Inoltre, se v_1 e v_2 sono due nodi distinti, allora $r(v_1) \cap r(v_2) = \emptyset$ (ciò segue direttamente dalla condizione 4, se i due nodi non sono uno l'antenato dell'altro; altrimenti, segue dalla definizione di r).
Possiamo concludere che gli insiemi $r(v)$ sono a due a due disgiunti, da cui:

$$|T| = \sum_{v \in T} 1 \leq \sum_{v \in T} |r(v)| \leq |Q| = n$$

Stati - IV

Definizione (Ordine esteso dei nodi in $\mathcal{T}_n$)

Siano v_1 e v_2 due nodi che non sono uno un antenato dell'altro e sia v il loro minimo antenato comune. Inoltre, siano v_1' e v_2' i figli di v antenati rispettivamente di v_1 e v_2. Diciamo che v_1 è a sinistra di v_2 se $v_1' < v_2'$ (i figli di ogni nodo sono ordinati).



Funzione di Transizione - I

Sia $R = (T, f, e, M)$ uno stato (albero) in $\mathcal{T}_n$ e sia a un elemento dell'alfabeto A.

Lo stato $\delta(R, a)$ viene calcolato nel seguente modo:

1. si esegue la transizione dell'automa di Büchi su a per tutti gli elementi dell'etichetta di ogni nodo e si cancellano gli eventuali marcatori presenti, ottenendo l'albero (T, f, e_1, M_1), con $M_1 = \emptyset$ e $e_1(v) = \Delta(e(v), a)$ per ogni v;
2. si aggiunge ad ogni nodo v un nuovo figlio, a destra dell'ultimo figlio, con etichetta $e_1(v) \cap F$ e si marca tale nodo. Tale nodo è scelto in modo arbitrario fra i nodi disponibili (assumiamo di scegliere il nodo più piccolo fra i nodi disponibili).

Funzione di Transizione - II

3. nell'etichetta di ogni nodo v vengono eliminati gli stati che appaiono nelle etichette dei nodi a sinistra di v.
4. si eliminano i nodi con etichetta vuota, aggiornando coerentemente la funzione f e la marcatura.
5. si marcano i nodi che hanno come etichetta l'unione delle etichette dei loro figli, e si eliminano tutti i discendenti.

Sia $(T_5, f_5, e_5, M_5) = \delta(R, a)$ l'albero così ottenuto.

Non è difficile verificare che tale albero soddisfa tutte le condizioni precedenti (è in $\mathcal{T}_n$).

Stato iniziale

Tre casi:

- se $I \cap F = \emptyset$, lo stato iniziale è un albero formato da un nodo non marcato etichettato con I
- se $I \subseteq F$, lo stato iniziale è un albero formato da un nodo marcato etichettato con I
- altrimenti, è un albero formato da un nodo non marcato etichettato con I, con un figlio marcato etichettato con $I \cap F$

Condizione di Accettazione

L'insieme Ω è definito nel seguente modo:

$$\Omega = \{(L_v, U_v) \mid v \in V\}$$

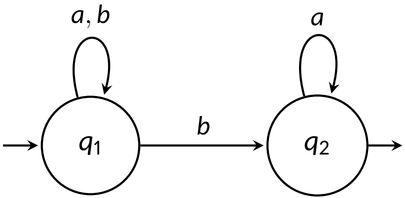
dove:

- $L_v = \{R \in \mathcal{T}_n \mid v \textit{ non è un nodo di } R\}$
- $U_v = \{R \in \mathcal{T}_n \mid v \textit{ è un nodo marcato di } R\}$

La computazione dell'automa $\mathcal{D}$ è pertanto di successo se esiste un nodo $v \in V$ tale che la computazione attraversa finite volte gli stati in cui v non è presente e infinite volte almeno uno stato in cui v è un nodo marcato.

Esempio - I

Sia $\mathcal{A}$ l'automa di Büchi sull'alfabeto $A = \{a, b\}$ che riconosce tutte e sole le ω-parole con un numero finito, maggiore di 0, di occorrenze del simbolo b:

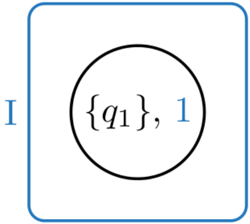


- $Q = \{q_1, q_2\}$
- $A = \{a, b\}$
- $I = \{q_1\}$
- $F = \{q_2\}$

Esempio - II

Per trovare l'automa di Rabin equivalente, eseguiamo l'algoritmo descritto in precedenza.

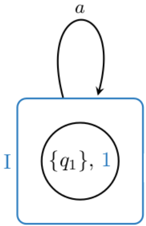
Stato iniziale:



$I \cap F = \emptyset \Rightarrow$ albero con un solo nodo non marcato, etichettato con $I = \{q_1\}$.

Esempio - III

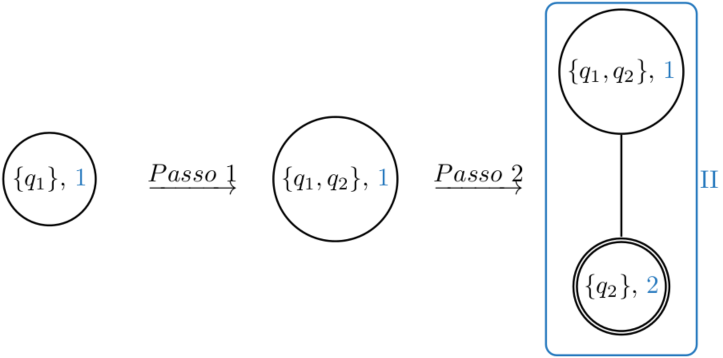
Esecuzione dell'azione a sullo stato iniziale:



Eseguiamo la transizione dell'automa di Büchi per ogni stato dell'etichetta e eliminiamo gli eventuali marcatori presenti. In questo caso, l'albero non viene modificato. Allo stesso modo, i passi successivi non modificano l'albero.

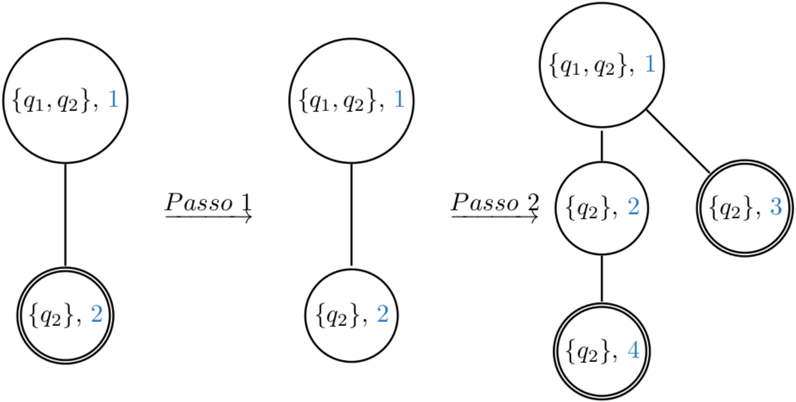
Esempio - IV

Esecuzione dell'azione b sullo stato iniziale:



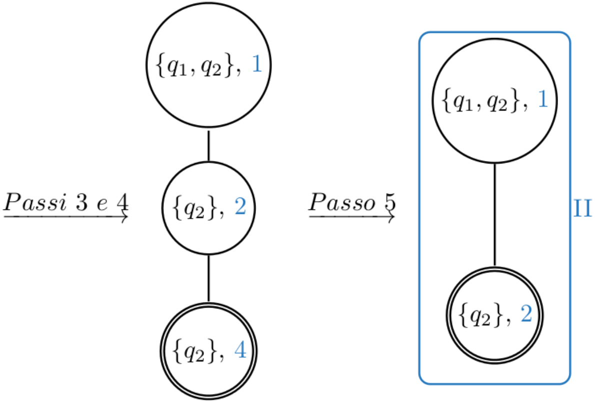
Esempio - V

Esecuzione dell'azione a sullo stato *II*:



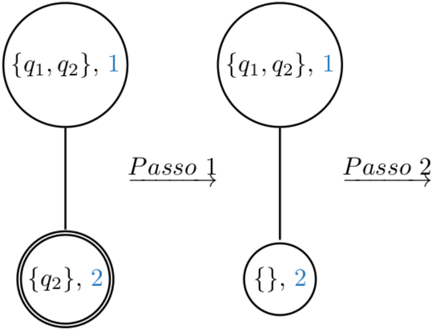
Esempio - VI

Esecuzione dell'azione a sullo stato *II*:



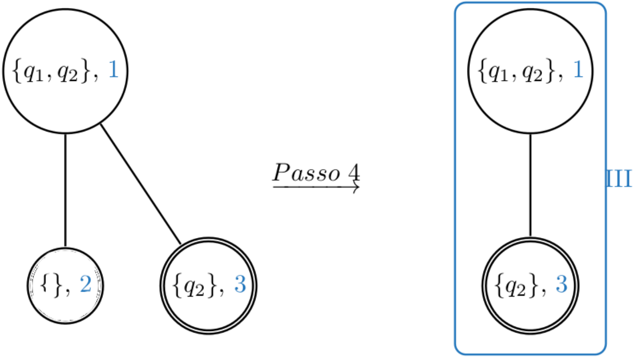
Esempio - VII

Esecuzione dell'azione b sullo stato *II*:



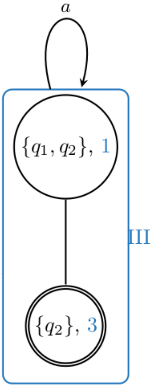
Esempio - VIII

Esecuzione dell'azione $\mathtt{b}$ sullo stato *II*:



Esempio - IX

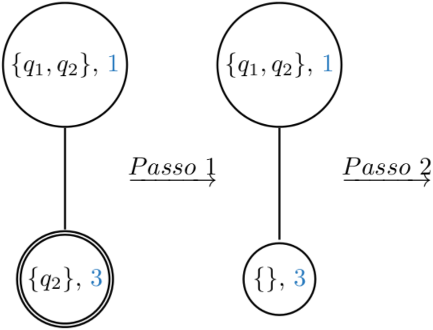
Esecuzione dell'azione a sullo stato *III*:



I passi che vengono eseguiti dall'algoritmo sono analoghi a quelli prodotti dall'esecuzione dell'azione *a* sullo stato *II*. Lo stato risultante è lo stato *III*.

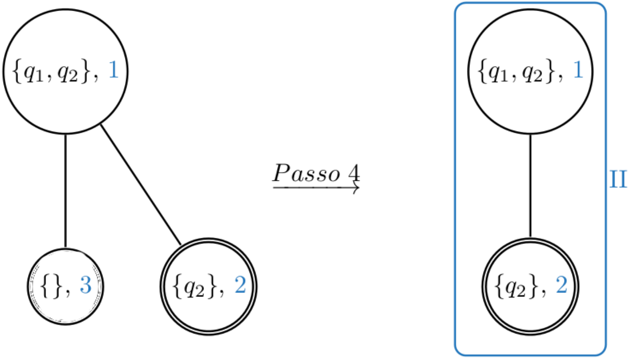
Esempio - X

Esecuzione dell'azione b sullo stato *III*:



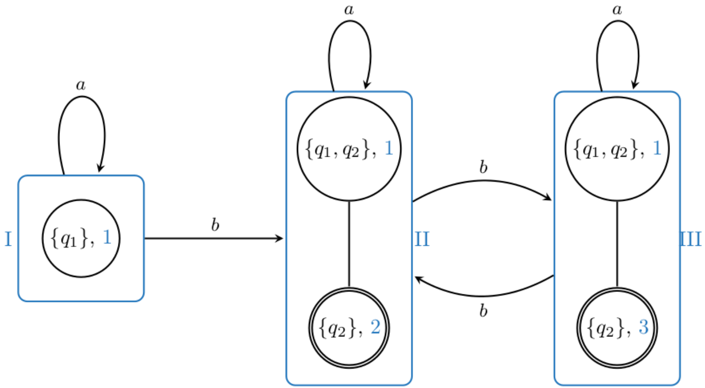
Esempio - XI

Esecuzione dell'azione b sullo stato *III*:



Esempio - XII

Automa risultante:



Esempio -XIII

Condizione di accettazione:

$$L_1 = \emptyset$$
$$L_2 = \{I, III\}$$
$$L_3 = \{I, II\}$$

$$U_1 = \emptyset$$
$$U_2 = \{II\}$$
$$U_3 = \{III\}$$

$$\mathcal{F} = \{(\{I, III\}, \{II\}), (\{I, II\}, \{III\})\}$$

Equivalenza dei due automi

Dimostriamo che l'automa di Rabin $\mathcal{D}$, ottenuto tramite l'esecuzione dell'algoritmo di Safra, è equivalente all'automa di Büchi iniziale $\mathcal{A}$.

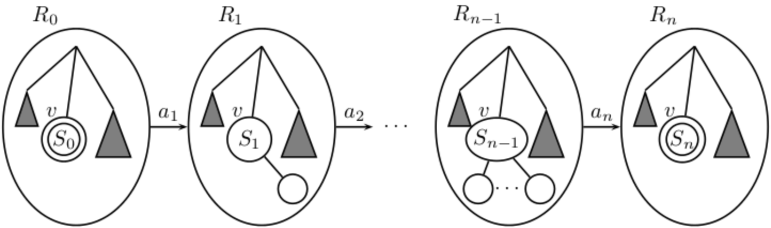
Come primo passo, dimostriamo un lemma che prova una proprietà fondamentale dell'algoritmo di Safra.

Lemma

- *Sia* $u = a_1 a_2 \cdots a_n$ *una parola finita;*
- *sia* R_0 *uno stato (albero) di* $\mathcal{D}$ *che contiene un nodo marcato* v *etichettato con* S_0*;*
- *per* $1 \leq i \leq n$*, gli stati* $R_i = \delta(R_0, a_1 \cdots a_i)$ *contengano il nodo* v *con etichetta* S_i *e tale nodo sia marcato solamente per* $i = n$*.*

Vale che:

1. *per* $0 \leq i \leq n-1$*,* S_{i+1} *è contenuto in* $\Delta(S_i, a_{i+1})$*;*
2. *per ogni* $q \in S_n$*, esiste in* $\mathcal{A}$ *una computazione, etichettata con* u*, che inizia in* S_0*, termina in* q *e visita almeno uno stato finale (dopo la sua origine).*



Dimostrazione di correttezza - I

Seguiamo passo passo l'esecuzione dell'algoritmo nello stato R_i (relativamente al nodo v con etichetta S_i).
Al primo passo, si ottiene

$$S_{i+1} = \Delta(S_i, a_{i+1})$$

Al terzo passo vengono eventualmente soppressi alcuni stati di S_{i+1}.

Il primo punto del lemma è dimostrato.

Dimostrazione di correttezza - II

Per quanto riguarda il secondo punto del lemma, dimostriamo, per induzione su i, che:

per ogni $0 \leq i \leq n-1$ e per ogni stato q_i che compare nell'etichetta di un discendente di v in R_i, esiste una computazione di $\mathcal{A}$, che legge la stringa $a_1, \cdots, a_i$, che inizia in S_0, termina in q_i e attraversa almeno uno stato finale.

Dimostrazione di correttezza - III

$i = 0$: v è marcato e deve pertanto essere una foglia di R_0 (priva di discendenti). La tesi vale banalmente.

Dimostrazione di correttezza - III

$i = 0$: v è marcato e deve pertanto essere una foglia di R_0 (priva di discendenti). La tesi vale banalmente.

$i > 0$: sia q_{i+1} uno stato che appare nell'etichetta di un discendente di v in R_{i+1}. Sono possibili due casi:

- $q_{i+1} \in \Delta(q_i, a_{i+1})$ per qualche q_i appartenente all'etichetta di un diretto discendente di v in R_i (la tesi segue dall'ipotesi induttiva).
- altrimenti, q_{i+1} deriva da un'etichetta creata al secondo passo dell'algoritmo (aggiunta di un figlio, a destra di tutti gli altri, contenente, per costruzione, solo stati finali). Ne segue immediatamente che $q_{i+1} \in F$, da cui la tesi.

Dimostrazione di correttezza - IV

Infine, dal momento che v è marcato in R_n, esso viene necessariamente marcato nel quinto passo dell'algoritmo.

Ne segue che se $q \in S_n$, allora o

- $q \in \Delta(S_{n-1}, a_n) \cap F$, ove S_{n-1} è l'etichetta di v in R_{n-1} (q è finito al passo 2 nel figlio "nuovo" di v, "riassorbito" poi al passo 5), oppure
- q appartiene all'unione dei $\Delta(q_{n-1}, a_n)$, dove q_{n-1} compare nell'etichetta di un discendente di v in R_{n-1} (q è in un figlio "vecchio" di v, "riassorbito" al passo 5).

Nel primo caso, esiste un cammino, etichettato con u, che inizia in S_0 e finisce in q, che è uno stato finale.

Nel secondo caso, possiamo sfruttare la conclusione dell'induzione precedente: esiste un cammino in $\mathcal{A}$ che inizia in S_0 e termina in q_{n-1}, etichettato con $a_1 \cdots a_{n-1}$, che attraversa almeno una volta uno stato finale. La tesi segue immediatamente.

Dimostrazione di correttezza - V

Sulla scorta del lemma precedente, siamo ora in grado di dimostrare il Teorema di McNaughton.

Sia $\mathcal{A} = (Q, A, \Delta, I, F)$ un automa di Büchi e sia $\mathcal{D} = (S, A, \delta, i, \Omega)$ il corrispondente automa di Rabin costruito mediante l'algoritmo di Safra.

Dimostriamo che i due automi sono equivalenti, ossia che per ogni ω-parola α in A^ω:

$$\text{esiste una computazione di successo di } \mathcal{A} \text{ su } \alpha$$
$$\Updownarrow$$
$$\text{la computazione di } \mathcal{D} \text{ su } \alpha \text{ è di successo}$$

Dimostrazione di correttezza - VI

$\Leftarrow$ Sia c la computazione di successo di $\mathcal{D}$ su $\alpha \in A^\omega$.

$$\Downarrow$$

esiste un nodo $v \in V$ tale che, da un certa posizione in poi, diciamo k, c visita solo stati dei quali v è un nodo; inoltre, visita infinite volte uno o più stati in cui v è un nodo marcato. Siano $j_0, j_1, \ldots$ tutte e sole le (infinite) posizioni $\geq k$ in cui tali stati sono visitati da c.

Dimostrazione di correttezza - VI

$\Leftarrow$ Sia c la computazione di successo di $\mathcal{D}$ su $\alpha \in A^\omega$.

$$\Downarrow$$

esiste un nodo $v \in V$ tale che, da un certa posizione in poi, diciamo k, c visita solo stati dei quali v è un nodo; inoltre, visita infinite volte uno o più stati in cui v è un nodo marcato. Siano $j_0, j_1, \ldots$ tutte e sole le (infinite) posizioni $\geq k$ in cui tali stati sono visitati da c.

Per il lemma precedente, posto $S_{-1} = I$, esistono una fattorizzazione $\alpha = u_0 u_1 \cdots$, dove $u_0 = \alpha(0, j_0 - 1)$ e $u_i = \alpha(j_{i-1}, j_i - 1)$, per ogni $i > 0$, e una sequenza di insiemi $S_i \subseteq Q$, dove S_i è l'insieme $e(v)$ nello stato $c(j_i)$, per ogni $i \geq 0$, tali che:

a. $\forall n \geq 0 \; S_n \subseteq \Delta(S_{n-1}, u_n)$
b. $\forall n \geq 0$ e $\forall q \in S_n$ esiste una computazione in $\mathcal{A}$, etichettata con u_n, che inizia in S_{n-1}, termina in q e visita almeno uno stato finale

$\Rightarrow$ Abbiamo provato le 2 condizioni della Slide 10

Dimostrazione di correttezza - VII

$\Longrightarrow$ Sia c una computazione di successo dell'automa $\mathcal{A}$ su α. Indichiamo con $c(i) \in Q$ lo stato raggiunto da $\mathcal{A}$ al passo i-esimo della computazione

$$c : c(0) \xrightarrow{a_0} c(1) \xrightarrow{a_1} c(2) \cdots$$

Esiste un'unica computazione d di $\mathcal{D}$ su α. Indichiamo con $d(i) \in S$ lo stato (albero) dell'automa $\mathcal{D}$ raggiunto al passo i-esimo.

$$d : d(0)(= I) \xrightarrow{a_0} d(1) \xrightarrow{a_1} d(2) \cdots$$

Ogni $c(i) \in Q$ appartiene all'etichetta della radice dell'albero $d(i)$ (primo passo dell'algoritmo). Tale radice non viene mai soppressa e risulta pertanto essere un elemento fisso v_0 di V.

Dimostrazione di correttezza - VIII

- Se v_0 compare marcato in un'infinità di stati/alberi $d(i)$, allora il percorso è di successo in $\mathcal{D}$ e la ω-parola α viene accettata.

- Altrimenti esiste un numero intero n_0 tale che n_0 è il numero più grande per il quale $d(n_0)$ ha la radice v_0 marcata.

$$d : \cdots \longrightarrow d(n_0) \longrightarrow d(n_0+1) \longrightarrow \cdots$$

Dimostrazione di correttezza - IX

Sia $m > n_0$ il più piccolo numero intero tale che $c(m) = q_f$ è uno stato finale ripetuto infinite volte.

$$\cdots \quad c(n_0) \longrightarrow \cdots \quad c(m) = q_f \rightarrow \cdots \quad c(k) = q_f \longrightarrow \cdots$$

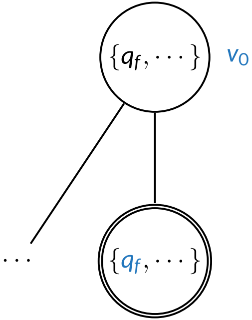
$$\cdots \quad d(n_0) \longrightarrow \cdots \quad d(m) \longrightarrow \cdots \quad d(k) \longrightarrow \cdots$$

Dimostrazione di correttezza - X

Dato che q_f è uno stato finale (dell'automa di Büchi di partenza), esso comparirà in un figlio della radice di $d(m)$ per effetto dell'esecuzione del passo 2 dell'algoritmo.

$$\cdots\ c(n_0) \longrightarrow \cdots\ c(m) = q_f \longrightarrow \cdots\ c(k) = q_f \longrightarrow \cdots$$

$$\cdots\ d(n_0) \longrightarrow \cdots\ d(m) \longrightarrow \cdots\ d(k) \longrightarrow \cdots$$

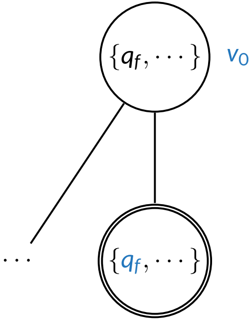


Dimostrazione di correttezza - X

Tuttavia il passo 3 potrebbe causare l'eliminazione di q_f dal nuovo figlio marcato di v_0 (ciò accade quando q_f è già presente in un altro discendente della radice). Infine, si noti come il passo 5 dell'algoritmo non si applichi mai a v_0 dopo n_0, perché, altrimenti, v_0 diventerebbe marcato. Ne segue che, a partire da m, v_0 ha sempre figli.

$$\cdots \quad c(n_0) \longrightarrow \cdots \; c(m) = q_f \longrightarrow \cdots \; c(k) = q_f \longrightarrow \cdots$$

$$\cdots \quad d(n_0) \longrightarrow \cdots \; d(m) \longrightarrow \cdots \; d(k) \longrightarrow \cdots$$

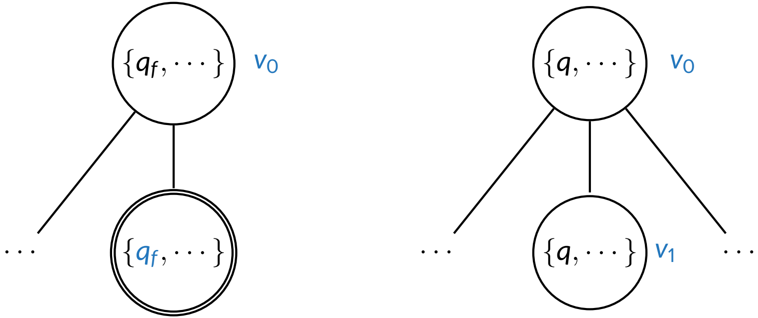


Dimostrazione di correttezza - XI

A partire da un qualche $n_1 \geq m$ in poi, ogni stato $c(n)$, con $n \geq n_1$, sarà presente in uno specifico figlio v_1 della radice di $d(n)$:

$$\cdots\ c(m) = q_f \longrightarrow \cdots\ c(n_1) = q\ \cdots$$

$$\cdots\ d(m) \longrightarrow \cdots\ d(n_1)\ \cdots$$



Dimostrazione di correttezza - XII

L'esistenza di un tale nodo v_1 può essere dimostrata nel seguente modo:

1. Se lo stato $c(n)$ è presente nell'etichetta di un nodo v, lo stato $c(n+1)$ è anch'esso presente nell'etichetta di v al passo successivo, a meno che non sia presente nell'etichetta di uno dei nodi a sinistra di v (in tal caso, esso viene eliminato dall'etichetta di v durante l'applicazione del terzo passo dell'algoritmo: eliminare gli stati presenti nelle etichette dei nodi a sinistra del nodo considerato).
2. Tali spostamenti a sinistra possono essere effettuati solamente un numero finito di volte.
3. Deve pertanto esistere un (eventualmente altro) figlio della radice che non viene eliminato: v_1 è tale figlio.

Dimostrazione di correttezza - XIII

- Ora, se v_1 viene marcato infinite volte, allora il percorso è di successo in $\mathcal{D}$
- Altrimenti, ripetiamo lo stesso procedimento sostituendo v_0 con v_1 (questo si può fare perché v_1 ha tutte le caratteristiche di v_0). Dato che gli alberi hanno altezza finita, riusciamo sempre a trovare un nodo marcato infinite volte.