Macchine di Turing e TM Universale

Indice

Abbiamo introdotto le macchine di Turing come un modello di calcolo che permette di definire formalmente il concetto, informale, di *passo elementare* alla base del concetto intuitivo di algoritmo.
Per risolvere un problema π è necessario progettare una macchina di Turing T_π *ad hoc* per quel problema. Abbiamo visto, che, una volta che eseguiamo le istruzioni codificate nelle quintuple di T_π su un dato input x, otteniamo una soluzione per l'istanza x di π.

Riassumendo: per risolvere un problema è necessario progettare una macchina di Turing *ad hoc* per quel problema. Dunque una macchina di Turing non è altro che un algoritmo. Ossia, da quanto visto fino ad ora, possiamo affermare che il modello di calcolo *macchina di Turing* è un linguaggio per descrivere gli algoritmi. In questa dispensa mostreremo che possiamo decrivere non solo gli algoritmi, con questo linguaggio, ma anche gli esecutori degli algoritmi, ossia i calcolatori. Ma questo significa, in ultima analisi, anche un calcolatore non è niente altro che un algoritmo ....

Prima di procedere con questo compito ambizioso, cui arriveremo per gradi dopo avere introdotto differenti modelli di macchine di Turing ed averne mostrato l'equivalenza, abbiamo bisogno di formalizzare alcune definizioni che sono state introdotte, intuitivamente, nella Dispensa 1, nonché di introdurne di nuove.

2.1 Qualche definizione, per cominciare

Dato un alfabeto finito Σ, con il simbolo Σ^* indichiamo l'insieme delle *parole*, ossia, l'insieme delle sequenze costitiuite da un numero finito (eventualmente nullo) di elementi di Σ:

$$\Sigma^* = \{\langle x_1 x_2 \ldots x_n\rangle : n \in \mathbb{N} \wedge \forall 1 \leq i \leq n\ [\ x_i \in \Sigma\ ]\ \}.$$

Nel resto di questo paragrafo, per semplicità, indicheremo con T una macchina di Turing ad un nastro. Le definizioni che seguono sono facilmente estendibili a macchine a più nastri.

Uno *stato globale SG* di una macchina di Turing T è una "fotografia" della macchina ad un certo istante: SG contiene una descrizione della porzione non blank del nastro di T, della posizione della testina (e, dunque, del carattere da essa letto) e dello stato interno.

Esiste una *transizione* da uno stato globale SG_1 ad uno stato globale SG_2 quando esiste $\langle q_1, \sigma_1, \sigma_2, q_2, m\rangle \in P$ tale che q_1 e σ_1 sono, rispettivamente, lo stato interno e il simbolo letto dalla testina nello stato SG_1, q_2 e σ_2 sono, rispettivamente, lo stato interno e il simbolo che sostituisce σ_1 nello stato SG_2 e la testina si è spostata in accordo ad m nel passaggio da SG_1 a SG_2.

Una *computazione* (deterministica) $T(x)$ della macchina T su input $x = \langle x_1, x_2, \ldots, x_n\rangle \in \Sigma^*$ è una successione

$$SG_0(x), SG_1(x), \ldots, SG_k(x), \ldots$$

di stati globali di T tali che

- SG_0 è lo *stato globale iniziale di* $T(x)$, in cui la testina è posizionata sulla cella del nastro sulla quale è scritto x_1;
- se per qualche $k \geq 0$ lo stato interno di $SG_k(x)$ è uno stato finale, allora, per ogni $i > k$, SG_i non è definito; in questo caso, SG_k è uno *stato globale finale di* $T(x)$;
- per ogni $i \geq 0$ tale che $SG_i(x)$ non è uno stato globale finale, esiste una transizione da $SG_i(x)$ a $SG_{i+1}(x)$.

Se una computazione $T(x)$ contiene uno stato globale finale, allora la computazione *termina*, ossia, una volta raggiunto lo stato finale globale, T non esegue più alcuna istruzione. Nella Dispensa 1 abbiamo visto esempi di computazioni che terminano. Comunque, non tutte le computazioni terminano: come semplice esempio di questo comportamento, pensiamo ad una macchina ad un nastro nel cui insieme di quintuple sono definite le seguenti due

$$\langle q_1, a, a, q_2, destra\rangle \qquad \langle q_2, b, b, q_1, sinistra\rangle,$$

in cui q_1 e q_2 non sono stati finali. Allora, la computazione $T(ab)$ non termina.

2.2 Trasduttori e riconoscitori

Nella prima dispensa di questo corso abbiamo visto diversi esempi di macchine di Turing: la macchina (informalmente descritta) che *ordina* gli elementi di un insieme, la macchina che *verifica* se una parola è palindroma, la macchina che *calcola* la somma di due numeri. Osserviamo che, mentre ciò che risulta dalle operazioni eseguite dalla prima e dalla terza macchina è qualcosa scritto sul nastro (che costituisce il risultato del problema da esse risolto), il lavoro eseguito dalla seconda macchina ha come unico risultato lo stato interno in cui termina la computazione: ossia, *l'output della seconda macchina è codificato nello stato interno ed è di tipo booleano*.
Macchine del primo tipo sono dette *trasduttori* ed il loro compito è *calcolare* un valore. Tipicamente, ciò che viene calcolato è il valore di una funzione. Poiché l'output di una macchina di tipo trasduttore è sempre una sequenza di caratteri scritta sul nastro, il particolare stato finale in cui termina una computazione non fornisce alcuna informazione sul suo esito. Per questa ragione, da ora in avanti assumeremo sempre insieme degli stati finali di una macchina di tipo trasduttore sia costituito da un singolo elemento che denoteremo q_F. Inoltre, per comodità, assumeremo sempre che ogni macchina di tipo trasduttore disponga di almeno due nastri, uno dei quali, il *nastro di output*, è quello sul quale, al termine della computazione, sarà scritto il valore calcolato. Il nastro di output viene utilizzato esclusivamente per scrivervi l'output: esso è inizialmente vuoto e, una volta che vi viene scritto un carattere, tale carattere non potrà essere cancellato e farà parte della parola calcolata. Osserviamo anche che la testina che opera sul nastro di output è, in effetti, una testina di sola scrittura e che si muove sempre in una sola direzione (da sinistra a destra). Pertanto, poiché il nastro di output è un nastro preposto a svolgere una funzione specifica e poiché è presente in ogni trasduttore, parlando di una macchina di Turing di tipo trasduttore a k nastri ci riferiremo sempre ad una macchina che utilizza k *nastri di input e di lavoro* più il nastro di output (che non sarà, quindi conteggiato nel computo del numero di nastri).
Macchine del secondo tipo sono dette *riconoscitori* ed il loro compito è *decidere* se l'input appartiene ad un insieme. Nell'esempio della palindromia, la macchina verificava se la parola scritta sul nastro apparteneva all'insieme delle parole palindrome costituite da caratteri a e b. Poiché l'insieme delle risposte di una macchina di tipo riconoscitore è sempre $\{appartiene, non\_appartiene\}$, allora l'insieme degli stati finali di una macchina di tipo riconoscitore è sempre costituito da due stati; da ora in avanti assumeremo sempre che l'insieme degli stati finali di una macchina di tipo riconoscitore sia $\{q_A, q_R\}$: una computazione che termina nello *stato di accettazione* q_A indica che l'input appartiene all'insieme, una computazione che termina nello *stato di rigetto* q_R indica che l'input non appartiene all'insieme.

Una macchina R di tipo riconoscitore che utilizza k nastri può essere facilmente trasformata in una macchina T di tipo trasduttore che utilizza k nastri più il nastro di output. Per dimostrare questa affermazione, indichiamo con Σ_R, Q_R e P_R, rispettivamente, l'alfabeto, l'insieme degli stati e l'insieme delle quintuple di R. Allora, T avrà come alfabeto $\Sigma_T = \Sigma_R \cup \{q_A, q_R\}$, come insieme degli stati $Q_T = Q_R \cup \{q_F\}$ (si osservi che q_A e q_R sono in T sia stati che simboli dell'alfabeto) e come insieme delle quintuple l'insieme $P_T = P_R \cup P_A \cup P_R$, dove una quitupla di P_R viene interpretata come quintupla di P_T che non scrive sul nastro di output di T e, inoltre,

$$P_A = \{\langle q_a, (x_1, \ldots, x_k, \Box), (x_1, \ldots, x_k, q_A), q_F, ferma\rangle : \ x_1, \ldots, x_k \in \Sigma_T^k\}$$

$$P_R = \langle q_R, (x_1, \ldots, x_k, \Box), (x_1, \ldots, x_k, q_R), q_F, ferma\rangle : \ x_1, \ldots, x_k \in \Sigma_T^k\}.$$

I due insiemi di quintuple aggiuntive, P_A e P_R, fanno in modo che, una volta raggiunto lo stato q_A o q_R, qualunque cosa venga letta sui primi k nastri ($x_1, \ldots, x_k \in \Sigma_T^k$), la macchina T scriva sul nastro $k+1$ lo stato finale raggiunto dalla computazione $R(x)$.

Nel seguito di questo corso indicheremo come $o_T(x)$ *l'esito* della computazione $T(x)$ della macchina T sull'input x. Chiariamo, di seguito, il significato di $o_T(x)$, differenziandolo per trasduttori e riconoscitori.
Sia T una macchina di Turing di tipo trasduttore; la funzione $o_T : \Sigma^* \to \Sigma^*$ è definita per i soli $x \in \Sigma^*$ tali che $T(x)$ termina e, per tali x, il valore $o_T(x)$ è la parola calcolata da tale computazione (scritta sul nastro di output di T).
Sia T una macchina di Turing di tipo riconoscitore; la funzione $o_T : \Sigma^* \to \{q_A, q_R\}$ è definita per i soli $x \in \Sigma^*$ tali che $T(x)$ termina e, per tali x, il valore $o_T(x)$ è lo stato finale di terminazione della computazione.

Nel seguito di questo corso ci riferiremo principalmente a macchine di Turing di tipo riconoscitore.

2.3 Struttura di P e Macchine di Turing non deterministiche

Per semplificare la trattazione, ci limiteremo, in questo pargrafo a considerare macchine di Turing ad un solo nastro. Tutto quanto diremo può essere facilmente esteso agli altri modelli.

Sino ad ora non abbiamo posto alcun vincolo sulla struttura dell'insieme P delle quintuple che definisce una particolare macchina di Turing.
Ricordiamo che, data una macchina di Turing T, che, per semplicità supporremo di tipo riconoscitore e ad un solo nastro[1], definita sull'alfabeto Σ e sull'insieme Q di stati, un elemento di P è una quintupla $\langle q_1, s_1, s_2, q_2, m\rangle$ il cui scopo è quello di indicare alla macchina quali azioni intraprendere quando, trovandosi essa nello stato interno q_1, la sua testina legge sul nastro il simbolo s_1. Una quintupla, dunque, non è altro che una *istruzione* del linguaggio associato alle macchine di Turing e, in effetti, può essere vista come una sorta di istruzione condizionale: *se la macchina T si trova nello stato interno $q_1 \in Q$ e legge il simbolo $s_1 \in \Sigma$ sul nastro, allora deve scrivere sul nastro il simbolo s_2, assumere lo stato interno q_2 e muovere la testina in accordo a quanto specificato da m.*
Possiamo, quindi, dire che P è una corrispondenza che associa ad elementi dell'insieme $Q \times \Sigma$ (la condizione dell'istruzione) elementi dell'insieme $\Sigma \times Q \times \{destra, sinistra, ferma\}$ (le azioni da intraprendere). Ma di quale tipo di corrispondenza si tratta?

Totalità. Innanzi tutto, non abbiamo detto nulla circa la *totalià* di tale corrispondenza, ossia, se *per ogni coppia* $(q_1, s_1) \in (Q - Q_F) \times \Sigma$ (ove, sottolineiamo, q_1 non è uno stato finale) esiste una tripla $(s_2, q_2, m) \in \Sigma \times Q \times \{destra, sinistra, ferma\}$ tale che $\langle q_1, s_1, s_2, q_2, m\rangle \in P$. In generale, questa questione è connessa alla verifica delle *precondizioni*: quando progettiamo una macchina di Turing (per risolvere un certo problema) *assumiamo* che l'input venga scritto sul nastro in un certo formato, ossia, soddisfi un certo insieme di vincoli. Ad esempio, quando abbiamo progettato il trasduttore ad un nastro che calcola la somma di due numeri codificati in binario abbiamo assunto che *le cifre del primo numero fossero scritte in celle consecutive, con la cifra più significativa a sinistra, che fossero immediatamente seguite (senza $\Box$ intermedi) da + e che quest'ultimo fosse immediatamente seguito dalle cifre del secondo numero scritte in celle consecutive*. Così, la parola $1\Box + 1$ non è un input valido per la nostra macchina. Per questa ragione non abbiamo previsto alcuna quintupla i cui primi due elementi fossero $(q_0, \Box)$.
In altri termini, la corrispondenza P può non essere totale (nel senso prima introdotto): questo significa che, considerando P come un insieme di quintuple, esso può non contenere le quintuple che iniziano con coppie di simboli

(stato_attuale, simbolo_letto)

che si riferiscono a configurazioni del nastro che non rispettano le precondizioni che abbiamo previsto. In effetti, il codice di una macchina di Turing (così come il codice di ogni algoritmo o di ogni programma) dovrebbe essere sempre fornito insieme alla specifica delle precondizioni rispetto alle quali ne garantiamo il corretto funzionamento: in tal modo, da un lato guidiamo l'utente nell'utilizzo del nostro codice, dall'altro ci liberiamo dalle responsabilità inerenti gli eventuali danni derivanti da un suo utilizzo improprio.

Resta da chiarire il significato che intendiamo associare all'assenza di una quintupla che inizi con un dato stato e un dato simbolo.
A questo proposito, torniamo a considerare la macchina di tipo riconoscitore ad un nastro T, e supponiamo che essa si trovi nello stato q, che la sua testina legga il simbolo s e che nell'insieme P delle sue quintuple non esista alcuna quintupla che inizi con la coppia (q, s): in tal caso, certamente, la macchina T non riuscirà a raggiungere lo stato di accettazione. Possiamo, pertanto, aggiungere all'insieme P la quintupla

$$\langle q, s, s, q, fermo\rangle$$

senza alterare l'insieme delle parole accettate da T. Analoga è la questione nel caso di macchine di tipo trasduttore. Sia T_t una macchina di tipo trasduttore, ad un nastro, che, ad un certo passo della computazione iniziata con input $x \in \Sigma^*$, si trovi nello stato q e con la testina che legge il simbolo s, nel caso in cui nell'insieme P delle sue quintuple non esista alcuna quintupla che inizia con la coppia (q, s): in tal caso, *T_t non è in grado di completare il suo compito*, ossia, T_t non è in grado di produrre l'output corrispondente al'input x che era scritto sul suo nastro all'inizio della computazione e che la ha portata, nello stato q, a leggere s. In altri termini, possiamo affermare che la computazione

[1] Quanto stiamo per dire è immediatamente estendibile alle altre tipologie di macchina di Turing.

$T_t(x)$ non produce alcun output. Pertanto, possiamo considerare una nuova macchina T_t' il cui insieme delle quintuple P' sia l'unione dell'insieme P e dell'insieme di quintuple aggiuntive

$$\{\langle q,(s,x),(s,x),q,fermo\rangle : x \in \Sigma\}$$

il cui comportamento coincide sostanzialmente con il comportamento di T_t: $T_t'(x) = T_t(x)$ per ogni $x \in \Sigma^*$ tale che $T_t(x)$ calcola un output, e $T_t'(x)$ non termina per ogni $x \in \Sigma^*$ tale che $T_t(x)$ non calcola un output.

In conclusione, possiamo sempre assumere che *in ogni macchina di Turing la corrispondenza P che definisce il suo insieme delle quintuple sia totale. Inoltre, nel resto di queste dispense assumeremo sempre che la computazione di una macchina di Turing non termini se l'input non rispetta le specifiche rispetto alla quale la macchina è stata progettata*.
In particolare, assumeremo da ora in poi che, ogni volta che nella descrizione dell'insieme P delle quintuple di una macchina di Turing non descriveremo alcuna quintupla che inizi con la coppia $(q,\sigma) \in Q \times \Sigma$, allora P conterrà la quintupla $\langle q,\sigma,\sigma,q,fermo\rangle$.
Questa assunzione giocherà un ruolo rilevante nella dimostrazione del Teorema 5.4 nella Dispensa 5.

P è una corrispondenza o una funzione?. Abbiamo detto che l'insieme P delle quintuple è una *corrispondenza* fra $Q \times \Sigma$ e $\Sigma \times Q \times \{destra, sinistra, ferma\}$. In effetti, negli esempi che abbiamo visto tale corrispondenza si è rivelata essere una *funzione*, ossia, P non conteneva coppie di quintuple che avevano gli stessi due elementi iniziali. Questo corrisponde naturalmente alla nostra idea intuitiva di algoritmo: una sequanza di istruzioni (ordini, se preferiamo) che devono essere eseguite ogni qualvolta si verifica una certa eventualità. Quando vogliamo verificare se una data parola è palindroma, dobbiamo dire alla macchina che, se legge una a all'inizio della parola allora deve verificare che la parola termini con una a. Non prendiamo in considerazione altre eventualità. In altri termini, non lasciamo gradi di libertà alla macchina: il suo comportamento è totalmente determinato. Macchine di Turing di questo genere vengono dette *deterministiche*.

È stato definito anche un modello di macchina di Turing *non deteministica*, il cui insieme P può contenere anche un numero arbitrario di quintuple che iniziano con la stessa coppia stato-carattere. Il *grado di non determinismo* di una macchina di Turing è il massimo numero di quintuple della macchina che iniziano con la stessa coppia stato-carattere. Osserviamo che il grado di non determinismo di una macchina di Turing è al più $3|Q| \cdot |\Sigma|$.
Nel seguito, limiteremo la nostra attenzione a macchine di Turing non deterministiche di tipo riconoscitore.
Possiamo visualizzare, quindi, una computazione non deterministica come un albero sorgente (ossia, orientato dalla radice verso le foglie) i cui nodi sono gli stati globali utilizzati dalla computazione. La radice di tale albero è lo stato globale iniziale della computazione e i figli di ciascun nodo interno SG sono gli stati globali che corrispondono alla esecuzione di una istruzione (non deterministica) a partire da SG. Pertanto, se k è il grado di non deteminismo di una macchina di Turing NT, ogni nodo dell'albero della computazione non deterministica $NT(x)$ ha al più k figli.
Chiamiamo *computazione deterministica di $NT(x)$* ciascun percorso nell'albero uscente dalla radice e che termina solo quando incontra una foglia.

Resta da chiarire quale sia l'esito di una computazione di una macchina di Turing non deterministica. Sia, dunque, NT una macchina di Turing non deterministica, siano q_A e q_R, rispettivamente, gli stati di accettazione e di rigetto di NT, e sia x un input di NT: allora,

$$o_{NT}(x) = \begin{cases} q_A & \text{se } \textit{almeno una} \text{ delle computazioni determenistiche di } NT(x) \text{ termina in } q_A, \\ q_R & \text{se } \textit{tutte} \text{ le computazioni determenistiche di } NT(x) \text{ terminano in } q_R, \\ \text{non definito} & \text{altrimenti.} \end{cases}$$

Osserviamo esplicitamente che una computazione non deterministica non termina se

1) nessuna delle sue computazioni deterministiche terminano nello stato q_A e, inoltre,

2) almeno una delle sue computazioni deterministiche non termina.

Una macchina di Turing non deterministica sembra inerentemente più potente di una macchina di Turing deterministica. In realtà, è possibile mostrare che

Teorema 2.1: *Per ogni macchina di Turing non deterministica NT esiste una macchina di Turing detreministica T tale che, per ogni possibile input x di NT, l'esito della computazione $NT(x)$ coincide con l'esito della computazione di $T(x)$.*

Schema della dimostrazione: In quanto segue, presentiamo soltanto una descrizione informale della prova di questo teorema. La dimostrazione è una applicazione della tecnica della *simulazione*, ossia, costruiamo una macchina deterministica T che simula il comportamento della macchina non deterministica NT con grado di non determinismo k.
La macchina T su input x esegue, sostanzialmente, una visita dell'albero corrispondente alla computazione $NT(x)$. Tale visita non può essere in profondità, ossia, non possiamo simulare prima una intera computazione deterministica, poi un'altra, e così via fino all'ultima, perché non conosciamo la lunghezza delle varie computazioni deterministiche e, in effetti, qualcuna di loro potrebbe anche non terminare. Dunque, quello che facciamo è una particolare visita in ampiezza, basata sulla tecnica della *coda di rondine con ripetizioni*. Partiamo dallo stato globale iniziale $SG(T,x,0)$ e simuliamo *tutte* le computazioni lunghe un passo (che sono al più k); se, a questo punto, non possiamo concludere nulla sull'esito della computazione $NT(x)$, allora torniamo ad $SG(T,x,0)$ e simuliamo *tutte* le computazioni lunghe due passi (che sono al più k^2). Se, neanche a questo punto, non possiamo concludere nulla sull'esito della computazione $NT(x)$, allora torniamo ad $SG(T,x,0)$ e simuliamo *tutte* le computazioni lunghe tre passi (che sono al più k^3), e così via. □

2.4 Equivalenza fra le TM ad un nastro e le TM a k nastri

Sia data una Macchina di Turing T_k a k nastri (più l'eventuale nastro di output, se T_k è di tipo trasduttore) definita sull'alfabeto Σ e sull'insieme di stati Q, con stato iniziale q_0 e insieme di stati finali Q_F. Vogliamo simulare T_k mediante una Macchina di Turing T_1 ad un nastro (più l'eventuale nastro di output, se T_k è di tipo trasduttore), ossia, vogliamo definire una Macchina di Turing T_1 tale che, per ogni $x \in \Sigma^*$, l'esito della computazione $T_k(x)$ coincide con quello della computazione $T_1(x)$.

Eseguiremo questo compito in due fasi: innanzi tutto, simuleremo una macchina a testine indipendenti mediante una macchina a testine solidali (si veda la Dispensa 1) e, successivamente, simuleremo una macchina a k nastri s testine solidali mediante una macchina ad un nastro. Per semplicità, in quanto segue ci limiteremo a considerare macchine di tipo riconoscitore.

Quanto dimostrato in questo paragrafo ci permetterà di limitarci a considerare d'ora in avanti solo macchine di Turing ad un nastro (più l'eventuale nastro di output).

2.4.1 Macchine a testine indipendenti e macchine a testine solidali

Ricordiamo che abbiamo definito macchine di Turing a più nastri di due tipi differenti.

- In una macchina di Turing a *testine solidali*, in ogni istruzione, le celle dei nastri scandite dalle testine di lettura/scrittura hanno il medesimo indirizzo: così, assumendo che all'inizio della computazione le testine siano posizionate sulle celle di indirizzo 0 dei rispettivi nastri, all'istante 1 esse saranno *tutte* posizionate sulle celle di indirizzo $+1$ oppure -1 oppure 0 (dipendentemente dal movimento specificato dalla quintupla eseguita). In generale, se dopo un certo numero di passi le testine sono posizionate sulle celle di indirizzo h, al passo successivo (ricordiamo che un passo corrisponde all'esecuzione di una quintupla) esse saranno *tutte* posizionate sulle celle di indirizzo $h+1$ oppure $h-1$ oppure h.

  Una quintupla di una macchina di Turing a k nastri a testine solidali ha la forma

  $$\langle q_i, \overline{s}_1, \overline{s}_2, q_j, mov \rangle,$$

  dove $\overline{s}_1 = (s_{1_1}, s_{1_2}, \ldots, s_{1_k})$, $\overline{s}_2 = (s_{2_1}, s_{2_2}, \ldots, s_{2_k})$ e $mov \in \{sinistra, fermo, destra\}$. Il significato di una quintupla è il seguente: *se* la macchina è nello stato q_i, la testina 1 legge il simbolo s_{1_1} sul nastro 1, la testina 2 legge il simbolo s_{1_2} sul nastro 2, …, e la testina k legge il simbolo s_{1_k} sul nastro k, *allora* la testina 1 scrive il simbolo s_{2_1} sul nastro 1, la testina 2 scrive il simbolo s_{2_2} sul nastro 2, …, la testina k scrive il simbolo s_{2_k} sul nastro k, la macchina entra nello stato q_j e *tutte* le testine eseguono il movimento *mov*.

- In una macchina di Turing a *testine indipendenti*, in seguito all'esecuzione di una quintupla le testina si muovono indipendentemente le une dalle altre. Dunque, dopo l'esecuzione di un certo numero di passi, la posizione di

una testina non ha alcuna relazione con la posizione delle altre testine. Una quintupla di una macchina di Turing a k nastri a testine indipendenti ha quindi la forma

$$\langle q_i, \bar{s}_1, \bar{s}_2, q_j, \overline{mov} \rangle,$$

dove $\bar{s}_1 = (s_{1_1}, s_{1_2}, \ldots, s_{1_k})$, $\bar{s}_2 = (s_{2_1}, s_{2_2}, \ldots, s_{2_k})$, $\overline{mov} = (mov_1, mov_2, \ldots, mov_k)$ e $mov_h \in \{sinistra, fermo, destra\}$, per $h = 1, \ldots, k$. Il significato di una quintupla è il seguente: *se* la macchina è nello stato q_i, la testina 1 legge il simbolo s_{1_1} sul nastro 1, la testina 2 legge il simbolo s_{1_2} sul nastro 2, ..., e la testina k legge il simbolo s_{1_k} sul nastro k, *allora* la testina 1 scrive il simbolo s_{2_1} sul nastro 1, la testina 2 scrive il simbolo s_{2_2} sul nastro 2, ..., la testina k scrive il simbolo s_{2_k} sul nastro k, la macchina entra nello stato q_j e la testina 1 esegue il movimento mov_1, la testina 2 esegue il movimento mov_2, ... , e la testina k esegue il movimento mov_k.

I due modelli di macchine a k nastri appena definiti sono sostanzialmente equivalenti, ossia, possiamo simulare il comportamento di una macchina T a testine indipendenti mediante una macchina T' a testine solidali e viceversa. Poiché una macchina a testine solidali può essere considerata come una particolare macchine a testine indipendenti tale che, in ogni sua quintupla, tutte le testine eseguono sempre lo stesso movimento, in quanto segue dimostreremo che una macchina a testine indipendenti può essere simulata da una mecchina a testine solidali.

Per semplicità, cominciamo con il considerare una macchina di Turing T_2 a testine indipendenti dotata di due nastri. Mostriamo come simulare il comportamento di T_2 mediante una macchina di Turing T_3 a testine solidali e dotata di 3 nastri: i primi due nastri di T_3 sono inizialmente una copia dei due nastri di T_2, mentre il terzo nastro contiene un solo carattere non appartenente a Σ, diciamo il carattere '$*$', nella cella di indirizzo 0, e viene utilizzato per segnalare alle testine su quali celle posizionarsi. Assumiamo, senza perdita di generalità che all'inizio della computazione le testine di T_2 (così come quelle di T_3) scandiscano le celle di indirizzo 0 dei rispettivi nastri.

Senza perdita di generalità, assumiamo che T_2 non scriva mai $\Box$ su nastro né che tale simbolo sia contenuto nella stringa input: allora, i caratteri $\Box$ delimitano a destra e a sinistra la porzione di nastro occupata ad ogni passo.

La macchina T_3 simula T_2 mediante una *sequenza di shift* dei contenuti dei primi due nastri in modo tale che, ad ogni passo, la testina di T_3 sia sempre posizionata sulle celle il cui indirizzo coincide con quello della cella del terzo nastro in cui è scritto il carattere '$*$'.

Sia $\langle q_1, (s_{1_1}, s_{1_2}), (s_{2_1}, s_{2_2}), q_2, (mov_1, mov_2) \rangle$ una quintupla di T_2. Mostriamo, ora, come, dipendentemente dai valori di mov_1 e mov_2, tale quintupla viene trasformata in un insieme di quintuple di T_3.

- Se $mov_1 = mov_2 = destra$, allora alla quintupla $\langle q_1, (s_{1_1}, s_{1_2}), (s_{2_1}, s_{2_2}), q_2, (mov_1, mov_2) \rangle$ di T_2 corrisponde la seguente coppia di quintuple di T_3:

  $$\langle q_1, (s_{1_1}, s_{1_2}, *), (s_{2_1}, s_{2_2}, \Box), q_2^*, destra \rangle$$

  $$\langle q_2^*, (x, y, \Box), (x, y, *), q_2, fermo \rangle, \forall x, y \in \Sigma \cup \{\Box\},$$

  in cui q_2^* è un nuovo stato, gemello dello stato q_2, non appartenente all'insieme degli stati di T_2.

- Se $mov_1 = mov_2 = sinistra$, allora alla quintupla $\langle q_1, (s_{1_1}, s_{1_2}), (s_{2_1}, s_{2_2}), q_2, (mov_1, mov_2) \rangle$ di T_2 corrisponde la seguente coppia di quintuple di T_3:

  $$\langle q_1, (s_{1_1}, s_{1_2}, *), (s_{2_1}, s_{2_2}, \Box), q_2^*, sinistra \rangle$$

  $$\langle q_2^*, (x, y, \Box), (x, y, *), q_2, fermo \rangle, \forall x, y \in \Sigma \cup \{\Box\}.$$

  Si osservi che la seconda quintupla di questo caso coincide con la seconda quintupla del caso precedente.

- Se $mov_1 = mov_2 = fermo$, allora alla quintupla $\langle q_1, (s_{1_1}, s_{1_2}), (s_{2_1}, s_{2_2}), q_2, (mov_1, mov_2) \rangle$ di T_2 corrisponde la seguente quintupla di T_3:

  $$\langle q_1, (s_{1_1}, s_{1_2}, *), (s_{2_1}, s_{2_2}, *), q_2, fermo \rangle.$$

- Se $mov_1 \neq mov_2$, allora, dopo aver scritto i caratteri 's_{2_1}' e 's_{2_2}', ripettivamente, sul primo e sul secondo nastro, eseguiamo uno shift dei primi due nastri in modo tale che i caratteri che devono essere letti al passo successivo si trovino nelle celle aventi lo stesso indirizzo della cella del terzo nastro in cui è scritto '$*$'. Consideriamo, a titolo di esempio, il caso in cui $mov_1 = destra$ e $mov_2 = sinistra$ (gli altri casi sono simili e la loro trattazione viene lasciata come esercizio), allora trasliamo ii contenuto del primo nastro di una cella a sinistra (che, nella macchina a testine indipendenti, corrisponde ad uno spostamento a destra della testina sul primo nastro) ed il contenuto del secondo nastro di una posizione a destra. In definitiva, alla quintupla $\langle q_1, (s_{1_1}, s_{1_2}), (s_{2_1}, s_{2_2}), q_2, (destra, sinistra)\rangle$ di T_2 corrisponde il seguente insieme di quintuple di T_3:

  $\langle q_1, (s_{1_1}, s_{1_2}, *), (s_{2_1}, s_{2_2}, *), q^{DS}_{1,D}(q_2), destra\rangle$

  $\langle q^{DS}_{1,D}(q_2), (x,y,z), (x,y,z), q^{DS}_{1,D}(q_2), destra\rangle \ \forall x \in \Sigma \ \forall y \in \Sigma \cup \{\square\} \ \forall z \in \{*, \square\}$

  (sposta la testina sull'ultimo carattere non $\square$ sul primo nastro)

  $\langle q^{DS}_{1,D}(q_2), (\square,y,z), (\square,y,z), q^{DS}_{1,S}(q_2,\square), sinistra\rangle \ \forall y \in \Sigma \cup \{\square\} \ \forall z \in \{*, \square\}$

  (si prepara a spostare ciascun carattere sul primo nastro una cella a sinistra)

  $\langle q^{DS}_{1,S}(q_2,a), (x,y,z), (a,y,z), q^{DS}_{1,S}(q_2,x), sinistra\rangle \ \forall a \in \Sigma \cup \{\square\} \ \forall z \in \{*, \square\} \ \forall x,y \in \Sigma \cup \{\square\} : \ x \neq \square \vee y \neq \square$

  (esegue lo spostamento di ciascun carattere sul primo nastro una cella a sinistra)

  $\langle q^{DS}_{1,S}(q_2,a), (\square,\square,z), (a,\square,z), q^{DS}_{2,D}(q_2,\square), destra\rangle \ \forall a \in \Sigma \cup \{\square\} \forall z \in \{*, \square\}$

  (raggiunta la prima posizione a destra in cui le celle del primo e secondo nastro contengono $\square$, si prepara a spostare ciascun carattere sul secondo nastro una cella a destra)

  $\langle q^{DS}_{2,D}(q_2,b), (x,y,z), (x,b,z), q^{DS}_{2,D}(q_2,y), destra\rangle \ \forall b \in \Sigma \cup \{\square\} \ \forall x \in \Sigma \cup \{\square\} \ \forall y \in \Sigma \ \forall z \in \{*, \square\}$

  (esegue lo spostamento di ciascun carattere sul secondo nastro una cella a destra)

  $\langle q^{DS}_{2,D}(q_2,b), (x,\square,z), (x,b,z), q^{DS}_{2,S}(q_2), fermo\rangle \ \forall b \in \Sigma \cup \{\square\} \ \forall x \in \Sigma \cup \{\square\} \ \forall z \in \{*, \square\}$

  (terminato lo spostamento di ciascun carattere sul secondo nastro una cella a destra, si prepara a tornare sulla cella del terzo nastro che contiene $*$)

  $\langle q^{DS}_{2,S}(q_2), (x,y,\square), (x,b,\square), q^{DS}_{2,S}(q_2), sinistra\rangle \ \forall x,y \in \Sigma \cup \{\square\}$

  (raggiunge la cella del terzo nastro che contiene $*$)

  $\langle q^{DS}_{2,S}(q_2), (x,y,*), (x,b,), q_2, fermo\rangle \ \forall x,y \in \Sigma \cup \{\square\}$

  (entra nello stato q_2: la simulazione dell'esecuzione della quintupla di T_2 è terminata.)

  che utilizzano nuovi stati interni non appartenenti all'insieme degli stati di T_2.

La simulazione di una macchina T_k a $k > 2$ nastri a testine indipendenti mediante una macchina T_{k+1} a $k+1$ nastri a testine solidali segue lo stesso schema: il nastro $k+1$ di T_{k+1} viene utilizzato per rappresentare la posizione della testina, e, nelle quintuple in cui le testine non si muovono solidarmente, vengono eseguiti uno shift a sinistra di *tutti* i nastri le cui testine devono muoversi a destra ed uno shift a destra di *tutti* i nastri le cui testine devono muoversi a sinistra.

2.4.2 Macchine a k nastri e macchine a 1 nastro

In questo paragrafo, utilizzando ancora la tecnica della simulazione, mostriamo che la capacità computazionale di una macchina di Turing non dipende dal numero di nastri di cui è dotata. Più precisamente, mostriamo che una macchina di Turing a k nastri può essere simulata mediante una macchina di Turing T_1 ad un nastro.

Per semplicità, ci limiteremo a considerare Macchine di Turing a k a testine solidali; alla luce di quanto dimostrato nel precedente paragrafo, questo non costituisce perdita di generalità.

Sia, dunque, T_k una macchina di Turing a k nastri a testine solidali. Definiamo, ora, una macchina di Turing T_1 ad 1 nastro che utilizza lo stesso alfabeto Σ utilizzato da T_k ed il cui insieme degli stati è $Q \times \Sigma^k$. Inizialmente, l'input $x = (x_{1_1}, x_{1_2}, \ldots, x_{1_k})(x_{2_1}, x_{2_2}, \ldots, x_{2_k}) \ldots (x_{n_1}, x_{n_2}, \ldots, x_{n_k})$ di T_k è scritto sull'unico nastro di T_1, a partire dalla cella 1, come concatenazione di tutti i simboli di x, ossia, nella forma seguente: $x_{1_1} x_{1_2} \ldots x_{1_k} x_{2_1} x_{2_2} \ldots x_{2_k} \ldots x_{n_1} x_{n_2} \ldots x_{n_k}$ (ovviamente, un solo simbolo per cella).

Sia ora $\langle q_1, (s_{1_1}, s_{1_2}, \ldots, s_{1_k}), (s_{2_1}, s_{2_2}, \ldots, s_{2_k}), q_2, m \rangle$ (eventualmente, $s_{1_i} = s_{2_i}$ per qualche i) una qualsiasi quintupla di T_k. Trasformiamo ora tale quintupla in un insieme di quintuple di T_1.

1) Osserviamo, innanzi tutto, che, mentre a T_k è sufficiente una singola operazione di lettura per poter garantire la corretta esecuzione della quintupla, T_1 deve eseguire k operazioni di lettura consecutive, in quanto la quintupla può essere eseguita solo se viene letto s_{1_1} *ed* esso è seguito da s_{1_2} ... *ed* esso è seguito da s_{1_k}. Allora, per poter decidere circa la possibilità di eseguire la quintupla è necessario leggere e ricordare k simboli consecutivi:

   $\langle q_1, s_{1_1}, s_{1_1}, q(q_1, s_{1_1}), destra \rangle$

   $\langle q(q_1, s_{1_1}), s_{1_2}, s_{1_2}, q(q_1, s_{1_1}, s_{1_2}), destra \rangle$

   ...

   $\langle q(q_1, s_{1_1}, s_{1_2}, \ldots, s_{1_{k-1}}), s_{1_k}, s_{1_k}, q(q_1, s_{1_1}, s_{1_2} \ldots, s_{1_{k-1}}, s_{1_k}), sinistra \rangle$.

2) Ora, T_1 ha verificato che la quintupla può essere eseguita e, per poterlo fare, deve riportare la testina a sinistra di k celle:

   $\langle q(q_1, s_{1_1}, s_{1_2}, \ldots, s_{1_{k-1}}, s_{1_k}), s_{1_{k-1}}, s_{1_{k-1}}, q(q_1, s_{1_1}, s_{1_2} \ldots, s_{1_{k-1}}, s_{1_k}, k-2), sinistra \rangle$

   $\langle q(q_1, s_{1_1}, s_{1_2}, \ldots, s_{1_{k-1}}, s_{1_k}, i), s_{1_i}, s_{1_i}, q(q_1, s_{1_1}, s_{1_2} \ldots, s_{1_{k-1}}, s_{1_k}, i-1), sinistra \rangle \; \forall i = 2, \ldots, k-2.$

3) La testina di T_1 è ora posizionata sul carattere corrispondente al carattere scritto sul primo nastro di T_k (s_{1_1}) e può procedere all'esecuzione della quintupla sovrascrivendo i k caratteri:

   $\langle q(q_1, s_{1_1}, s_{1_2}, \ldots, s_{1_{k-1}}, s_{1_k}, 1), s_{1_1}, s_{2_1}, q^{write}(q_1, s_{1_1}, s_{1_2} \ldots, s_{1_{k-1}}, s_{1_k}, 2), sinistra \rangle$

   $\langle q^{write}(q_1, s_{1_1}, s_{1_2}, \ldots, s_{1_{k-1}}, s_{1_k}, i), s_{1_i}, s_{2_i}, q^{write}(q_1, s_{1_1}, s_{1_2} \ldots, s_{1_{k-1}}, s_{1_k}, i+1), sinistra \rangle \; \forall i = 2, \ldots, k$

   $\langle q^{write}(q_1, s_{1_1}, s_{1_2}, \ldots, s_{1_{k-1}}, s_{1_k}, k), s_{1_k}, s_{2_k}, q', m' \rangle$

   dove q' ed m' dipendono dal valore di m come specificato nel seguito.

4) T_1 ha eseguito la prima parte della quintupla, scrivendo i k nuovi caratteri sul nastro; in questo istante, la sua testina è posizionata sulla cella contenente l'ultimo simbolo scritto. Restano da eseguire il cambio di stato interno ed il movimento della testina. Queste operazioni avvengono in maniera differente, dipendentemente dal valore di m.

   - Se $m = destra$, allora $q' = q_2$ e $m' = destra$ e l'esecuzione della quintupla è terminata.

- Se $m = fermo$, allora $q' = q^{sin}(q_2, k-1)$, $m' = sinistra$ e l'esecuzione della quintupla di T_k termina con le quintuple seguenti:

  $\langle q^{sin}(q_2,i), x, x, q^{sin}(q_2,i-1), sinistra\rangle \ \forall x \in \Sigma \cup \{\square\} \ \forall i = 2,\ldots,k-1$

  $\langle q^{sin}(q_2,1), x, x, q_2, fermo\rangle \ \forall x \in \Sigma \cup \{\square\}.$

- se $m = sinistra$, allora $q' = q^{sin}(q_2, 2k-1)$, $m' = sinistra$ e l'esecuzione della quintupla di T_k termina con le quintuple seguenti:

  $\langle q^{sin}(q_2,i), x, x, q^{sin}(q_2,i-1), sinistra\rangle \ \forall x \in \Sigma \cup \{\square\} \ \forall i = 2,\ldots,2k-1$

  $\langle q^{sin}(q_2,1), x, x, q_2, fermo\rangle \ \forall x \in \Sigma \cup \{\square\}.$

Si osservi che l'insieme degli stati di T_1 ha cardinalità molto maggiore dell'insieme degli stati di T_k; inoltre, l'insieme degli stati finali di T_1 coincide con quello di T_k.

Un'ultima osservazione merita particolare attenzione. Invece di utilizzare per T_1 un insieme di stati di cardinalità largamente maggiore rispetto a quella dell'insieme degli stati di T_k mantenendo inalterato l'alfabeto Σ, possiamo utilizzare per T_1 lo stesso insieme di stati utilizzato da T_k ed incrementare la cardinalità dell'alfabeto. In particolare, definiamo l'alfabeto di T_1 come $\Sigma_1 = \Sigma^k$, ossia, un simbolo scritto in una cella del nastro di T_1 è una k-pla di simboli di Σ. In questo modo, ciascun simbolo letto (o scritto) dalla testina di T_1 corrisponde ad un unico insieme di k simboli letti (o scritti) dalle k testine di T_k e la simulazione di T_k da parte di T_1 avviene mantenendo sostanzialmente inalterato l'insieme delle quintuple (a parte ovvi cambi di notazione).

2.5 Riduciamo la cardinalità di Σ

Mostriamo, in questo paragrafo, come ogni macchina di Turing (riconoscitore o trasduttore) T definita su un alfabeto Σ tale che $|\Sigma| > 2$ possa essere simulata da una macchina di Turing T_{01} definita sull'alfabeto $\{0,1\}$. Indichiamo con Q l'insieme degli stati di T e con P l'insieme delle sue quintuple. La simulazione è molto simile a quella descritta nel precedente paragrafo per trasformare una macchina a k nastri in una macchina ad un nastro. In quanto segue, descriveremo la simulzione per sole macchine di tipo riconoscitore ad un nastro.
Sia $\Sigma = \{\sigma_0, \sigma_2, \ldots, \sigma_{n-1}\}$; iniziamo con il rappresentare ciascun simbolo di Σ mediante una codifica binaria b che utilizza $k = \lceil \log_2 n \rceil$ cifre: per ogni $\sigma \in \Sigma$, indichiamo con

$$b(\sigma) = (b_1(\sigma), b_2(\sigma), \ldots, b_k(\sigma))$$

la sua codifica binaria. Inoltre, indichiamo con $b(\square)$ una sequenza di k simboli $\square$: osserviamo, infatti, che, ogni volta che T scrive (o legge) un singolo $\square$, T_{01} dovrà scrivere (o leggere) una sequenza di k $\square$ al fine di sovrascrivere tutti i simboli della codifica binaria dell'elemento di $\Sigma \cup \{\square\}$ sovrascritto da T.
Poi, utilizzando un insieme di stati Q_{01} molto più ampio di Q (e che si evincerà dall'esame delle quintuple di T_{01}), costruiamo l'insieme delle quintuple di T_{01} a partire da P: in particolare, ogni quintupla $p = \langle q, \sigma, \sigma', q', m\rangle \in P$ viene trasformata in un insieme di quintuple $P_{01}(p) = P_1(p) \cup P_2(p)$ tali che le quintuple in $P_1(p)$ verificano l'eseguibilità di p e sovrascrivono i bit di $b(\sigma)$ con quelli di $b(\sigma')$, e le quintuple in $P_2(p)$ simulano il movimento della testina corrispondente a m e portano la macchina nello stato q'. Pertanto, ecco l'insieme $P_1(p)$:

$$\begin{array}{rcl}
\langle q, b_1(\sigma), b_1(\sigma), q(b_1(\sigma)), d\rangle & & \langle q(b_1(\sigma)), b_2(\sigma), b_2(\sigma), q(b_1(\sigma), b_2(\sigma)), d\rangle \\
\ldots & & \langle q(b_1(\sigma)), \ldots, b_{k-1}(\sigma)), b_k(\sigma), b_k(\sigma'), q(\sigma, k-1), s\rangle \\
\langle q(\sigma,k-1), b_{k-1}(\sigma), b_{k-1}(\sigma'), q(\sigma,k-2,), s\rangle & \ldots & \langle q(\sigma,k-i), b_{k-i}(\sigma), b_{k-i}(\sigma'), q(\sigma,k-i-1), s\rangle \\
\ldots & & \langle q(\sigma,1), b_1(\sigma), b_1(\sigma'), q^{mv}(\sigma,k), f\rangle.
\end{array}$$

Le quintuple in $P_2(p)$ dipendono dal valore di $m \in \{s,d,f\}$: se $m = f$, allora $P_2(p)$ consiste della sola quintupla

$$\langle q^{mv}(\sigma,k), b_1(\sigma'), b_1(\sigma'), q', f\rangle.$$

Invece, se $m = s$, allora $P_2(p)$ consiste delle k quintuple

$$\langle q^{mv}(\sigma,k), b_1(\sigma'), b_1(\sigma'), q^{mv}(\sigma,k-1), s\rangle \qquad \langle q^{mv}(\sigma,k-1), a, a, q^{mv}(\sigma,k-2), s\rangle \ \forall a \in \{0,1,\square\}$$
$$\dots$$
$$\langle q^{mv}(\sigma,2), a, a, q^{mv}(\sigma,1), s\rangle \ \forall a \in \{0,1,\square\} \qquad \langle q^{mv}(\sigma,1), a, a, q', s\rangle \ \forall a \in \{0,1,\square\}.$$

Le quintuple in $P_2(p)$ relative al caso $m = d$ sono simili e la loro descrizione è, pertanto, omessa.

È immediato verificare che, per ogni $x \in \Sigma^*$, l'esito della computazione $T(x)$ coincide con l'esito della computazione $T_{01}(b(x))$.

Quanto dimostrato in questo paragrafo ci permette di limitarci a considerare, d'ora in avanti, solo macchine di Turing definite sull'alfabeto $\{0,1\}$.

2.6 La macchina di Turing Universale

La macchina di Turing Universale è una macchina di Turing particolare U che riceve in input la descrizione di un'altra macchina di Turing T e un possibile input x di T ed esegue la computazione $U(T,x)$ il cui esito coincide con quello della computazione $T(x)$. In altri termini, la macchina di Turing Universale è capace di *simulare la computatazione che qualunque macchina di Turing potrebbe eseguire su qualunque parola* una volta che le descrizioni della macchina e della parola vengono scritti sul suo nastro di input. Dunque, la macchina di Turing Universale è il *progetto logico di un calcolatore*.

In quanto segue, per semplicità, ci limitiamo a descrivere la macchina di Turing Universale di tipo riconoscitore, ossia, una macchina di Turing U che, presi in input la descrizione di una macchina di Turing di tipo riconoscitore T (ad un nastro) ed un input $x \in \{0,1\}^*$ di T, esegue una computazione con esito uguale a quello di $T(x)$.

La macchina di Turing universale U che andiamo a definire è una macchina di Turing che utilizza 4 nastri a testine indipendenti (come sappiamo dal Paragrafo 2.4, è poi possibile trasformare U in una macchina ad un solo nastro):

- N_1, il nastro su cui, all'inizio della computazione, è memorizzata la descrizione di T;
- N_2, il nastro di lavoro di U su cui, all'inizio della computazione, è memorizzato l'input x della macchina di Turing T la cui computazione $T(X)$ deve essere simulata da U;
- N_3, il nastro su cui, ad ogni istante della computazione che simula $T(x)$, sarà memorizzato lo stato attuale della macchina T;
- N_4, il nastro su cui verrà scritto lo stato di accettazione della macchina T.

Presentiamo inizialmente una descrizione ad alto livello di U, che utilizza gli stessi simboli utilizzati da T.
Per evitare confusione fra i simboli utilizzati per la macchina T e per la macchina universale U che ci accingiamo a definire, indichiamo l'insieme degli stati della generica macchina di Turing T con $Q_T = \{\omega_0, \dots, \omega_m\}$, ove ω_0, ω_1 e ω_2 sono, rispettivamente, lo stato iniziale, lo stato di accettazione e lo stato di rigetto di T. Indichiamo, poi, con $P = \{p_1, \dots, p_h\}$ l'insieme delle quintuple di T e con $p_i = \langle \omega_{i_1}, b_{i_1}, b_{i_2}, \omega_{i_2}, m_i\rangle$ la sua i-esima quintupla, con $\omega_{i_1}, \omega_{i_2} \in Q_T$, $b_{i_1}, b_{I_2} \in \{0,1,\square\}$ e $m_i \in \{s,f,d\}$. Assumiamo che T sia descritta dalla parola $\rho_T \in [Q_T \cup \{0,1,\oplus,\otimes,-\}]^*$ seguente:

$$\rho_T = \omega_0 - \omega_1 \otimes \omega_{1_1} - b_{1_1} - b_{1_2} - \omega_{1_2} - m_1 \oplus \omega_{2_1} - b_{2_1} - b_{2_2} - \omega_{2_2} - m_2 \oplus \dots \oplus \omega_{h_1} - b_{h_1} - b_{h_2} - \omega_{h_2} - m_h \oplus$$

Come già anticipato, l'input della macchina U è costituito da una stringa $\rho_T \in [Q_T \cup \{0,1,\oplus,\otimes,-\}]^*$, descrizione di una macchina di Turing di tipo riconoscitore ad un nastro, scritta sul nastro N_1, e da una parola $x \in \{0,1\}^*$ scritta sul nastro N_2 (ricordiamo anche che tali parole sono precedute e seguite da $\square$). Quando la computazione $U(\rho_T, x)$ ha inizio, le testine di N_1 e N_2 sono posizionate sui simboli diversi da $\square$ più a sinistra scritti sui due nastri. La macchina U esegue sostanzialmente l'algoritmo di seguito descritto, in cui si utilizza la convenzione per cui rimangono ferme le testine delle quali non viene specificato il movimento (in particolare, rimangono sempre ferme la testina sui nastri N_3 e N_4).

1) Nello stato q_0, vengono copiati ω_0 sul nastro N_3 e ω_1 sul nastro N_4, la testina di N_1 viene spostata sul simbolo a destra del primo carattere '$\otimes$' che incontra e la macchina entra nello stato q_1:

$$\begin{array}{ll}
\langle q_0,(x,a,\square,\square),(x,a,x,\square),q_0,(d,f,f,f)\rangle & \forall x \in Q_T \ \wedge \ \forall a \in \{0,1,\square\} \\
\langle q_0,(-,a,x,\square),(-,a,x,\square),q_0,(d,f,f,f)\rangle & \forall a \in \{0,1,\square\} \ \wedge \ \forall x \in Q_T, \\
\langle q_0,(y,a,x,\square),(y,a,x,y),q_0,(d,f,f,f)\rangle & \forall a \in \{0,1,\square\} \ \wedge \ \forall x,y \in Q_T, \\
\langle q_0,(\otimes,a,x,y),(\otimes,a,x,y),q_1,(d,f,f,f)\rangle & \forall a \in \{0,1,\square\} \ \wedge \ \forall x,y \in Q_T,
\end{array}$$

2) Nello stato q_1 ha inizio la ricerca di una quintupla su N_1 che abbia come primo simbolo lo stesso simbolo letto dalla testina di N_3 e come secondo simbolo lo stesso simbolo letto dalla testina di N_2:

   (a) se nello stato q_1 legge lo stesso simbolo sui nastri N_1 ed N_3, sposta la testina di N_1 a destra di due posizioni ed entra nello stato $q_{statoCorretto}$:

$$\begin{array}{ll}
\langle q_1,(x,a,x,y),(x,a,x,y),q_1,(d,f,f,f)\rangle & \forall x,y \in Q_T \ \wedge \ \forall a \in \{0,1,\square\} \\
\langle q_1,(-,a,x,y),(-,a,x,y),q_{statoCorretto},(d,f,f,f)\rangle & \forall a \in \{0,1,\square\} \ \wedge \ \forall x,y \in Q_T.
\end{array}$$

   Ora la testina di N_1 è posizionata sul secondo elemento (ossia, il carattere letto) della quintupla che si sta esaminando.

      i. Se nello stato $q_{statoCorretto}$ legge lo stesso simbolo sui nastri N_1 e N_2, allora ha trovato la quintupla da eseguire; pertanto, sposta la testina di N_1 a destra di due posizioni (per superare il carattere separatore '-') ed entra nello stato q_{scrivi}.

      ii. Se nello stato $q_{statoCorretto}$ legge simboli differenti sui nastri N_1 e N_2, allora la quintupla che sta scandendo su N_1 non è quella da eseguire; pertanto, entrando nello stato q_2, sposta la testina di N_1 a destra fino a posizionarla sul primo simbolo successivo al primo '$\oplus$' che incontra e, se tale simbolo non è $\square$, entra nello stato q_1, altrimenti entra nello stato di rigetto.

   (b) se nello stato q_1 legge simboli differenti sui nastri N_1 e N_3, allora la quintupla che sta scandendo su N_1 non è quella da eseguire; pertanto, entrando nello stato q_3, sposta la testina di N_1 a destra fino a posizionarla sul primo simbolo successivo al primo '$\oplus$' che incontra e, se tale simbolo non è $\square$, entra nello stato q_1, altrimenti confronta lo stato attuale che sta leggendo su N_3 con lo stato di accettazione ω_2 di T scritto su N_4 e, se sono uguali, entra nello stato di accettazione, altrimenti entra nello stato di rigetto.

3) Nello stato q_{scrivi}, inizia l'esecuzione della quintupla che ha individuato sul nastro N_1 scrivendo il nuovo simbolo su N_2: dunque, nello stato q_{scrivi}, scrive su N_2 il simbolo che legge su N_1 ed entra nello stato $q_{cambiaStato}$ muovendo a destra di due posizioni (per superare il carattere separatore '-') la testina di N_1.

4) Nello stato $q_{cambiaStato}$, prosegue l'esecuzione della quintupla che ha individuato sul nastro N_1 modificando il contenuto del nastro N_3: dunque, nello stato $q_{cambiaStato}$, scrive su N_3 il simbolo che legge su N_1 ed entra nello stato q_{muovi} muovendo a destra di due posizioni la testina di N_1.

5) Nello stato q_{muovi}, termina l'esecuzione della quintupla che ha individuato sul nastro N_1 muovendo la testina del nastro N_2: dunque, nello stato q_{muovi}, muove la testina di N_2 in accordo con il simbolo letto su N_1 ed entra nello stato $q_{riavvolgi}$ muovendo a sinistra la testina di N_1.

6) Nello stato $q_{riavvolgi}$, riposiziona la testina del nastro N_1 sul primo simbolo a destra del carattere '$\otimes$ in esso contenuto: rimane nello stato $q_{riavvolgi}$ muovendo a sinistra la testina di N_2 fino a quando non legge un $\otimes$ su N_1 e poi entra nello stato q_1 muovendo a destra la testina di N_1.

Osserviamo che la computazione $U(\rho_T,x)$ rigetta ogni volta che U non trova la quintupla da eseguire e lo stato attuale di T (scritto sul nastro N_3) *non* è lo stato di accettazione di T (scritto sul nastro N_4). Dunque, U rigetta il suo input (ρ_T,x) senza *verificare* che la computazione $T(x)$ abbia rigettato. Ciò è in accordo con quanto è stato discusso nel Paragrafo 2.3: in assenza di quintuple eseguibili, se la macchina non si trova nello stato di accettazione, possiamo sempre assumere che l'input venga rigettato.

La descrizione dell'algoritmo sopra riportata è ad alto livello perché utilizza l'insieme degli stati Q della macchina T da simulare come parte dell'alfabeto di lavoro. Poiché U deve simulare le computazioni di *qualsiasi* macchina di

Turing, questa non è, ovviamente, una assunzione ragionevole. Per ovviare a tale problema, assumiamo, allora, che l'alfabeto di lavoro di U sia l'insieme $\Sigma = \{0,1,\oplus,\otimes,-,f,s,d\}$ (ricordiamo che il carattere blank $\square$ è sempre esterno all'alfabeto di lavoro).
Sia $b^Q : Q \to \lceil \log|Q| \rceil$ una funzione che codifica in binario gli stati di T utilizzando per ciascuno di essi $m = \lceil \log|Q| \rceil$ cifre, e, per ogni $\omega \in Q$, indichiamo con $b^Q(\omega) = b_1^Q(\omega) b_2^Q(\omega) \ldots b_m^Q(\omega)$ la codifica di ω. Allora, la descrizione di T che costituirà l'input del nastro N_1 di U è la parola $\beta_T \in \Sigma^*$ descritta di seguito:

$$\beta_T = b^Q(\omega_0) - b^Q(\omega_1) \otimes b^Q(\omega_{1_1}) - b_{1_1} - b_{1_2} - b^Q(\omega_{1_2}) - m_1 \oplus \ldots \oplus b^Q(\omega_{h_1}) - b_{h_1} - b_{h_2} - b^Q(\omega_{h_2}) - m_h \oplus$$

Nella descrizione ad alto livello di U proposta sopra, dobbiamo, per così dire, "raffinare" la descrizione delle operazioni compiute quando la macchina U si trova negli stati q_0, in cui esegue copia degli stati ω_0 e ω_1 di T, rispettivamente, sui nastri N_3 e N_4, e q_1, in cui cerca una quintupla di T da eseguire: le operazioni di copia e di confronto di stati che nella descrizione ad alto livello erano operazioni atomiche diventano ora operazioni da eseguire mediante cicli. Le modifiche sono descritte di seguito:

1') A partire dallo stato q_0, vengono copiati gli m caratteri della codifica $b^Q(\omega_0)$ di ω_0 sul nastro N_3 e gli m caratteri della codifica $b^Q(\omega_1)$ di ω_1 sul nastro N_4; successivamente le testine di N_3 e di N_4 vengono spostate a sinistra sul primo carattere scritto, la testina di N_1 viene spostata sul simbolo a destra del primo carattere '$\otimes$' che incontra e la macchina entra nello stato q_1:

$$\begin{array}{ll}
\langle q_0,(x,a,\square,\square),(x,a,x,\square),q_0,(d,f,d,f)\rangle & \forall x \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \\
\langle q_0,(-,a,\square,\square),(-,a,\square,\square),q_{01},(d,f,f,f)\rangle & \forall a \in \{0,1,\square\}, \\
\langle q_{01},(y,a,\square,\square),(y,a,\square,y),q_{01},(d,f,f,d)\rangle & \forall y \in \{0,1\} \wedge \forall a \in \{0,1,\square\}, \\
\langle q_{01},(\otimes,a,\square,\square),(\otimes,a,\square,\square),q_{02},(d,f,s,s)\rangle & \forall a \in \{0,1,\square\}, \\
\langle q_{02},(b,a,x,y),(x,a,y,z),q_{02},(f,f,s,s)\rangle & \forall x,y \in \{0,1\} \wedge \forall a,b \in \{0,1,\square\}, \\
\langle q_{02},(b,a,\square,\square),(z,a,\square,\square),q_1,(f,f,d,d)\rangle & \forall a,b \in \{0,1,\square\}.
\end{array}$$

2') Nello stato q_1 ha inizio la ricerca di una quintupla su N_1 che abbia come primo simbolo la parola scritta sul nastro N_3 e come secondo simbolo lo stesso simbolo letto dalla testina di N_2:

(a) se nello stato q_1 legge la stessa sequenza di simboli sui nastri N_1 ed N_3 fino a quando incontra il carattere '-' su N_1 e il carattere $\square$ su N_3 allora, a questo punto, sposta la testina di N_1 a destra di due posizioni, la testina di N_3 a sinistra di m posizioni, ed entra nello stato $q_{statoCorretto}$:

$$\begin{array}{ll}
\langle q_1,(x,a,x,y),(x,a,x,y),q_1,(d,f,d,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \\
\langle q_1,(-,a,\square,y),(-,a,\square,y),q_{11},(d,f,s,f)\rangle & \forall y \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \\
\langle q_{11},(b,a,x,y),(b,a,x,y),q_{11},(f,f,s,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a,b \in \{0,1,\square\} \\
\langle q_{11},(b,a,\square,y),(b,a,\square,y),q_{statoCorretto},(f,f,d,f)\rangle & \forall y \in \{0,1\} \wedge \forall a,b \in \{0,1,\square\}
\end{array}$$

Ora la testina di N_1 è posizionata sul secondo elemento (ossia, il carattere letto) della quintupla che si sta esaminando, e la testina di N_3 è nuovamente posizionata sul primo carattere della parola che rappresenta il nuovo stato attuale di T.

i. Se nello stato $q_{statoCorretto}$ legge lo stesso simbolo sui nastri N_1 e N_2, allora ha trovato la quintupla da eseguire; pertanto, sposta la testina di N_1 a destra di due posizioni (per superare il carattere separatore '-') ed entra nello stato q_{scrivi}:

$$\begin{array}{ll}
\langle q_{statoCorretto},(a,a,x,y),(a,a,x,y),q'_{statoCorretto},(d,f,f,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \\
\langle q'_{statoCorretto},(-,a,x,y),(-,a,x,y),q_{scrivi},(d,f,f,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a \in \{0,1,\square\}.
\end{array}$$

ii. Se nello stato $q_{statoCorretto}$ legge simboli differenti sui nastri N_1 e N_2, allora la quintupla che sta scandendo su N_1 non è quella da eseguire; pertanto, entrando nello stato q_2, sposta la testina di N_1 a destra fino a posizionarla sul primo simbolo successivo al primo '$\oplus$' che incontra e, se tale simbolo è 0 oppure 1 allora entra nello stato q_1, altrimenti entra nello stato di rigetto:

$$\begin{array}{ll}
\langle q_{statoCorretto},(b,a,x,y),(b,a,x,y),q_2,(d,f,f,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a,b \in \{0,1,\square\} : a \neq b \\
\langle q_2,(z,a,x,y),(z,a,x,y),q_2,(d,f,f,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \wedge \forall z \in \{0,1,-\} \\
\langle q_2,(\oplus,a,x,y),(\oplus,a,x,y),q_{21},(d,f,f,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \\
\langle q_{21},(z,a,x,y),(z,a,x,y),q_1,(f,f,f,f)\rangle & \forall x,y,z \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \\
\langle q_{21},(z,a,x,y),(z,a,x,y),q_R,(f,f,f,f)\rangle & \forall x,y \in \{0,1\} \wedge \forall a \in \{0,1,\square\} \wedge \forall z \notin \{0,1\}.
\end{array}$$

(b) se nello stato q_1 legge simboli differenti sui nastri N_1 e N_3, allora la quintupla che sta scandendo su N_1 non è quella da eseguire; pertanto, entrando nello stato q_3, sposta la testina di N_3 a sinistra fino a posizionarla sul primo simbolo non $\square$ ivi scritto, sposta la testina di N_1 a destra fino a posizionarla sul primo simbolo successivo al primo '$\oplus$' che incontra e, se tale simbolo è 0 oppure 1 allora entra nello stato q_1, altrimenti confronta lo stato attuale che sta leggendo su N_3 con lo stato di accettazione ω_1 di T scritto su N_4 e, se sono uguali, entra nello stato di accettazione, altrimenti entra nello stato di rigetto:

$$\begin{array}{ll}
\langle q_1,(z,a,x,y),(z,a,x,y),q_3,(f,f,s,f)\rangle & \forall x,y,z\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\} : z\neq x\\
\langle q_3,(z,a,x,y),(z,a,x,y),q_3,(f,f,s,f)\rangle & \forall x,y,z\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_3,(z,a,\square,y),(z,a,\square,y),q_{31},(f,f,d,f)\rangle & \forall y,z\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_{31},(z,a,x,y),(z,a,x,y),q_{31},(d,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma-\{\oplus\}\\
\langle q_{31},(\oplus,a,x,y),(\oplus,a,x,y),q_{32},(d,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_{32},(z,a,x,y),(z,a,x,y),q_1,(f,f,f,f)\rangle & \forall x,y,z\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_{32},(z,a,x,y),(z,a,x,y),q_{33},(f,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\notin\{0,1\}\\
\langle q_{33},(z,a,x,x),(z,a,x,x),q_{33},(f,f,d,d)\rangle & \forall x\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma\\
\langle q_{33},(z,a,\square,\square),(z,a,\square,\square),q_A,(f,f,f,f)\rangle & \forall x\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma\\
\langle q_{33},(z,a,x,y),(z,a,x,y),q_R,(d,f,f,f)\rangle & \forall x,y\in\{0,1\} : x\neq y\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma.
\end{array}$$

3) Nello stato q_{scrivi}, inizia l'esecuzione della quintupla che ha individuato sul nastro N_1 scrivendo il nuovo simbolo su N_2: dunque, nello stato q_{scrivi}, scrive su N_2 il simbolo che legge su N_1 ed entra nello stato $q_{cambiaStato}$ muovendo a destra di due posizioni (per superare il carattere separatore '-') la testina di N_1:

$$\begin{array}{ll}
\langle q_{scrivi},(b,a,x,y),(b,b,x,y),q'_{scrivi},(d,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q'_{scrivi},(-,a,x,y),(-,a,x,y),q_{cambiaStato},(d,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}
\end{array}$$

.

4') Nello stato $q_{cambiaStato}$, prosegue l'esecuzione della quintupla che ha individuato sul nastro N_1 modificando il contenuto del nastro N_3: dunque, nello stato $q_{cambiaStato}$, scrive su N_3 la sequenza di simboli che legge su N_1 fino a quando incontra il carattere '-' su N_1 e il carattere $\square$ su N_3; infine, sposta a destra di una posizione la testina di N_1 (che così si posiziona su uno dei caratteri 's', 'f', 'd' che indicano lo spostamento della testina di T che deve essere simulato sul nastro N_1), sposta a sinistra la testina di N_3 fino a posizionarla sul carattere a destra del primo $\square$ che incontra, ed entra nello stato q_{muovi}:

$$\begin{array}{ll}
\langle q_{cambiaStato},(z,a,x,y),(z,a,z,y),q_{cambiaStato},(d,f,d,f)\rangle & \forall x,y,z\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_{cambiaStato},(-,a,\square,y),(-,a,\square,y),q_4,(d,f,s,f)\rangle & \forall y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_4,(z,a,x,y),(z,a,x,y),q_4,(f,f,s,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma\\
\langle q_4,(z,a,\square,y),(z,a,\square,y),q_{muovi},(f,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma.
\end{array}$$

5) Nello stato q_{muovi}, termina l'esecuzione della quintupla che ha individuato sul nastro N_1 muovendo la testina del nastro N_2: dunque, nello stato q_{muovi}, muove la testina di N_2 in accordo con il simbolo letto su N_1 ed entra nello stato $q_{riavvolgi}$ muovendo a sinistra la testina di N_1:

$$\begin{array}{ll}
\langle q_{muovi},(s,a,x,y),(s,a,z,y),q_{riavvolgi},(f,f,s,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_{muovi},(f,a,x,y),(f,a,z,y),q_{riavvolgi},(f,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\\
\langle q_{muovi},(d,a,x,y),(d,a,z,y),q_{riavvolgi},(f,f,d,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}.
\end{array}$$

6) Nello stato $q_{riavvolgi}$, riposiziona la testina del nastro N_1 sul primo simbolo a destra del carattere '$\otimes$' in esso contenuto rientrando, infine, nello stato q_1:

$$\begin{array}{ll}
\langle q_{riavvolgi},(z,a,x,y),(z,a,x,y),q_{riavvolgi},(s,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}\ \wedge\ \forall z\in\Sigma-\{\otimes\}\\
\langle q_{riavvolgi},(\otimes,a,x,y),(\otimes,a,x,y),q_1,(d,f,f,f)\rangle & \forall x,y\in\{0,1\}\ \wedge\ \forall a\in\{0,1,\square\}.
\end{array}$$