Lezione 3 – modelli di macchine di Turing

Lezione del 14/03/2023

Tanti modelli di macchine di Turing

- Siamo al paragrafo 2.4 della dispensa 2 (pag. 6). In questo paragrafo vengono introdotti diversi modelli di macchine di Turing
  - Macchine con ***tanti nastri*** con ***testine indipendenti*** : quando viene eseguita una quintupla, la testina su un nastro si può muovere come gli pare, indipendentemente da come si muovono le testine sugli altri nastri)
  - Macchine con ***tanti nastri*** con ***testine solidali*** : quando viene eseguita una quintupla, se la testina su un nastro si muove in una certa direzione, anche le testine sugli altri nastri si muovono nella stessa direzione
  - Macchine con **un solo nastro di lettura/scrittura**
  - Macchine che usano un ***alfabeto con tanti simboli***
  - Macchine che utilizzano un ***alfabeto binario***, ossia, con due soli simboli (0 e 1)
- e si dimostra che "***tutto quello che riusciamo a fare con una macchina di uno qualsiasi di questi modelli, riusciamo a farlo anche con una macchina di uno qualsiasi degli altri modelli***"

Testine indipendenti = Testine solidali

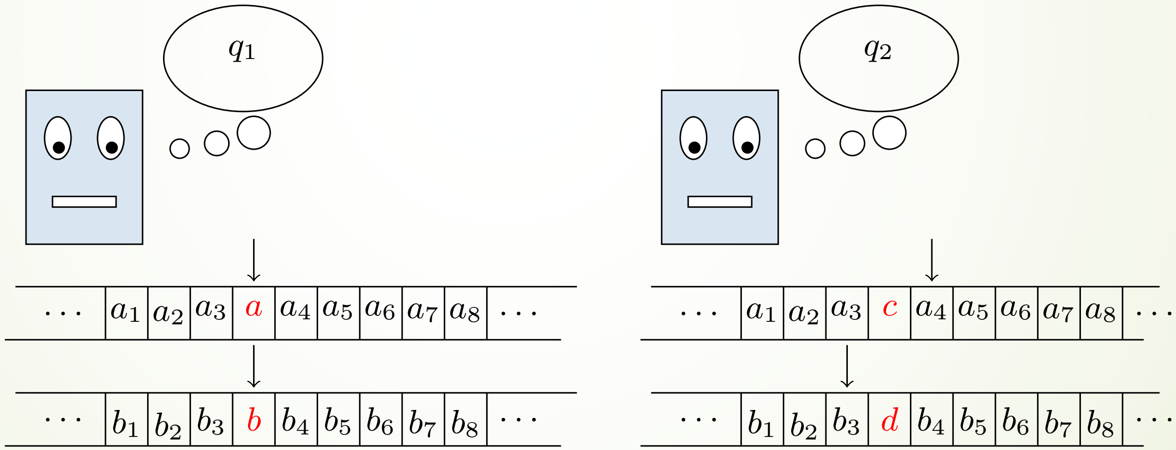
- Naturalmente, poiché una macchina a testine solidali è una particolare macchina a testine indipendenti nella quale, ogni volta che viene eseguita una quintupla, tutte le testine si muovono allo stesso modo, allora ***tutto ciò che facciamo con il modello a testine indipendenti riusciamo a farlo anche con il modello a testine solidali***
- Mostriamo ora l'inverso, ossia, che ***tutto ciò che facciamo con il modello a testine indipendenti riusciamo a farlo anche con il modello a testine solidali***
  - lo facciamo in un caso particolare: quando la macchina a testine indipendenti ha **2 nastri**
  - Ma si generalizza a quanti nastri ci pare
- Sia T una macchina a 2 nastri con testine indipendenti: una sua quintupla è **$\langle q_1, (a,b), (c,d), q_2, (m1,m2) \rangle$** dove m1 è il movimento della testina sul primo nastro e m2 è il movimento della testina sul secondo nastro
- Vediamo come trasformare quella quintupla in ***un insieme di quintuple*** di una macchina T' a tre nastri a testine solidali che "si comporta come" la quintupla di T

Testine indipendenti = Testine solidali

- Vediamo come trasformare una quintupla **⟨ q_1 , (a,b), (c,d), q_2 , (m1,m2) ⟩** di T
  - dove m1 è il movimento della testina sul primo nastro e m2 è il movimento della testina sul secondo nastro
- in un insieme di quintuple di una macchina T' a due nastri a testine solidali che "si comporta come" la quintupla di T
  - per farlo, descriviamo prima le quintuple di T', e solo in seguito il suo alfabeto e il suo insieme degli stati
- Intanto, ricordiamo che una quintupla di T' ha la seguente struttura: **⟨ q_x , (u,v), (w,z), q_y , m ⟩** - è presente un solo simbolo per descrivere il movimento delle testine
- Torniamo alla quintupla **⟨ q_1 , (a,b), (c,d), q_2 , (m1,m2) ⟩** di T
  - se **m1=m2** (ad esempio, entrambe le testine si muovono a destra), allora è facile! La quintupla **⟨ q_1 , (a,b), (c,d), q_2 , m1 ⟩ = ⟨ q_1 , (a,b), (c,d), q_2 , m2 ⟩** è inserita fra le quintuple di T' e fa le stesse cose che in T sono fatte dalla quintupla **⟨ q_1 , (a,b), (c,d), q_2 , (m1,m2) ⟩**
  - Se invece m1 e m2 sono diversi, ad esempio m1= destra e m2 = sinistra, allora le cose sono più complicate…

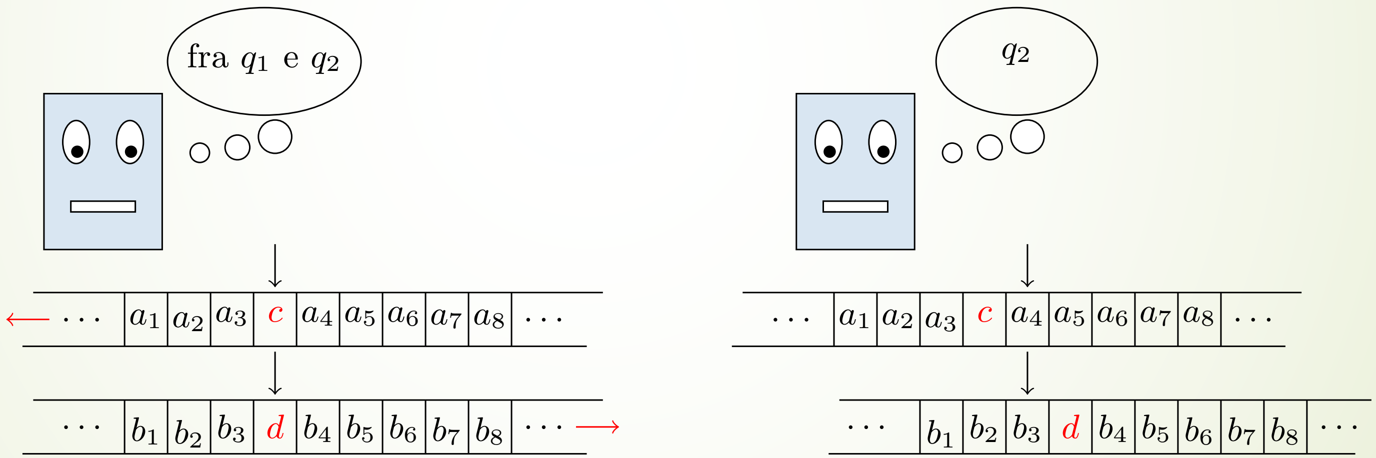
Il caso m1=destra, m2=sinistra

- Cominciamo con il vedere cosa accade in T quando esegue
  ⟨ q_1 , (a,b), (c,d), q_2 , (destra,sinistra) ⟩



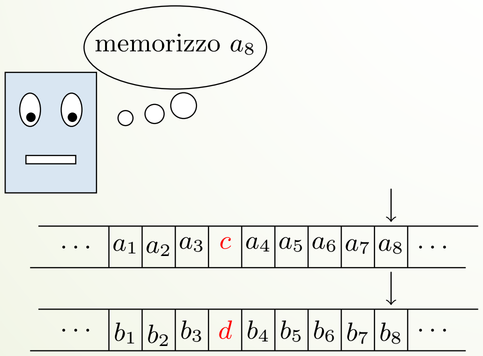
Il caso m1=destra, m2=sinistra

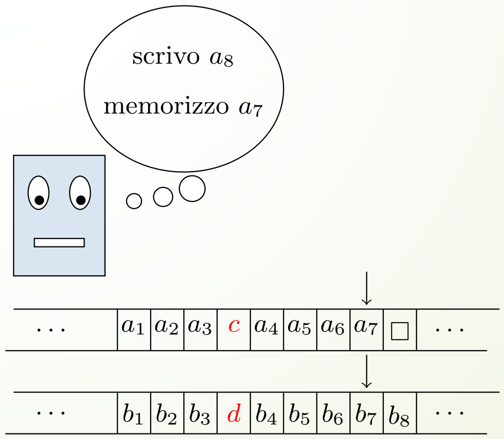
- Come facciamo ad ottenere lo stesso comportamento in T'?
- Pensate come sarebbe facile se, dopo aver scritto c sul primo nastro e d sul secondo nastro, potessimo tirare il primo nastro a sinistra e il secondo nastro a destra (tenendo ferme le testine), come indicato dalle frecce rosse nel disegno a sinistra, per ottenere lo stato globale nel disegno a destra:



Il caso m1=destra, m2=sinistra

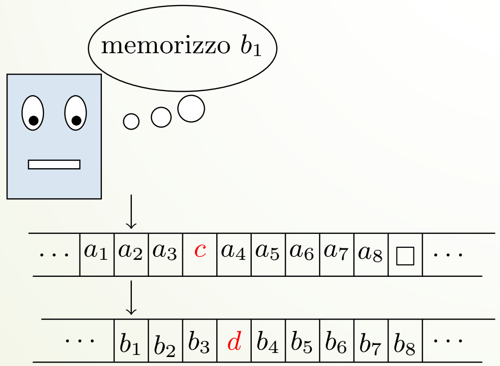
- Ma i nastri non si possono tirare da una parte o dall'altra…
  - Allora, dobbiamo armarci di santa pazienza e
    - **ricordandoci la coppia di celle dalla quale partiamo**
  - spostarci sul carattere più a destra del primo nastro,
  - leggere quel carrattere e **ricordandolo**, cancellarlo e copiarlo sulla cella a sinistra **ricordando** il carattere che vi era scritto in precedenza, e ripetere questo "shift" dei caratteri sul primo nastro, fino ad aver raggiunto il carattere più a destra
  - "memorizzo a_8" = entro in uno stato che dipende da a_8, del tipo $q(a_8)$

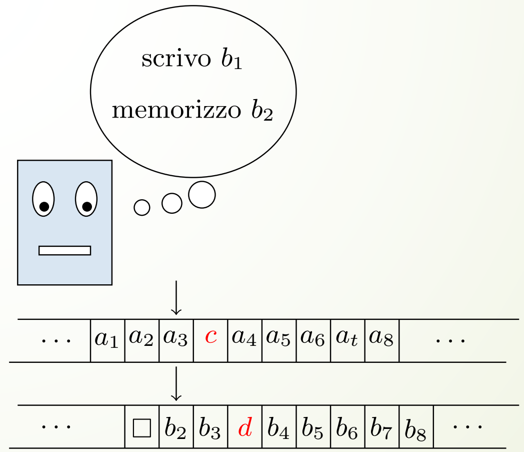




Il caso m1=destra, m2=sinistra

- Ma i nastri non si possono tirare da una parte o dall'altra…
  - … Terminato lo "shift" sul primo nastro, **sempre ricordandoci da quali celle eravamo partiti**, dobbiamo
  - spostarci sul carattere più a sinistra del secondo nastro
  - leggere quel carrattere e ricordandolo, cancellarlo e copiarlo sulla cella a destra ricordando il carattere che vi era scritto in precedenza, e ripetere questo "shift" dei caratteri sul secondo nastro, fino ad aver raggiunto il carattere più a destra



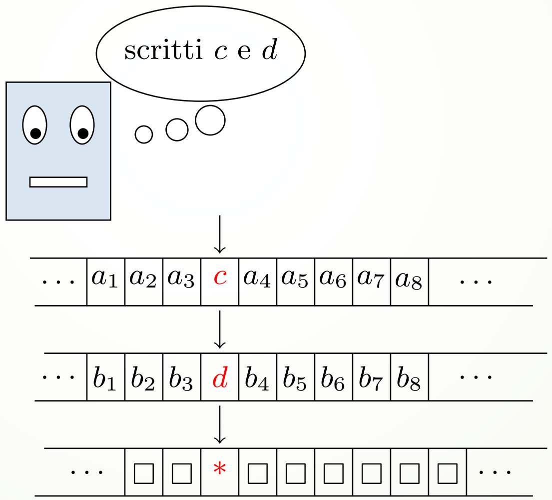


Il caso m1=destra, m2=sinistra

- Ed ora, non ci resta che posizionarci sulla cella dalla quale eravamo partiti
- Già, ma come facciamo a ricordarci da dove eravamo partiti?!
- Abbiamo bisogno di un terzo nastro sul quale scrivere un carattere speciale, tipo '*', che faccia da "segnaposto"
- E questo lo illustriamo nelle figure alle prossime pagine:
  - Nella prima, T' ha appena sostituto 'a' con 'c' sul primo nastro e 'b' con 'd' sul secondo nastro, e si prepara ad eseguire lo shift sul primo nastro
  - Nella seconda, T' ha appena finito lo shift sul primo nastro e si prepara ad eseguire lo shift sul secondo nastro
  - Nella terza, T' ha appena finito lo shift sul secondo nastro e si prepara a posizionare le testine
  - Nella quarta, T' ha posizionato le testine nella posizione indicata da '*': le testine sui primi due nastri leggono gli stessi caratteri letti dalle testine di T al termine dell'esecuzione della quintupla $\langle q_1, (a,b), (c,d), q_2, (destra,sinistra) \rangle$
  - la simulazione della quintupla $\langle q_1, (a,b), (c,d), q_2, (destra,sinistra) \rangle$ di T è terminata!

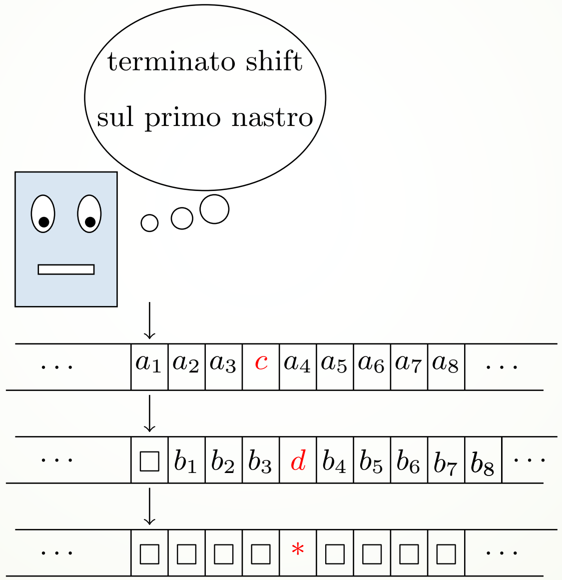
Il caso m1=destra, m2=sinistra

- In questo istante, T' ha appena sostituto 'a' con 'c' sul primo nastro e 'b' con 'd' sul secondo nastro, e si prepara ad eseguire lo shift sul primo nastro



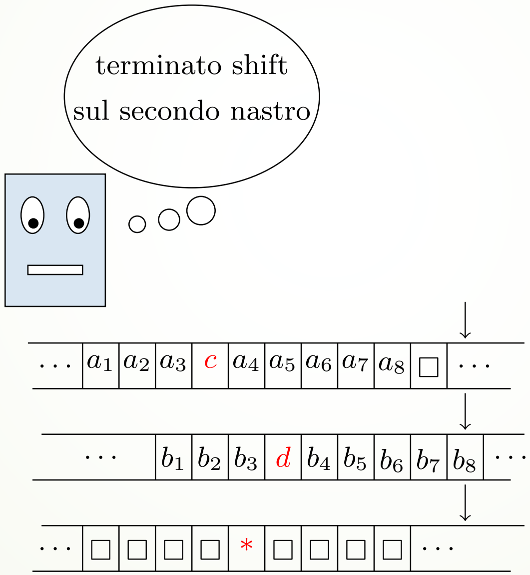
Il caso m1=destra, m2=sinistra

- In questo istante, T' ha appena finito lo shift sul primo nastro e si prepara ad eseguire lo shift sul secondo nastro



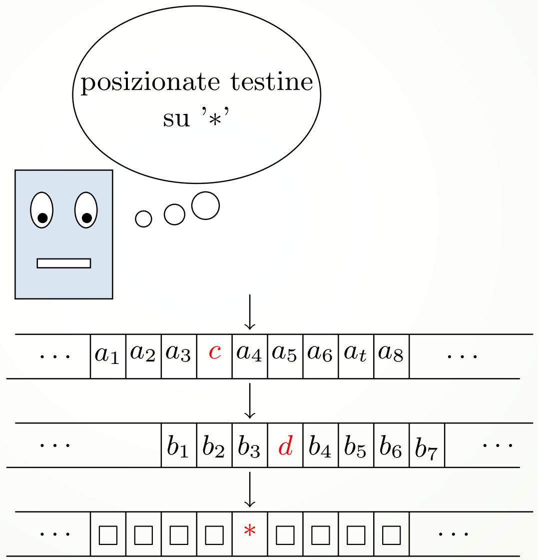
Il caso m1=destra, m2=sinistra

- In questo istante, T' ha appena finito lo shift sul secondo nastro e si prepara a posizionare le testine



Il caso m1=destra, m2=sinistra

- In questo istante, T' ha posizionato le testine nella posizione indicata da '*': le testine sui primi due nastri leggono gli stessi caratteri letti dalle testine di T al termine dell'esecuzione della quintupla 〈 q_1 , (a,b), (c,d), q_2 , (destra,sinistra) 〉



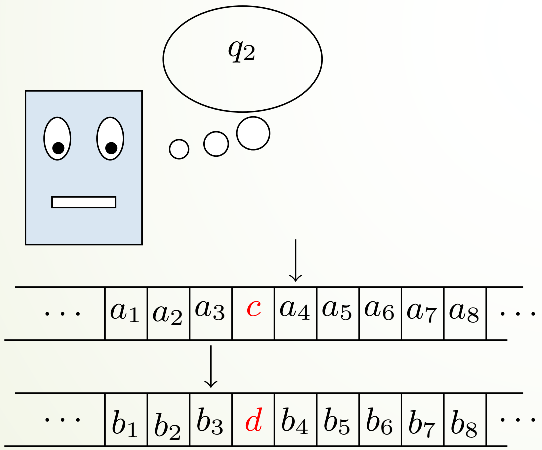
- la simulazione della quintupla **〈 q_1 , (a,b), (c,d), q_2 , (destra,sinistra) 〉** di T è terminata!

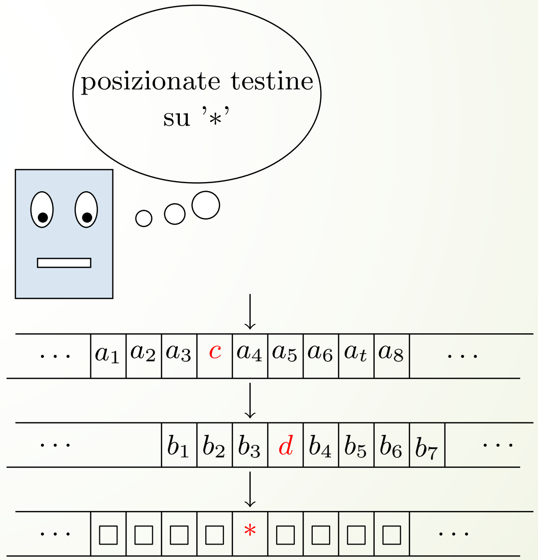
Il caso m1=destra, m2=sinistra

- Riassumendo: una computazione di T' *simula* una computazione di T – ossia, impiegando un (bel) po' di tempo in più, <u>fa passo passo le stesse cose che fa T</u>
- Più in particolare, per ogni quintupla p in T, in T' è definito <u>un insieme p' di quintuple</u> tali che: quando
  - i contenuti dei nastri di T e dei primi due nastri di T' sono uguali e
  - le testine di T e le prime due testine di T' leggono gli stessi caratteri e
  - la quintupla p può essere eseguita da T
- allora
  - le quintuple nell'insieme p' possono essere eseguite da T' e, inoltre,
- al termine dell'esecuzione di p da parte di T e dell'insieme p' da parte di T'
  - i contenuti dei primi due nastri di T' e dei nastri di T sono uguali e
  - le testine di T e le prime due testine di T' leggono gli stessi caratteri

Il caso m1=destra, m2=sinistra

- Infatti:
  - a sinistra, T dopo aver eseguito la quintupla **⟨ q_1 , (a,b), (c,d), q_2 , (destra,sinistra) ⟩**
  - a destra, T' dopo aver eseguito l'insieme di quintuple che corrispondono a ⟨ q_1 , (a,b), (c,d), q_2 , (destra,sinistra) ⟩ (subito dopo, T' entra nello stato q_2)





Il caso m1=destra, m2=sinistra

- E la codifica (o meglio, la *formalizzazione*) di questo bel procedimento, che vi ho raccontato alla bell'e meglio in questa lezione, la trovate alle pag. 6-8 della dispensa 2
- Ah, e naturalmente per le altre coppie di spostamenti (m1=fermo e m2=sinistra, m1=fermo e m2=destra, m1=sinistra e m2=fermo, m1=destra e m2= fermo, m1=sinistra e m2=destra) si procede in modo analogo
- In definitiva, abbiamo ***simulato*** il comportamento di una macchina con k nastri e testine indipendenti mediante una macchina a k+1 nastri e testine solidali

- Abbiamo, dunque, introdotto la tecnica della ***simulazione***
  - che consiste nel progettare una macchina T' con certe caratteristiche che "fa la stessa cosa" di un'altra macchina T che ha altre caratteristiche
- Su questa tecnica è basata la dimostrazione di un sacco di teoremi che vedremo in questo corso
  - fra i quali i due che seguono (e che terminano questa lezione)

Da tanti nastri a un solo nastro

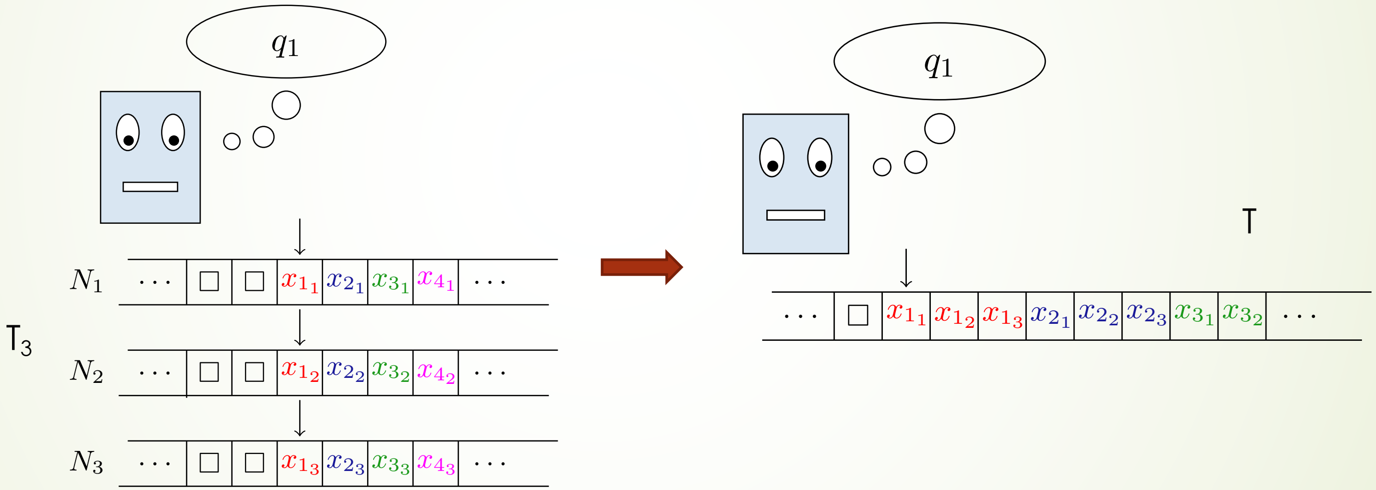
- Siamo al sotto-paragrafo 2.4.2 (pag. 9) della dispensa 2
- Vogliamo far vedere che tutto quello che possiamo fare con macchine "ricche" (che hanno tanti nastri) possiamo farlo anche con macchine "povere" (che hanno un nastro solo, meschine)
- Abbiamo una macchina T_k che ha k nastri
  - Ricordate sempre che k deve essere (indovinate un po'?) *costante*
  - ossia, non deve dipendere dall'input - sia che l'input sia di 4 caratteri, sia che l'input sia di un milione e mezzo di caratteri, T_k sempre lo stesso numero k di nastri ha!
  - E (spero) non devo chiedervi perché… Anzi, guarda un po', io ve lo chiedo: perché?
- Vogliamo far vedere che esiste una macchina T con un nastro solo che fa le stesse cose che fa T_k

Da tanti nastri a un solo nastro

- E come facciamo a far vedere che esiste una macchina T con un nastro solo che fa le stesse cose che fa T_k ?
  - Ancora con la tecnica della **simulazione**!
- Costruiamo la macchina T a partire da T_k : e facciamo l'esempio con k=3
  - grazie al teorema precedente, **possiamo supporre che T_3 sia a testine solidali**
- Per prima cosa, scriviamo l'input di T_3 sull'unico nastro di T
  - assegniamo "indirizzi" alle celle dei nastri di T_3 in modo tale che le testine solidali di T_3 siano sempre posizionate su celle che hanno lo stesso indirizzo
    - praticamente, le celle che hanno lo stesso "indirizzo" sono una "colonna di celle"
  - se $(x_{1_1}, x_{1_2}, x_{1_3})$ sono i tre caratteri scritti sulle celle "di indirizzo 1" dei 3 nastri di T_3, noi scriviamo x_{1_1} sulla cella 1, x_{1_2} sulla cella 2 e x_{1_3} sulla cella 3 di T
  - e proseguiamo così per tutto l'input di T_3 (osservate che la tripla di caratteri che occupa le celle di indirizzo h in T_3 viene scritto nelle celle 3h-2, 3h-1, 3h di T)

Da tanti nastri a un solo nastro

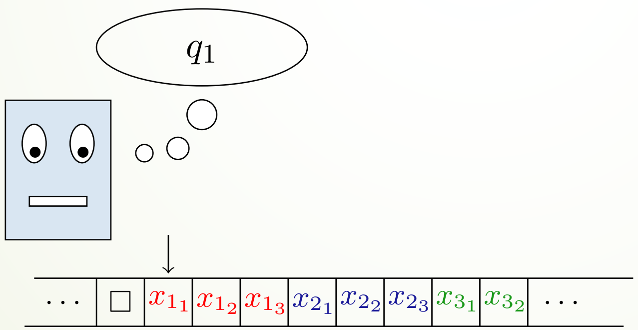
- Per prima cosa, scriviamo l'input di T_3 sull'unico nastro di T
  - se $(x_{1_1}, x_{1_2}, x_{1_3})$ sono i tre caratteri scritti sulle celle "di indirizzo 1" dei k nastri di T_3, noi scriviamo x_{1_1} sulla cella 1, x_{1_2} sulla cella 2 e x_{1_3} sulla cella 3 di T, e così via...



Le celle di T_3 con lo stesso "indirizzo" sono colorate con lo stesso colore

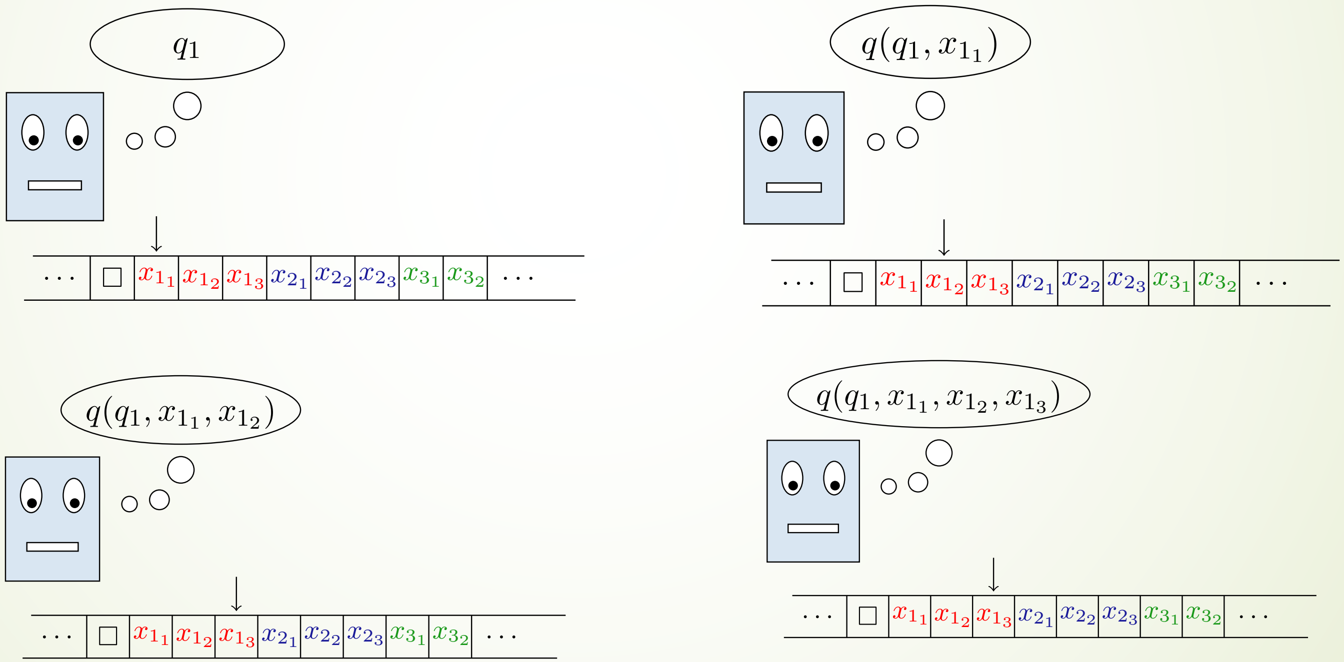
Da tanti nastri a un solo nastro

- A questo punto, sia $\langle$ q_1, $(x_{1_1}, x_{1_2}, x_{1_3})$, $(y_{1_1}, y_{1_2}, y_{1_3})$, q_2 , m $\rangle$ una quintupla di T_3
- naturalmente, T (che, meschina, ha un solo nastro) vede solo un carattere – non riesce a vedere contemporaneamente x_{1_1}, x_{1_2} e x_{1_3}
- Perciò anche se si trova nello stato q_1 e legge x_{1_1}, la quintupla $\langle$ q_1, $(x_{1_1}, x_{1_2}, x_{1_3})$, $(y_{1_1}, y_{1_2}, y_{1_3})$, q_2 , m $\rangle$ T non può eseguirla!



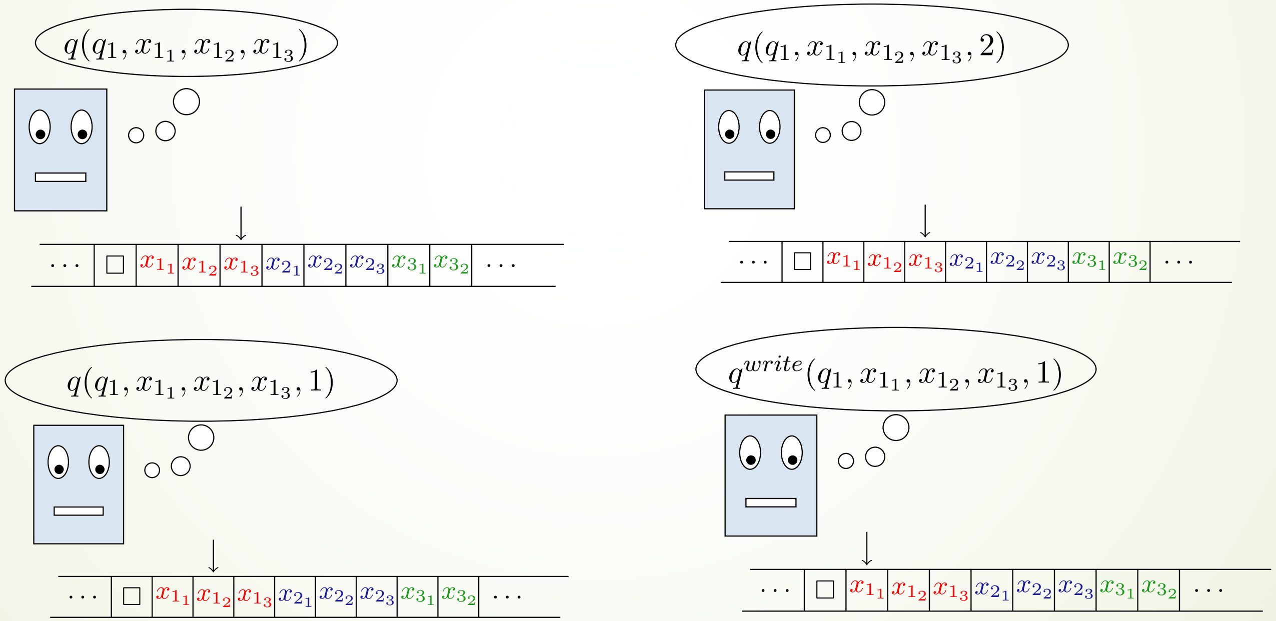
Da tanti nastri a un solo nastro

- per poter capire se può eseguire oppure no la quintupla **$\langle\, q_1, (x_{1_1}, x_{1_2}, x_{1_3}), (y_{1_1}, y_{1_2}, y_{1_3}), q_2, m \,\rangle$** , T deve leggere 3 caratteri consecutivi e memorizzarli (nello stato interno) – punto 1) pag.9



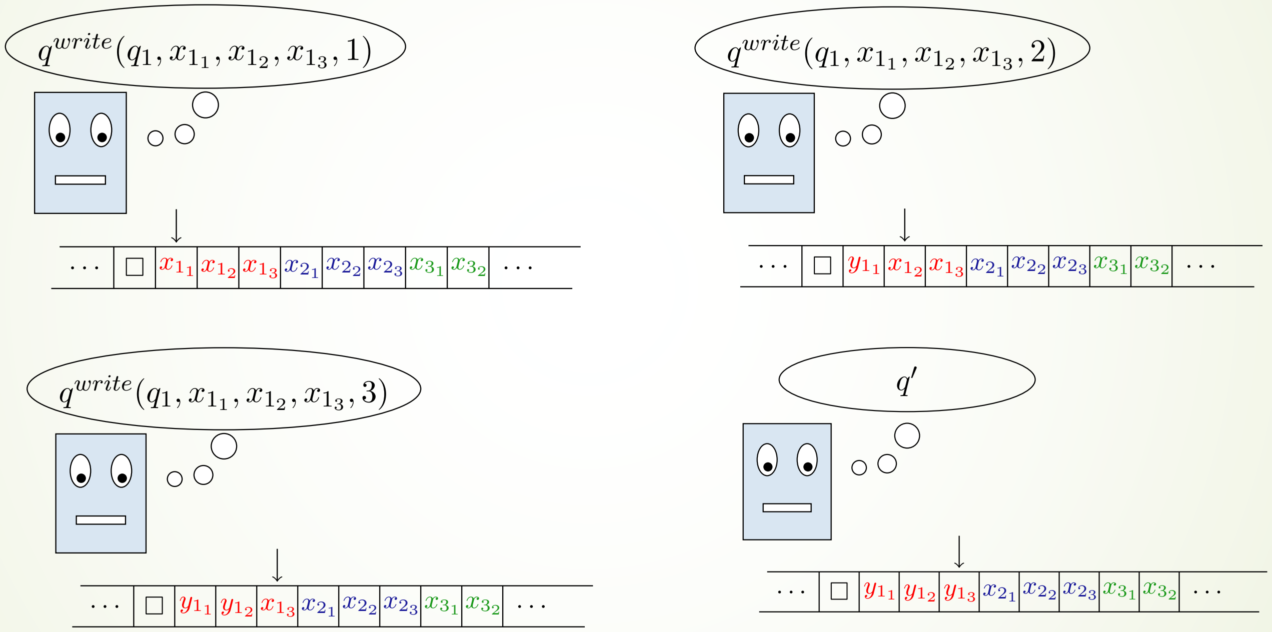
Da tanti nastri a un solo nastro

- Una volta letti i 3 caratteri x_{1_1}, x_{1_2}, x_{1_3} ed averli memorizzati nel suo stato (insieme con lo stato interno di partenza q_1), deve tornare indietro di 3 posizioni per predisporsi a scrivere y_{1_1}, y_{1_2}, y_{1_3} - punto 2) pag. 9



Da tanti nastri a un solo nastro

- Finalmente, può scrivere (in 3 passi) y_{1_1}, poi y_{1_2}, poi y_{1_3} – punto 3) pag. 9



- E, infine, T può simulare il movimento delle testine di T_3 - punto 4) pag. 9
  - ma non lo vediamo insieme

Da tanti nastri a un solo nastro

- Sia ⟨ q_1, $(x_{1_1}, x_{1_2}, x_{1_3})$, $(y_{1_1}, y_{1_2}, y_{1_3})$, q_2 , m ⟩ una quintupla di T_3
  - naturalmente, T (che, meschina, ha un solo nastro) vede solo un carattere – non riesce a vedere contemporaneamente x_{1_1}, x_{1_2} e x_{1_3}
  - Perciò anche se si trova nello stato q_1 e legge x_{1_1}, la quintupla ⟨ q_1, $(x_{1_1}, x_{1_2}, x_{1_3})$, $(y_{1_1}, y_{1_2}, y_{1_3})$, q_2 , m ⟩ non può eseguirla!
  - per poter capire se può eseguire oppure no quella quintupla, deve leggere 3 caratteri consecutivi e memorizzarli (nello stato interno) – punto 1) pag.9
  - Una volta letti i 3 caratteri x_{1_1}, x_{1_2}, x_{1_3} ed averli memorizzati nel suo stato (insieme con lo stato interno di partenza q_1), deve tornare indietro di 3 posizioni per predisporsi a scrivere y_{1_1}, y_{1_2}, y_{1_3} - punto 2) pag. 9
  - Finalmente, può scrivere (in 3 passi) y_{1_1}, poi y_{1_2}, poi y_{1_3} – punto 3) pag. 9
  - E, infine, T può simulare il movimento delle testine di T_3 - punto 4) pag. 9
- Insomma, con un po' di fatica in più, T si comporta proprio come T_3 sullo stesso input!
  - Noi abbiamo dato solo un'occhiata alla questione: **studiatela sulla dispensa!!!**

Da un alfabeto ricco a un alfabeto binario

- Siamo al paragrafo 2.5, a pag. 10 della dispensa 2
- Partiamo da una macchina T che è costruita su un alfabeto Σ con tanti, tanti caratteri – un alfabeto ricco!
- Si vuole mostrare che esiste una poverella macchina T_{01}, costruita sull'alfabeto (piccolo piccolo) {0,1}, che fa le stesse cose di T
- Visto che sappiamo che una macchina ad un nastro sa fare le stesse cose che sa fare una macchina a k nastri e viceversa, per semplificarci la vita prendiamo una macchina T con un nastro solo
  - Che, detto meglio, diventa "**senza perdita di generalità, possiamo assumere che T abbia un solo nastro**"
- Ancora una volta, utilizziamo la tecnica della simulazione: costruiamo una macchina T_{01} che, passo passo, "simula" l'esecuzione delle quintuple di T
  - e, per semplificarci la vita, dotiamo di **tanti nastri** la macchina T_{01}

Da un alfabeto ricco a un alfabeto binario

- Partiamo da una macchina T (ad un nastro) che è costruita su un alfabeto Σ con tanti, caratteri e costruiamo una macchina T_{01}, costruita sull'alfabeto (piccolo piccolo) {0,1}, che fa le stesse cose di T
- Partiamo dalla stessa codifica binaria b degli elementi di Σ mostrata nel paragrafo 2.5: per ogni elemento s di Σ, b(s) è una parola costituita da k = $\lceil \log_2 |\Sigma| \rceil$ caratteri '0' o '1'
- Ad esempio, se $\Sigma = \{ s_1, s_2, \dots, s_m \}$, possiamo codificare s_i con la rappresentazione binaria dell'intero i-1 con $\lceil \log_2 |\Sigma| \rceil$ bit (con $1 \leq i \leq m$)
  - esempio nell'esempio: se m = 10 allora k = 4 e $b(s_1)$ = 0000, $b(s_2)$ = 0001, ... , $b(s_9)$ = 1000, $b(s_{10})$ = 1001
- Poi, per ogni elemento s di Σ e per ogni h compreso fra 1 e k = $\lceil \log_2 |\Sigma| \rceil$, indichiamo con $b_h(s)$ l'h-esimo bit di b(s): ossia, b(s) = $b_1(s)\, b_2(s) \dots b_k(s)$
  - nell'esempio precedente: $b(s_6)$ = 0101 e $b_1(s_6)$ = 0
- A questo punto, costruiamo T_{01} come una macchina con k nastri e…
  - **osservazione: poiché $|\Sigma|$ è costante, allora anche k è costante!**

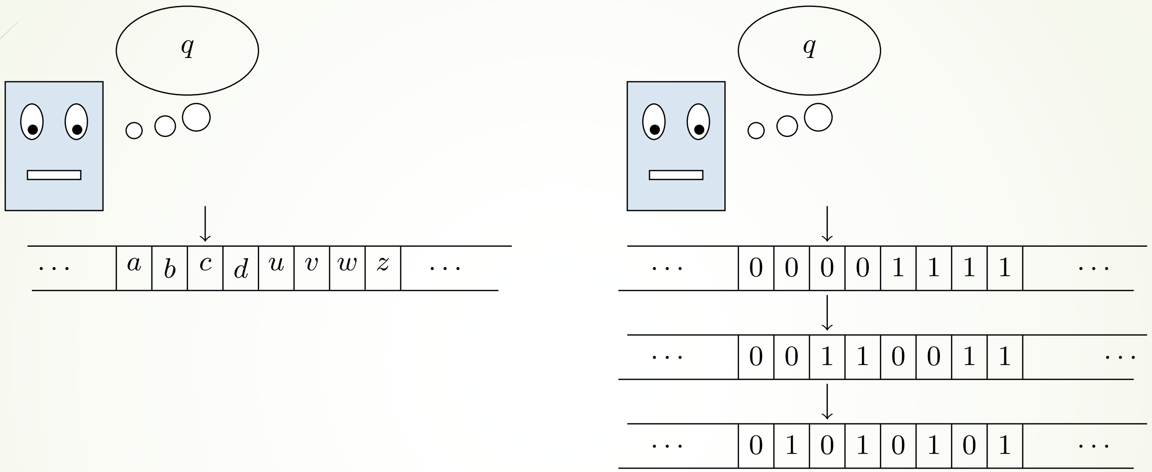
Da un alfabeto ricco a un alfabeto binario

- Cominciciamo con lo scrivere sui k nastri di T_{01} la codifica binaria dell'input scritto sull'unico nastro di T
  - Sia $x_1x_2 \dots x_h$ l'input di T (con $x_1\ x_2 \dots x_h$ elementi di Σ)
  - Nelle celle di indirizzo 1 dei nastri di T_{01} scriviamo i simboli binari della codifica di x_1: se $b(x_1)=b_1(x_1)\ b_2(x_1)\dots\ b_k(x_1)$ (ossia, $b_i(x_1)$ indica l'i-esimo bit di $b(x_1)$), allora scriviamo $b_1(x_1)$ nella cella 1 del primo nastro, $b_2(x_1)$ nella cella 1 del secondo nastro, e così via
  - Nelle celle di indirizzo 1 dei nastri di T_{01} scriviamo i simboli binari della codifica di $x_{2,}$, … e nelle celle di indirizzo k scriviamo i simboli binari della codifica di x_1.

- Nella figura nella prossima pagina trovate un esempio in cui l'alfabeto di T è {a,b,c,d,u,v,w,z}, e la codifica b (che usa 3 bit per carattere) è

b(a)=000	b(b)=001	b(c)=010	b(d)=011
b(u)=100	b(v)=101	b(w)=110	b(z)=111

Da un alfabeto ricco a un alfabeto binario



- A questo punto, una quintupla **⟨ q_1 , a, c, q_2 , m ⟩** di T diventa la quintupla **⟨ q_1 , (b_1(a), b_2(a), ... b_k(a)), (b_1(c), b_2(c), ... b_k(c)), q_2 , m ⟩** di T_{01}
- Facile facile: la macchina T con input x (parola di Σ) fa le stesse cose che fa T_{01} con input b(x)
- Facile! 😇

Da un alfabeto ricco a un alfabeto binario

- Riassumendo, a partire da T abbiamo costruito una macchina di Turing T_{01} , a tanti nastri e definita su alfabeto binario, che simula T
- A partire da T_{01} , utilizzando gli altri risultati presentati in questa lezione, possiamo poi costruire una macchina T'_{01} ad un solo nastro e definita su alfabeto binario, che simula T_{01}
  - e T'_{01} è la macchina descritta nel paragrafo 2.5
  - che è parecchio più complicata!
  - Ma a noi, in questa lezione, è sufficiente descrivere la macchina T_{01} , proprio perché abbiamo già imparato a simulare macchine a tanti nastri mediante macchine a un solo nastro

Uhmmm…

- Sì, va bene, siamo stati bravi: abbiamo visto come si fa a costruire una macchina che "fa le stesse cose" di un'altra macchina
- Ma che vuol dire "fare le stesse cose"?
- Beh, intanto, formalmente, "una macchina fa le stesse cose di un'altra macchina" si dice "***l'esito della computazione di una macchina su un input coincide con l'esito della computazione dell'altra macchina sullo stesso input (eventualmente codificato)***"
- E cosa sia, formalmente, l'esito di una computazione lo abbiamo visto nel corso della scorsa lezione