

Intelligenza Artificiale I

Esercitazione 2

Marco Piastra

Macchina di Turing (A. Turing, 1937)

- Un modello astratto per il calcolo effettivo

Un nastro con celle elementari, un simbolo in ogni cella

Una testina di lettura e scrittura delle celle, può muoversi

Uno stato della macchina (un simbolo)

Una CPU, governata da una **tabella di transizione**

$\{ \langle \text{stato attuale, lettura cella} \rangle \rightarrow \langle \text{nuovo stato, scrittura cella, movimento} \rangle \}$

La CPU è una *finite state machine*

L'input è il contenuto della cella

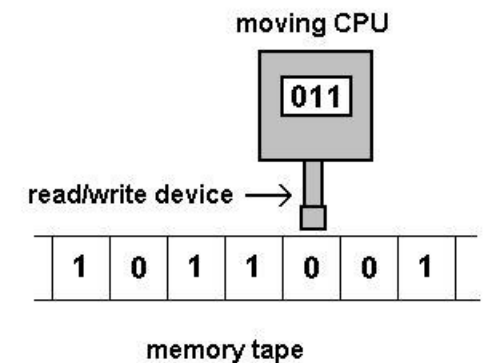
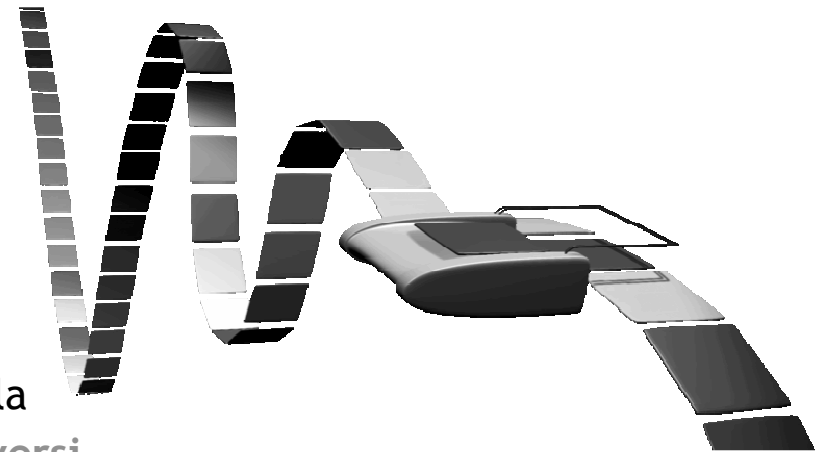
L'output è il nuovo contenuto della cella e il movimento

La macchina si ferma se non trova nella tabella una entry con chiave $\langle \text{stato attuale, lettura cella} \rangle$

- Descrive la potenza di calcolo di un computer

Qualunque **funzione** sia calcolabile da un computer, è calcolabile da una macchina di Turing

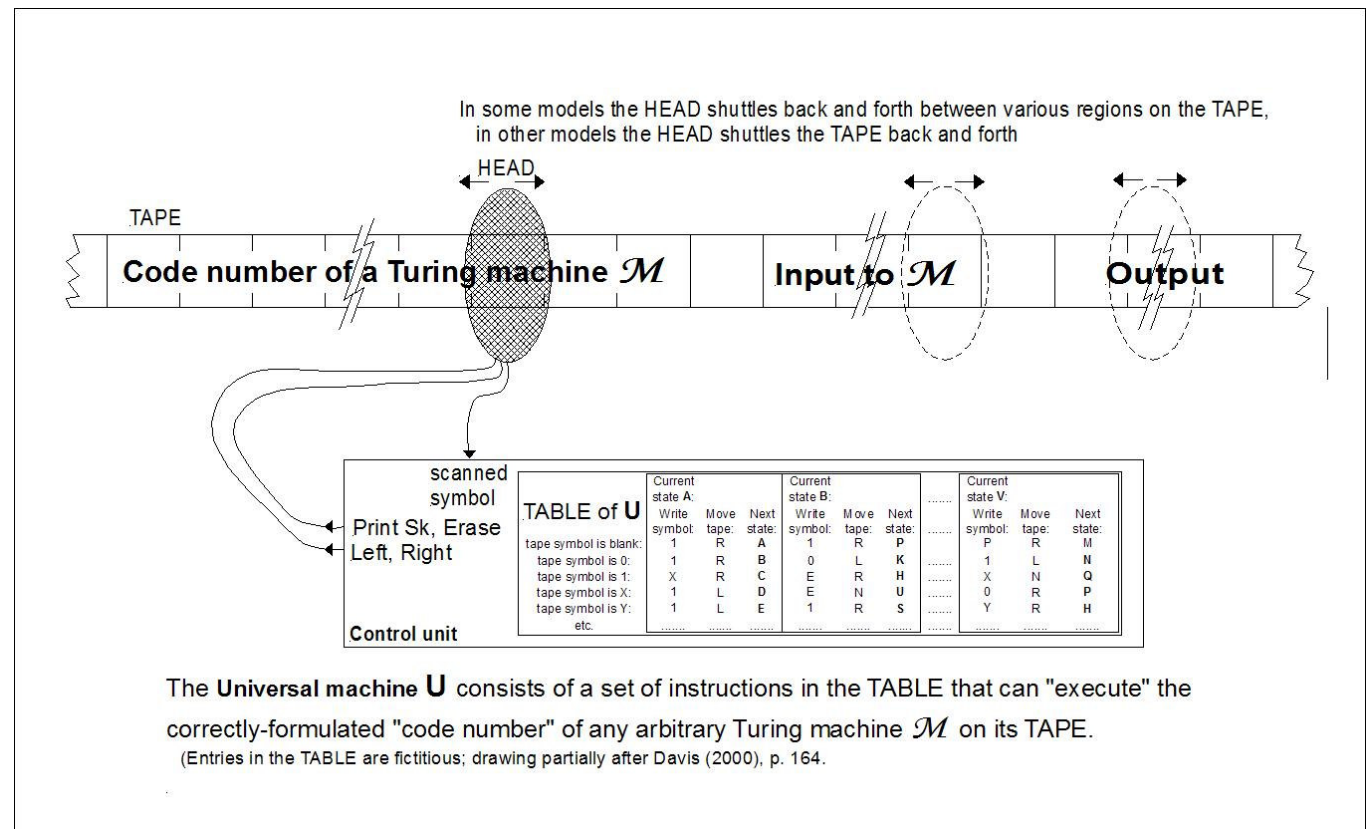
L'unica 'licenza' rispetto ad una macchina fisica è il nastro di lunghezza illimitata

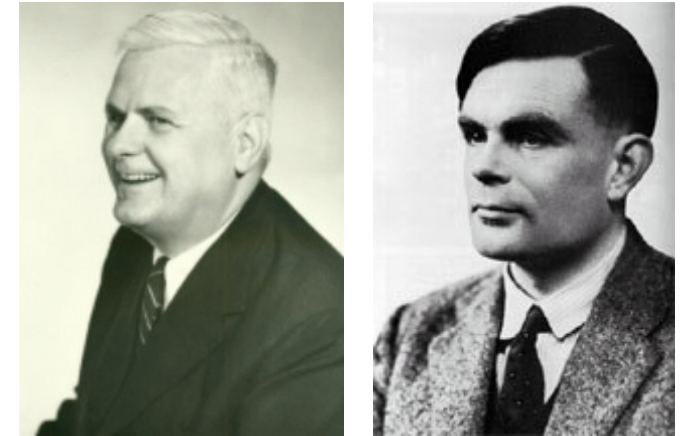


Macchina di Turing *universale* (A. Turing, 1937)

- Come il modello base, ma la tabella è sul nastro
 La tabella delle transizioni viene caricata dal nastro, all'inizio del calcolo

Nello stato iniziale, la tabella della macchina contiene solo le istruzioni necessarie a caricare la tabella scritta sul nastro





Tesi di Church-Turing

*Non esiste un'unica, concisa formulazione originale:
si tratta di un concetto espresso in più passaggi, da integrare con risultati teorici*

- Una possibile formulazione (Wikipedia)

“Every 'function which would naturally be regarded as computable' can be computed by a Turing machine.”

*La vaghezza della formulazione ha dato luogo a diverse interpretazioni,
una molto comune (e non del tutto esatta) può essere espressa come in (Wikipedia):*

“Every 'function that could be physically computed' can be computed by a Turing machine.”

Searle: “... At present, obviously, the metaphor is the digital computer.”

assert

(file turingmachine.jess)

- Asserisce un fatto in memoria

Esempio:

```
(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
  =>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
             (current-state ?ns))
  (assert (move ?tm ?mv))
)
```

Il fatto viene inserito in memoria

retract

(file turingmachine.jess)

- Ritratta (rimuove) un fatto dalla memoria

Esempio:

```
(defrule move-right
  (declare (salience 1))
  ?action <- (move ?tm right)
  ?tm <- (tm (left-part $?rest-left)
            (current-square ?sym)
            (right-part ?sym-right $?rest-right))

=>
  (modify ?tm (left-part ?sym ?rest-left)
             (current-square ?sym-right)
             (right-part ?rest-right))
  (retract ?action) Il fatto viene rimosso dalla memoria
)
```

retract

(file turingmachine.jess)

- Ritratta (rimuove) un fatto dalla memoria

Esempio:

```

La variabile (defrule move-right
deve essere (declare (salience 1))
legata al fatto ?action <- (move ?tm right)
da rimuovere ?tm <- (tm (left-part $?rest-left)
                        (current-square ?sym)
                        (right-part ?sym-right $?rest-right))
=>
                (modify ?tm (left-part ?sym ?rest-left)
                    (current-square ?sym-right)
                    (right-part ?rest-right))
                (retract ?action) Il fatto viene rimosso dalla memoria
)

```

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```
(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
=>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
            (current-state ?ns))
  (assert (move ?tm ?mv))
)
```

Variabile con un vincolo

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```

      Variabile
      (defrule event
        ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
                (current-square ?is))
        (event (current-state ?cs)
              (input-symbol ?is)
              (output-symbol ?os)
              (new-state ?ns)
              (head-move ?mv))
=>
        (printout t "From state " ?cs " input " ?is
                  " to state " ?ns " output " ?os " move " ?mv crlf)
        (modify ?tm (current-square ?os)
                  (current-state ?ns))
        (assert (move ?tm ?mv))
      )

```

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```

(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
=>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
             (current-state ?ns))
  (assert (move ?tm ?mv))
)

```

Vincolo

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```
(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
=>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
            (current-state ?ns))
  (assert (move ?tm ?mv))
)
```

Sintassi generale: &:<vincolo> (in forma positiva)

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```
(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
=>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
             (current-state ?ns))
  (assert (move ?tm ?mv))
)
```

Sintassi generale: &~:<vincolo> (in forma negativa)

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```
(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
=>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
             (current-state ?ns))
  (assert (move ?tm ?mv))
)
```

In questo caso il vincolo è negativo

Significato:
la variabile `?cs` si
lega ad un valore
che non sia `eq` a `halt`

Vincoli sulle variabili

(file turingmachine.jess)

- Limitano le possibilità di legarsi a dei valori

Esempio:

```
(defrule event
  ?tm <- (tm (current-state ?cs&~:(eq ?cs halt))
           (current-square ?is))
  (event (current-state ?cs)
         (input-symbol ?is)
         (output-symbol ?os)
         (new-state ?ns)
         (head-move ?mv))
=>
  (printout t "From state " ?cs " input " ?is
            " to state " ?ns " output " ?os " move " ?mv crlf)
  (modify ?tm (current-square ?os)
            (current-state ?ns))
  (assert (move ?tm ?mv))
)
```

In questo caso il vincolo è negativo

Significato:
la variabile `?cs` si
lega ad un valore
che non sia `eq` a `halt`

(Altri esempi in
futuro)

salience

(file turingmachine.jess)

- Modifica la priorità della regola

Esempio:

```
(defrule move-right
  (declare (salience 1))
  ?action <- (move ?tm right)
  ?tm <- (tm (left-part $?rest-left)
           (current-square ?sym)
           (right-part ?sym-right $?rest-right))
  =>
  (modify ?tm (left-part ?sym ?rest-left)
            (current-square ?sym-right)
            (right-part ?rest-right))
  (retract ?action)
)
```

Non è una condizione, è una dichiarazione

salience

(file turingmachine.jess)

- Modifica la priorità della regola

Esempio:

```
(defrule move-right
  (declare (salience 1)) E' opzionale
  ?action <- (move ?tm right)
  ?tm <- (tm (left-part $?rest-left)
           (current-square ?sym)
           (right-part ?sym-right $?rest-right))

=>
  (modify ?tm (left-part ?sym ?rest-left)
            (current-square ?sym-right)
            (right-part ?rest-right))
  (retract ?action)
)
```


salience

(file turingmachine.jess)

- Modifica la priorità della regola

Esempio:

```
(defrule move-right
  (declare (salience 1))
  ?action <- (move ?tm right)
  ?tm <- (tm (left-part $?rest-left)
           (current-square ?sym)
           (right-part ?sym-right $?rest-right))
  =>
  (modify ?tm (left-part ?sym ?rest-left)
            (current-square ?sym-right)
            (right-part ?rest-right))
  (retract ?action)
)
```

Definisce la priorità della regola

salience

(file turingmachine.jess)

- Modifica la priorità della regola

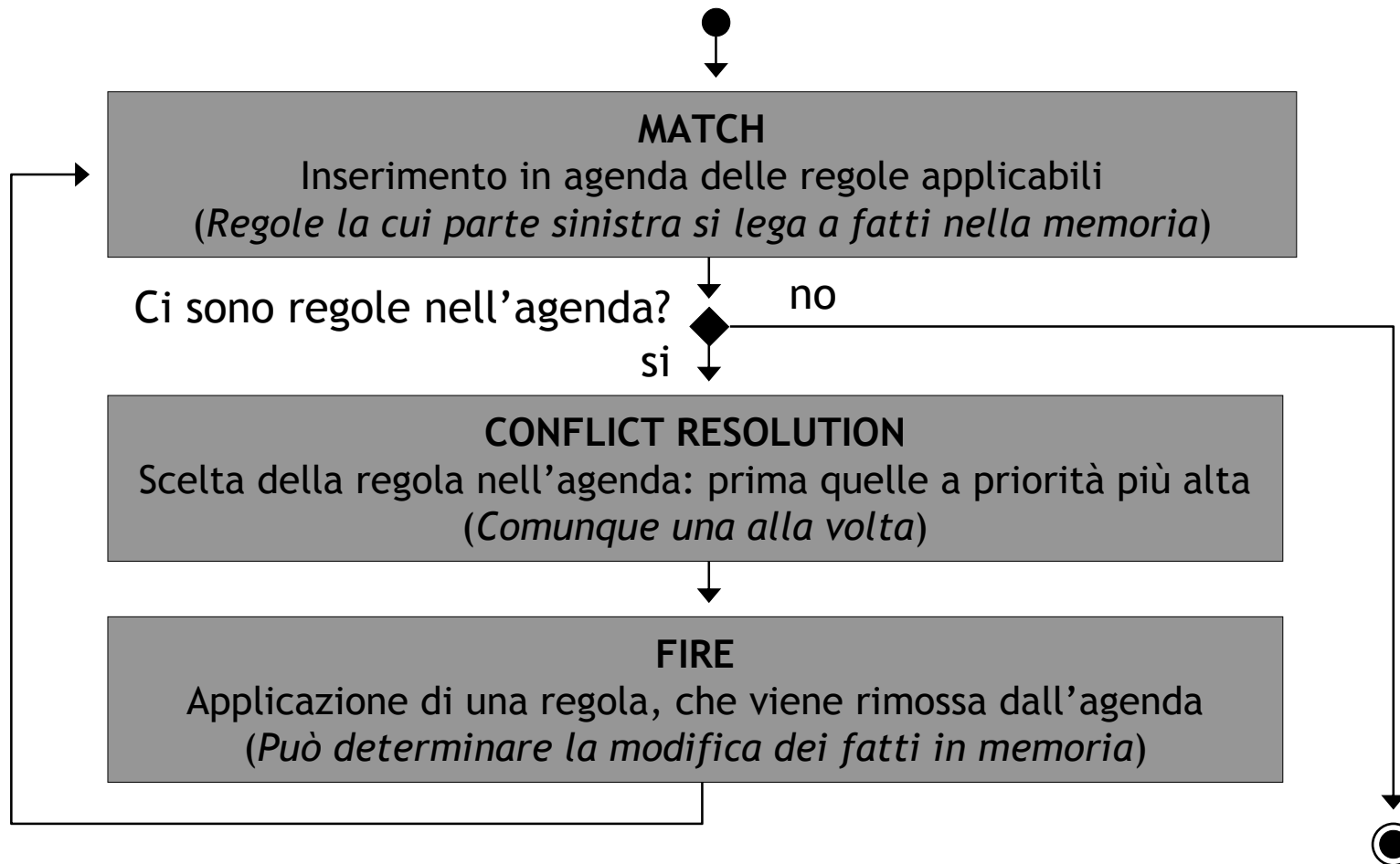
Esempio:

```
(defrule move-right
  (declare (salience 1))
  ?action <- (move ?tm right)
  ?tm <- (tm (left-part $?rest-left)
           (current-square ?sym)
           (right-part ?sym-right $?rest-right))
  =>
  (modify ?tm (left-part ?sym ?rest-left)
            (current-square ?sym-right)
            (right-part ?rest-right))
  (retract ?action)
)
```

La priorità è un valore relativo, per default è 0

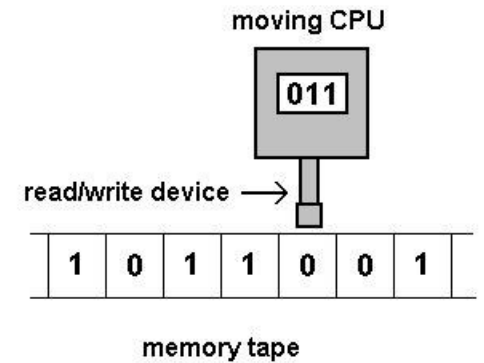
Agenda (Come funziona Jess - seconda approssimazione)

- L'agenda contiene le regole applicabili



Macchina di Turing

(file turingmachine.jess)

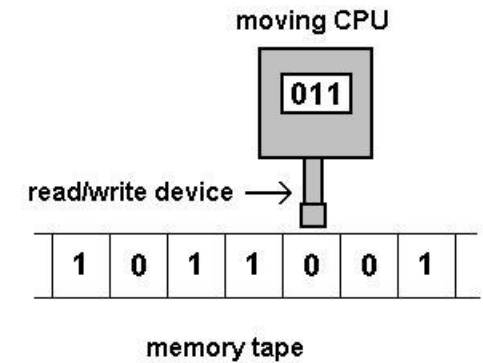


- Un'automa a stati finiti + una testina di lettura/scrittura + un nastro infinito

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

Macchina di Turing

(file turingmachine.jess)

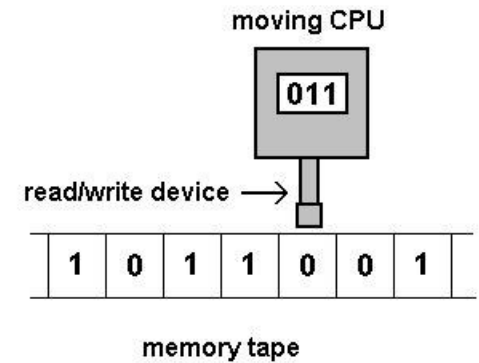


- Un'automa a stati finiti + una testina di lettura/scrittura + un nastro infinito

```
(deftemplate tm
  (slot current-state) Stato attuale dell'automa
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

Macchina di Turing

(file turingmachine.jess)



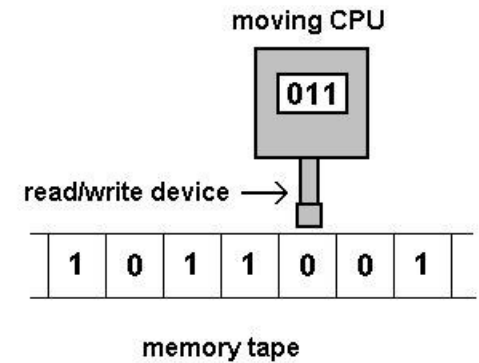
- Un'automa a stati finiti + una testina di lettura/scrittura + un nastro infinito

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

Parte sinistra del nastro

Macchina di Turing

(file turingmachine.jess)



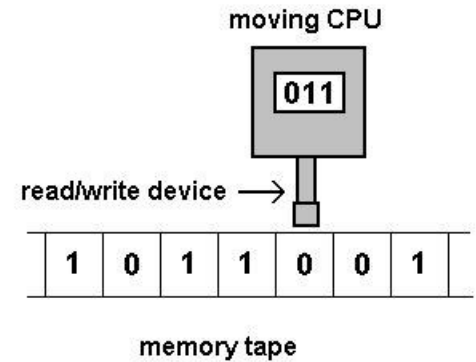
- Un'automa a stati finiti + una testina di lettura/scrittura + un nastro infinito

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

Cella corrispondente alla testina

Macchina di Turing

(file turingmachine.jess)

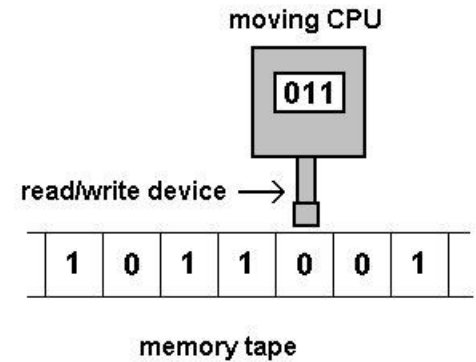


- Un'automa a stati finiti + una testina di lettura/scrittura + un nastro infinito

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part) Parte destra del nastro
)
```


Macchina di Turing

(file turingmachine.jess)



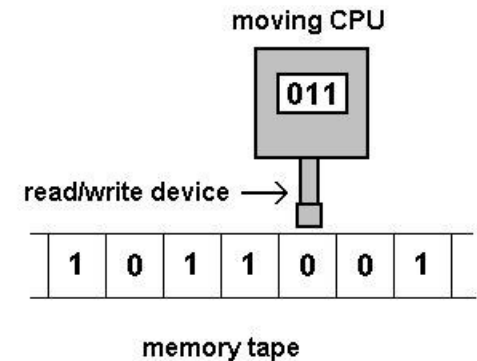
- La tabella delle transizioni descrive anche le mosse della testina

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Macchina di Turing

(file turingmachine.jess)



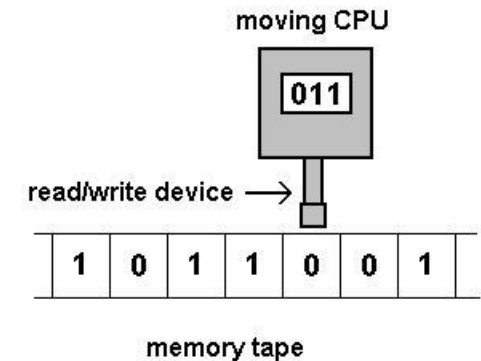
- La tabella delle transizioni descrive anche le mosse della testina

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event Schema, descrive una singola voce nella tabella
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Macchina di Turing

(file turingmachine.jess)



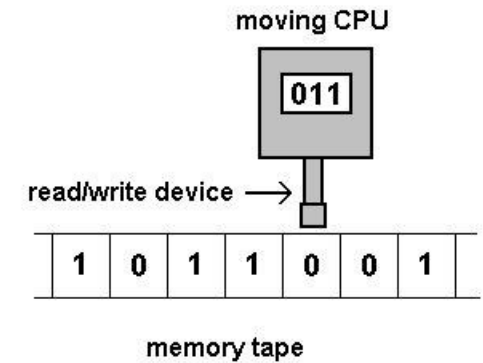
- La tabella delle transizioni descrive anche le mosse della testina

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state) Stato attuale dell'automa
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

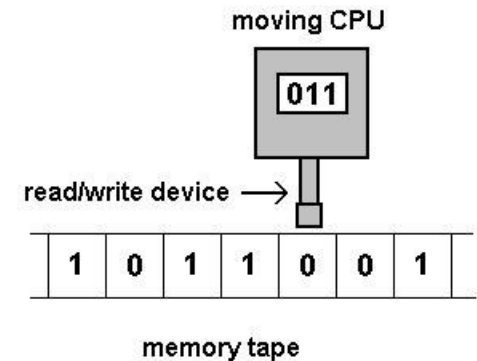
```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Simbolo letto dalla testina

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

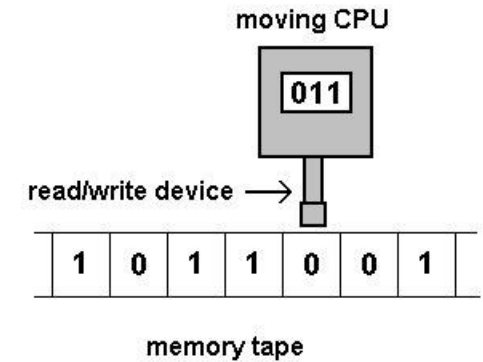
```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Simbolo da scrivere nella cella

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

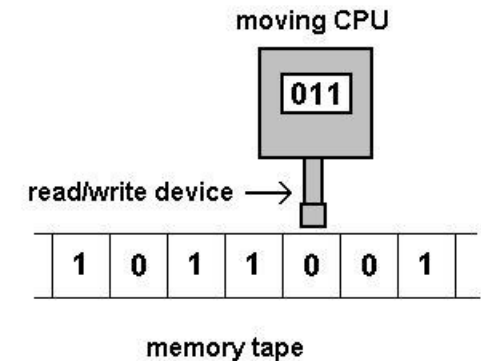
```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Nuovo stato dell'automa

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

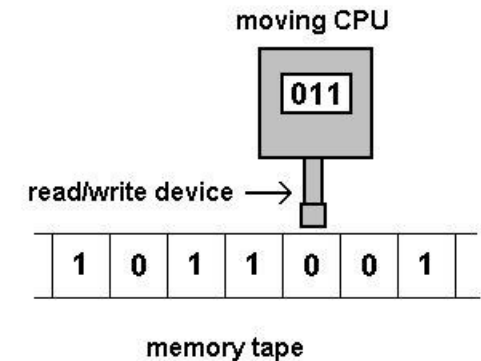
```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Movimento della testina: **left**, **right**, **none**

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

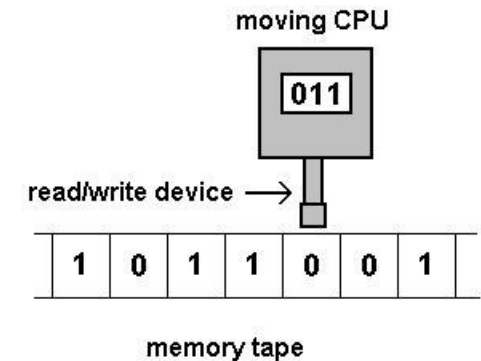
```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

Lo stato `halt` è uno stato speciale:
provoca l'arresto della macchina

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```


Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

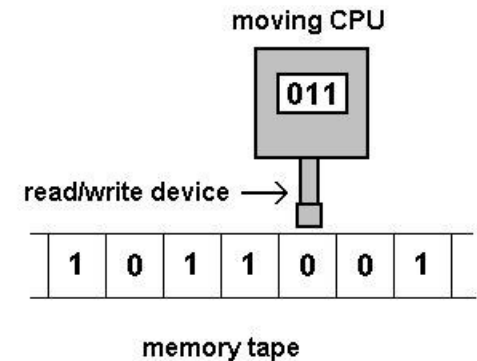
Lo stato `halt` è uno stato speciale:
provoca l'arresto della macchina

E' lo stato in cui ci si aspetta di trovare
la macchina al termine dell'esecuzione

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

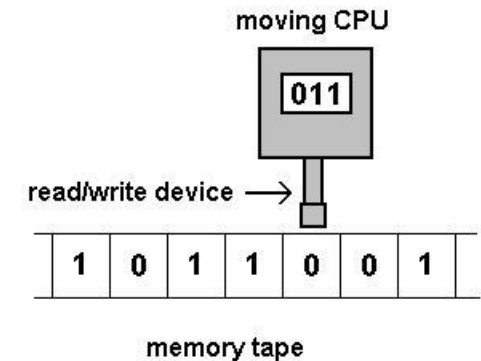
```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Stato **halt** a parte, la tabella deve contenere una voce per ogni stato corrente e simbolo letto

Macchina di Turing

(file turingmachine.jess)



- La tabella delle transizioni descrive anche le mosse della testina

```
(deftemplate tm
  (slot current-state)
  (multislot left-part)
  (slot current-square)
  (multislot right-part)
)
```

```
(deftemplate event
  (slot current-state)
  (slot input-symbol)
  (slot output-symbol)
  (slot new-state)
  (slot head-move)
)
```

Stato **halt** a parte, la tabella deve contenere una voce per ogni stato corrente e simbolo letto

La mancanza di una voce manda la macchina in errore (è un *bug* del programma)

Macchina di Turing

(file turingmachine.jess)

- Domande:

- Come funzionano le regole del programma?
- Come fa il programma a simulare un nastro infinito (nelle due direzioni)?
- Perchè sono definite delle priorità tra regole?
- Cosa succederebbe se eliminassimo le priorità?
per provare, prima di **(run)** inserire **(set-strategy breadth)**

(usare l'esempio paritychecker-tm.jess)

