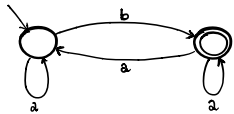


Ieri ci siamo lasciati chiedendoci se il seguente linguaggio è regolare.
 $\{w \mid w \text{ è una stringa sull'alfabeto } \{a, b\}, \text{ e } b \text{ occorre un numero dispari di volte}\}$

La risposta è sì:

- Espressione regolare: $r = (ba^*b|a)^*ba^*$

- DFA:



Il precedente DFA è minimo? Come si fa a identificare il DFA minimo?

Espressione regolare $\xrightarrow{\text{Thompson}}$ NFA $\xrightarrow{\text{Subset}}$ DFA $\xrightarrow{?}$ min DFA

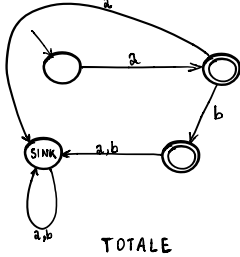
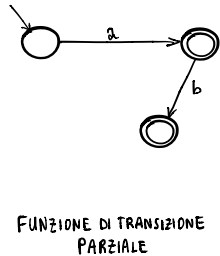
Minimizzazione DFA (light)

Ipotesi fondamentale: il DFA ha funzione di transizione totale. Il perché di questa ipotesi di partenza è dovuto al fatto che l'algoritmo lavora sulla funzione di transizione inversa. Se non è totale, non c'è l'inversa.

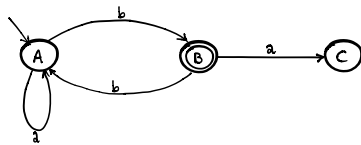
Se il DFA ha funzione di transizione parziale, possiamo trasformarlo in modo tale che la funzione di transizione diventi totale.

1. Aggiungo un nuovo stato, il pozzo (sink).
2. Per ogni stato s di D , se non c'è una transizione uscente per un certo simbolo $a \in A$ (alfabeto), allora aggiungo la transizione etichettata a dallo stato s al pozzo.
3. Aggiungo un self-loop al pozzo, per ogni $a \in A$.

ESEMPIO
 $\{a|a.b\}$

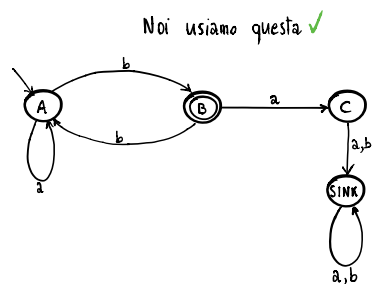
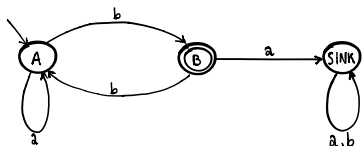


Algoritmo di partition-refinement: vediamo in azione e intuire come funziona prima di formalizzarlo.



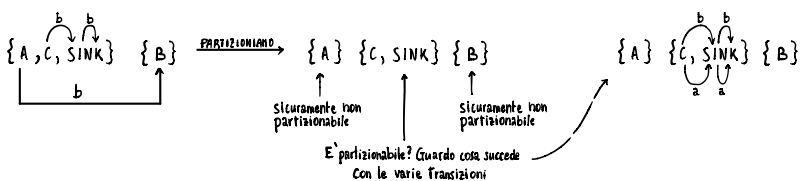
La funzione di transizione è totale? NO!

Due trasformazioni equivalenti:



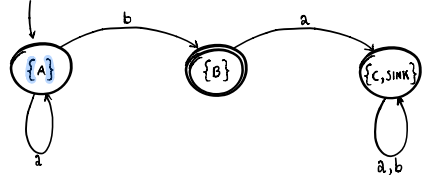
Partiamo da un partizionamento grossolano: $\{A, C, \text{SINK}\} \{B\}$

insieme degli stati non finali

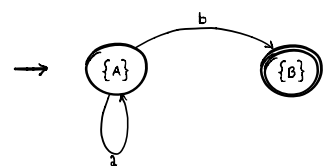


Gruppo/partizione: non è incrementabile, solo spezzabile. Sono finali i gruppi che rappresentano stati finali.

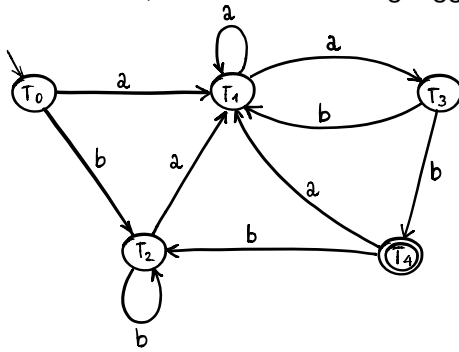
I nodi ora sono i gruppi



Non è ancora minimo: dopo che ho fatto partition refinement, devo eliminare gli stati senza transizione (dead states) e i pozzi.



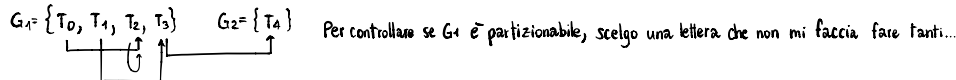
Riprendiamo il DFA di ieri, che riconosce il linguaggio denotato dall'espressione regolare $(a|b)^*abb$:



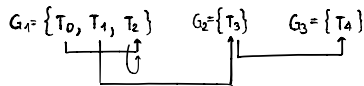
Minimizziando. Per cominciare, ci domandiamo se la funzione di transizione è totale: $ST \rightarrow OK$.

Procediamo con il raffinamento delle partizioni: $G_1 = \{T_0, T_1, T_2, T_3\}$ $G_2 = \{T_4\}$

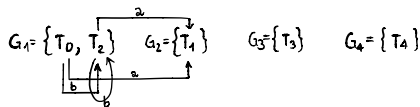
Per poter dividere un gruppo, devono esistere dei motivi, delle *ragioni*; trovo delle ragioni per cui non ritengo equivalenti gli elementi del gruppo. Ad esempio, un discriminante per il gruppo G_1 può essere il seguente: da alcuni elementi di G_1 esce una freccia (trasformazione) verso un elemento del gruppo G_2 , da altri esce una freccia verso un elemento di G_1 stesso.



Partizioniamo i due gruppi.



T_0, T_1 e T_2 non sono ancora equivalenti, perché sono differenti nel comportamento delle parole che finiscono sulla b . Partiziono ulteriormente G_1 :



• Con le 'a' vado sempre in G_2 • Con le 'b' vado sempre in G_1

