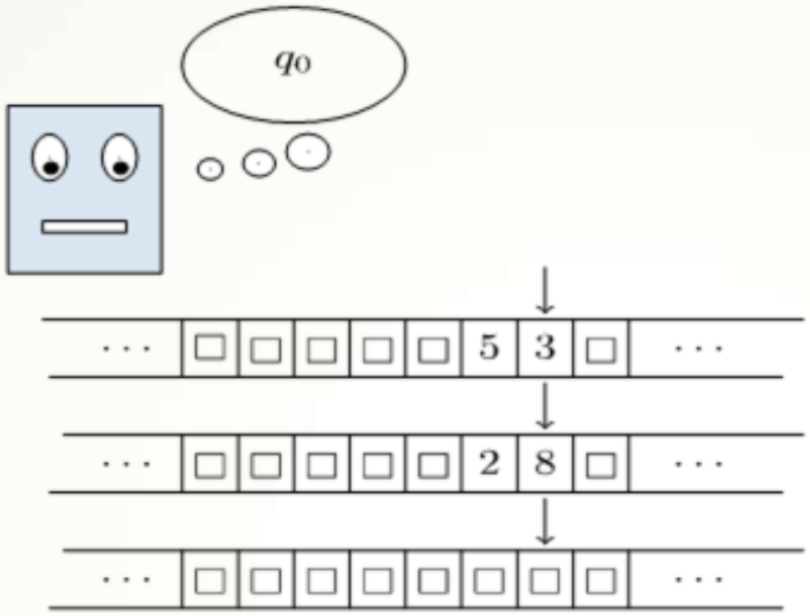
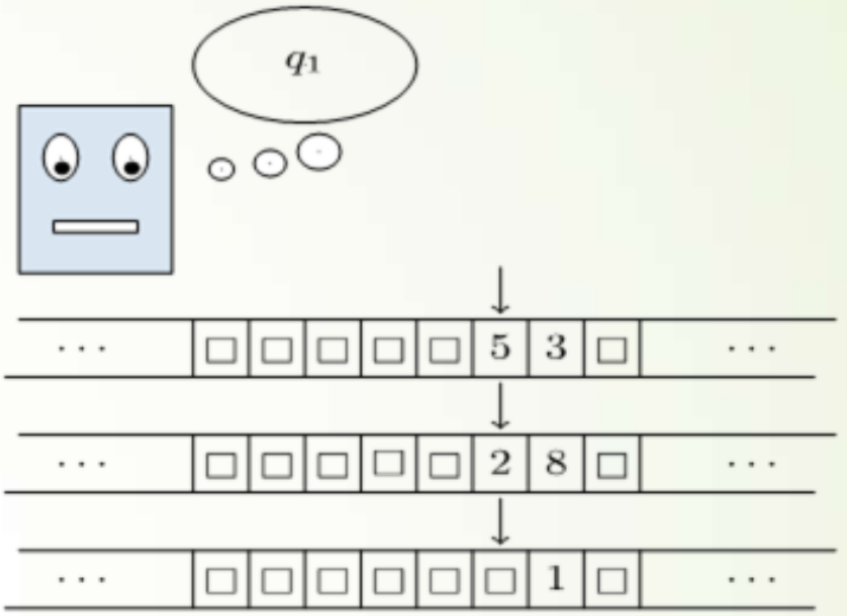
+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13
5	5	6	7	8	9	10	11	12	13	14
6	6	7	8	9	10	11	12	13	14	15
7	7	8	9	10	11	12	13	14	15	16
8	8	9	10	11	12	13	14	15	16	17
9	9	10	11	12	13	14	15	16	17	18

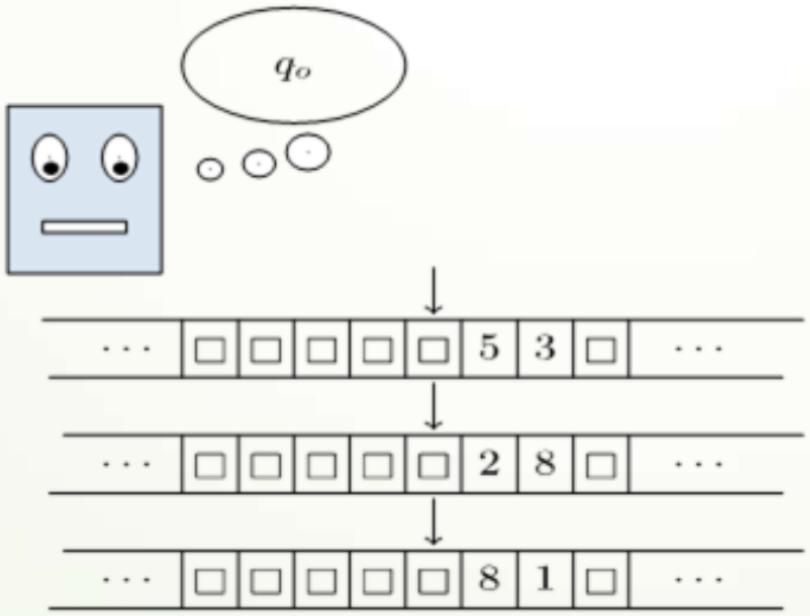
+	0	1	2	…	**37895**		1000000
0	0	1	2	...	37895	...	1000000
1	1	2	3	...	37896	...	1000001
2	2	3	4	...	37897	...	1000002
…	…	…	…	…		…	
441238	441238	441239	441240	...	**479133**	...	1441238
…	…	…	…	…		…	
1000000	1000000	1000001	1000002	...	1037895	...	2000000



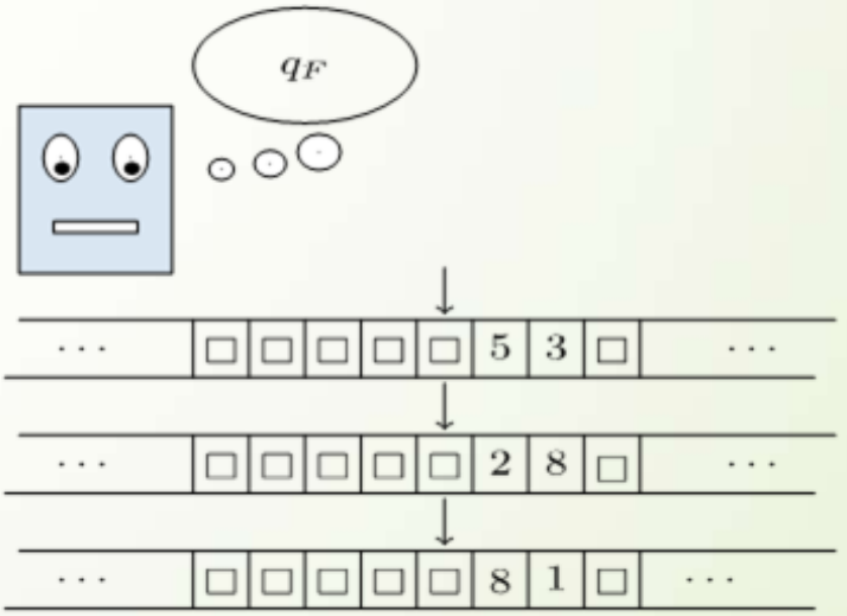
(a)



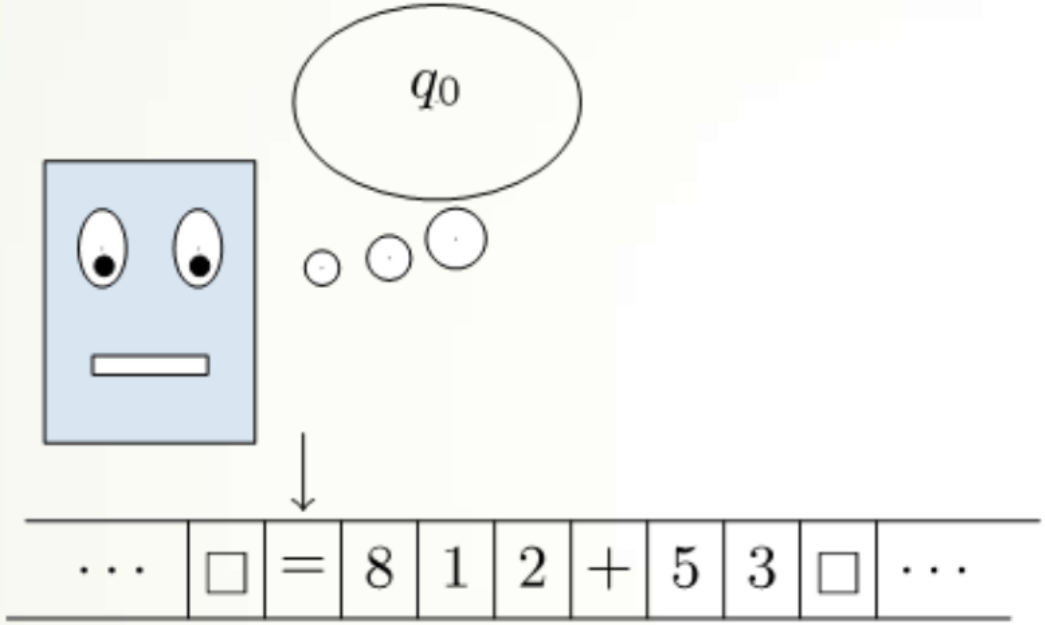
(b)



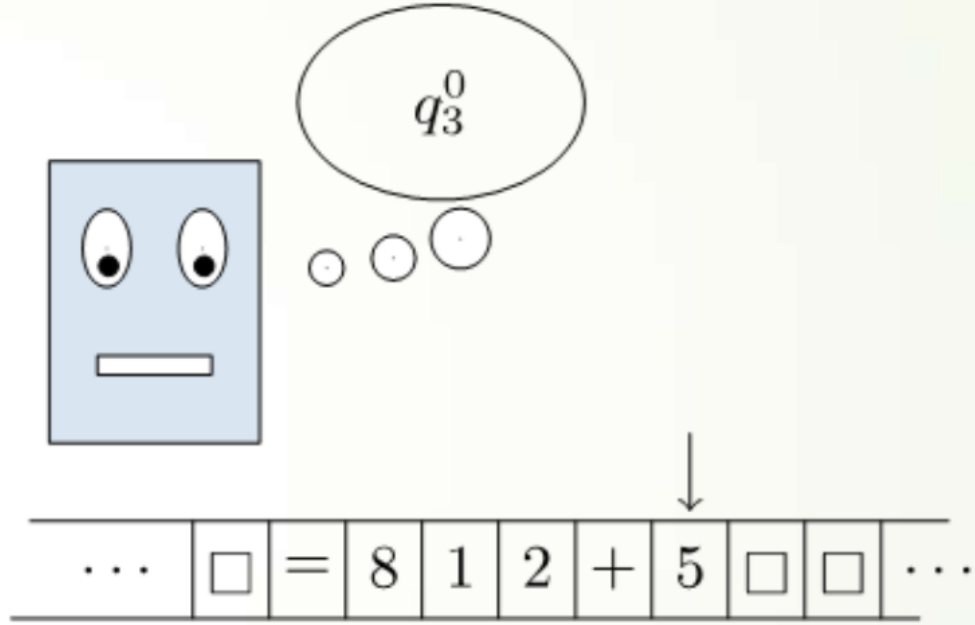
(c)



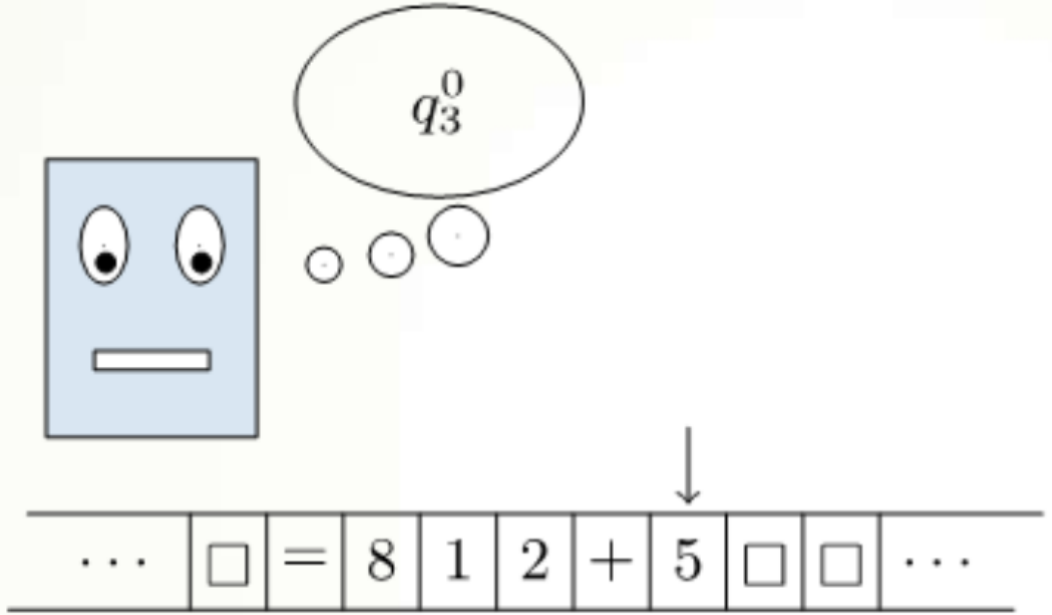
(d)



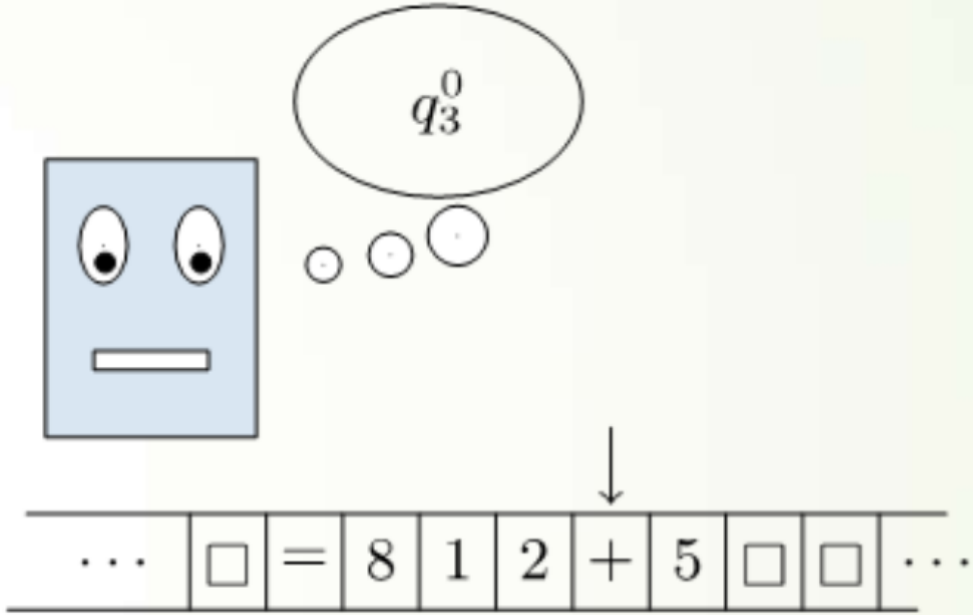
(a)



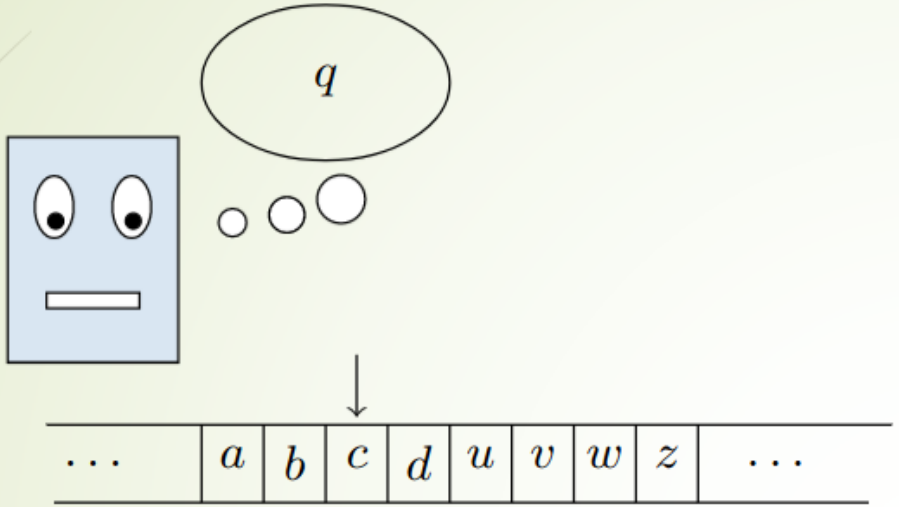
(b)

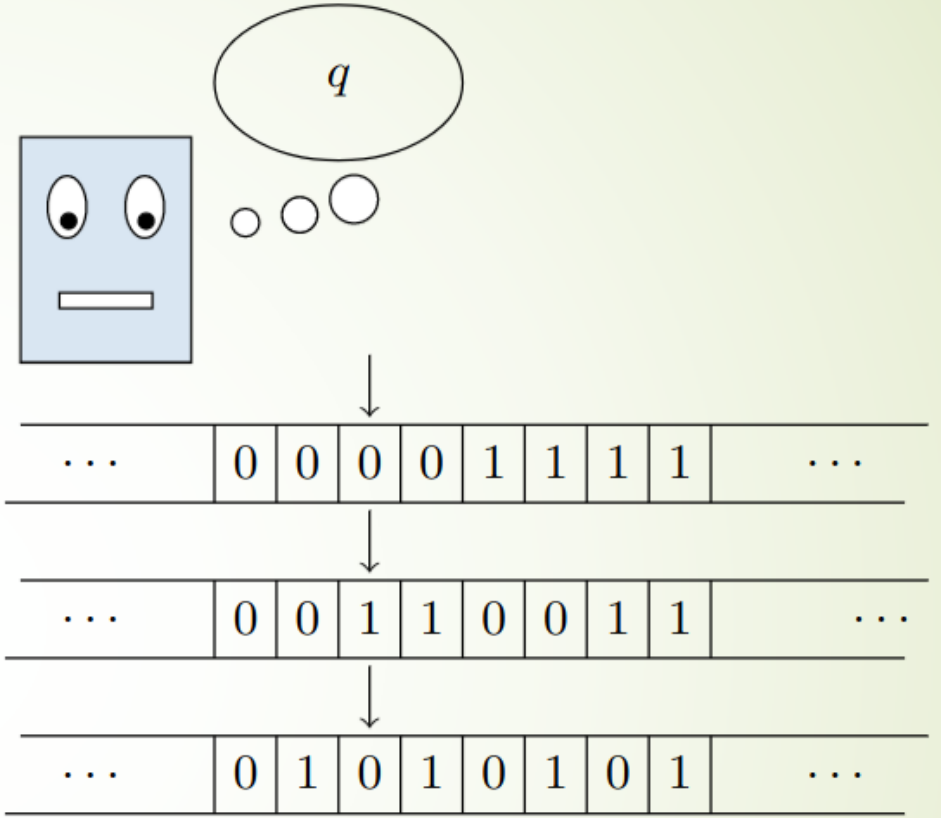


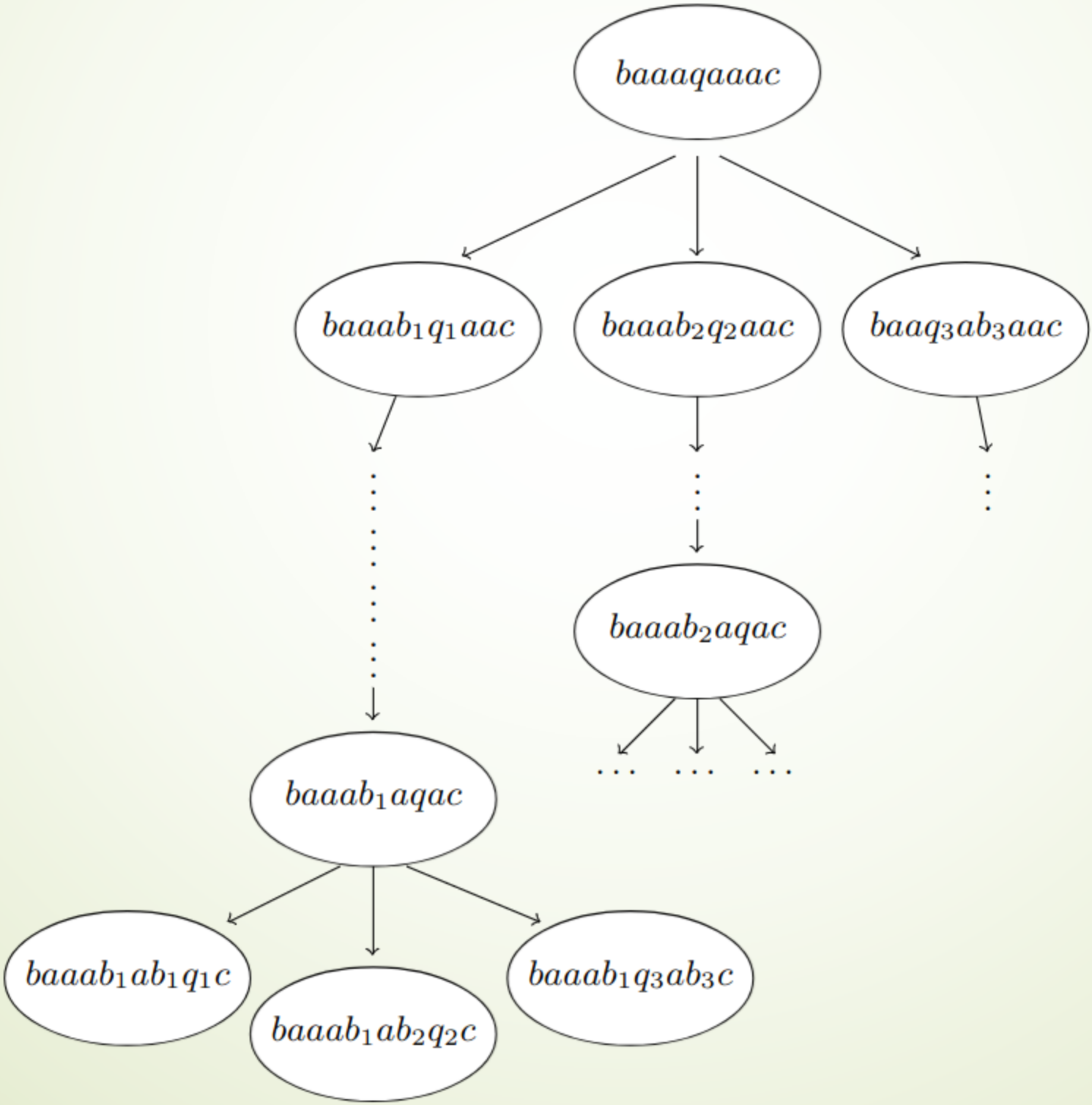
(a)

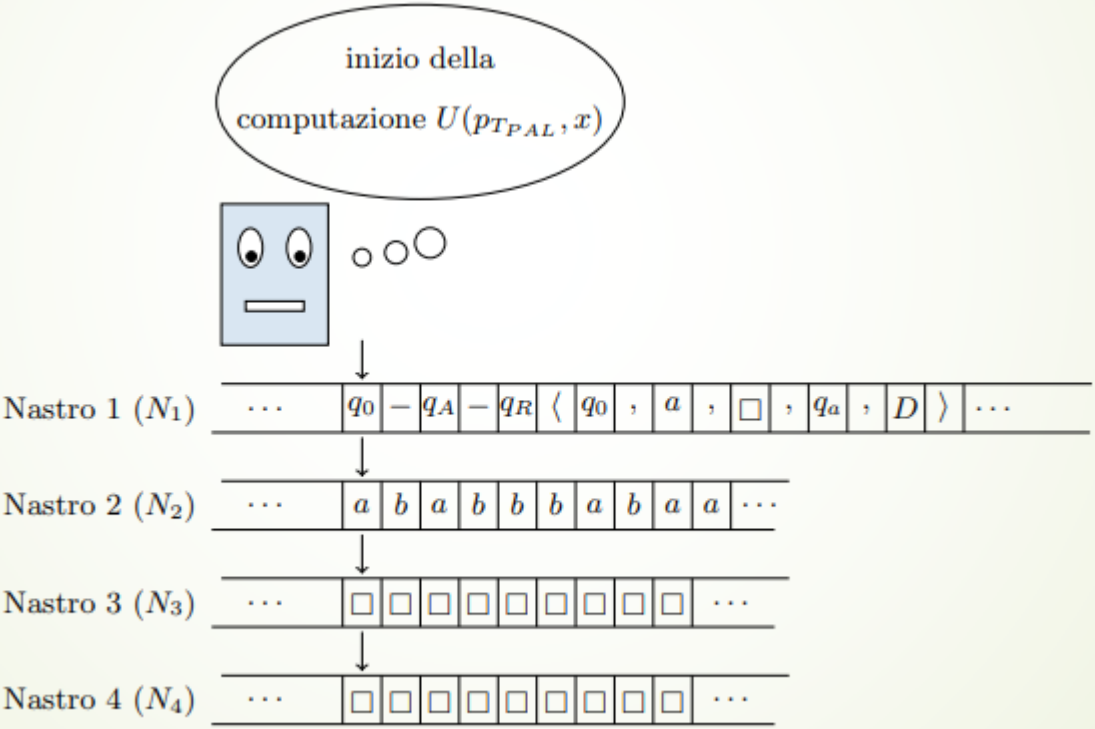


(b)


q
... a b c d u v w z ...


q
... 0 0 0 0 1 1 1 1 ...
... 0 0 1 1 0 0 1 1 ...
... 0 1 0 1 0 1 0 1 ...


$baaaqaaac$
$baaab_1q_1aac$
$baaab_2q_2aac$
$baaq_3ab_3aac$
$baaab_2aqac$
$baaab_1aqac$
$baaab_1ab_1q_1c$
$baaab_1ab_2q_2c$
$baaab_1q_3ab_3c$

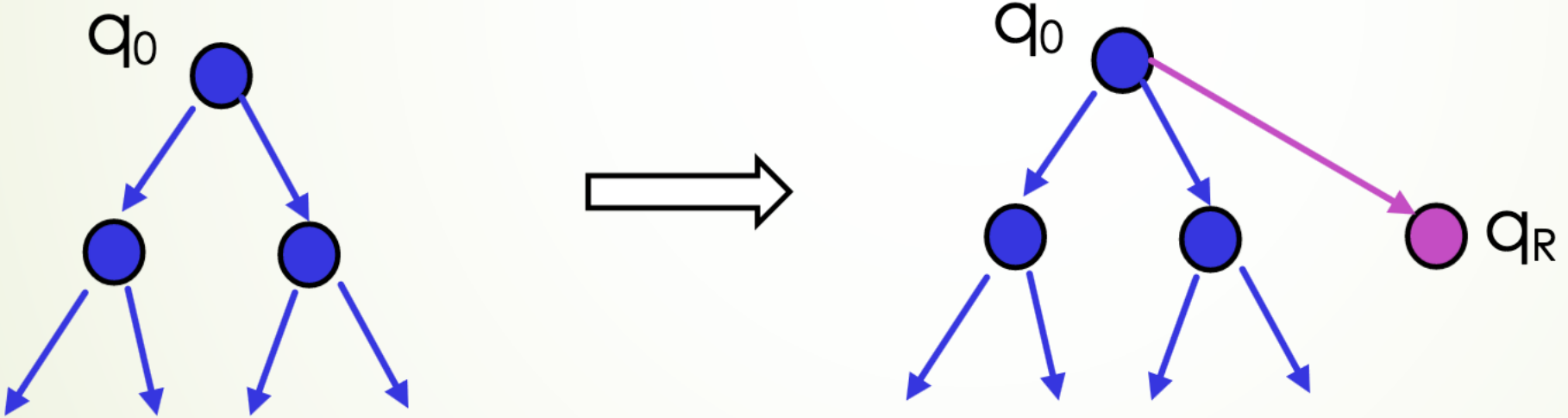

inizio della
computazione $U(p_{T_{PAL}}, x)$
Nastro 1 (N_1)
$\cdots$ q_0 $-$ q_A $-$ q_R $\langle$ q_0 , a , $\square$, q_a , D $\rangle$ $\cdots$
Nastro 2 (N_2)
$\cdots$ a b a b b b a b a a $\cdots$
Nastro 3 (N_3)
$\cdots$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\cdots$
Nastro 4 (N_4)
$\cdots$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\square$ $\cdots$

Input:	coppia di interi memorizzati nelle variabili n e m
1	$k \leftarrow 2$;
2	**if** $(n > m)$ **then**
3	$p \leftarrow n$;
4	**else**
5	$p \leftarrow m$;
6	**while** $(p \geq 2)$ **do begin**
7	$p \leftarrow p - k$;
8	**end**
9	**Output:** p

Tabella 3.1: Algoritmo che calcola se il massimo fra due interi è pari o dispari.

Input: stringa $x_1x_2\ldots x_n$ memorizzata nell'array N, con $N[i] = x_i$ per $i = 1,\ldots,n$, array Q, Σ, Q_1, S_1, S_2,Q_2, M descritti nel testo, q_0, q_A, q_R.

```
1   q ← q0;
2   t ← 1;
3   primaCella ← 1;
4   ultimaCella ← n;
5   while (q ≠ qA ∧ q ≠ qR) do begin
6       j ← 1;
7       trovata ← falso;
8       while (j ≤ k ∧ trovata = falso) do
9           if (q = Q1[j] ∧ N[t] = S1[j]) then trovata ← vero;
10          else j ← j + 1;
11      if (trovata = vero) then begin
12          N[t] ← S2[j];
13          q ← Q2[j];
14          t ← t + M[j];
15          if (t < primaCella) then begin
16              primaCella ← t;
17              N[t] ← □;
18          end
19          if (t > ultimaCella) then begin
20              ultimaCella ← t;
21              N[t] ← □;
22          end
23      end
24      else q ← qR;
25  end
26  Output: q
```

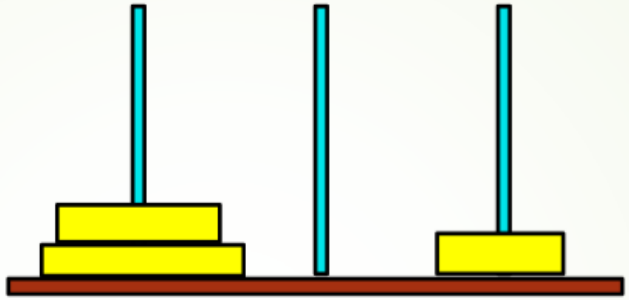


$NT(x)$

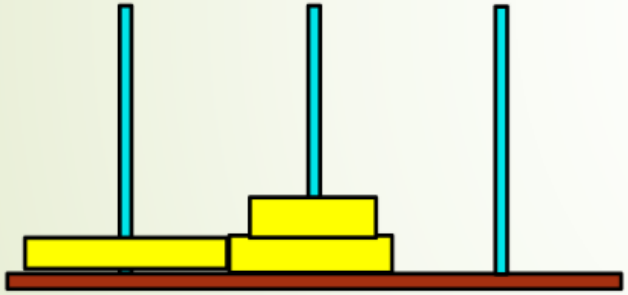
$NT_1(x)$

1)

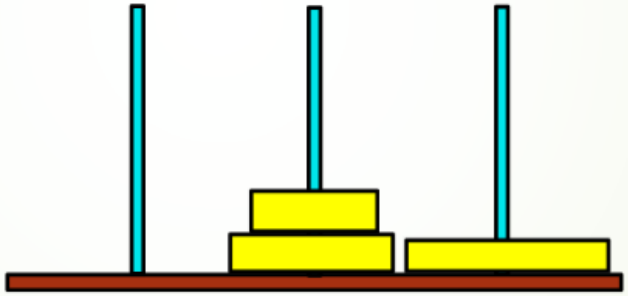
2)



3)

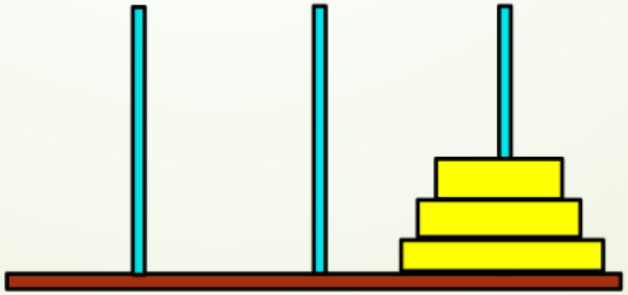


4)

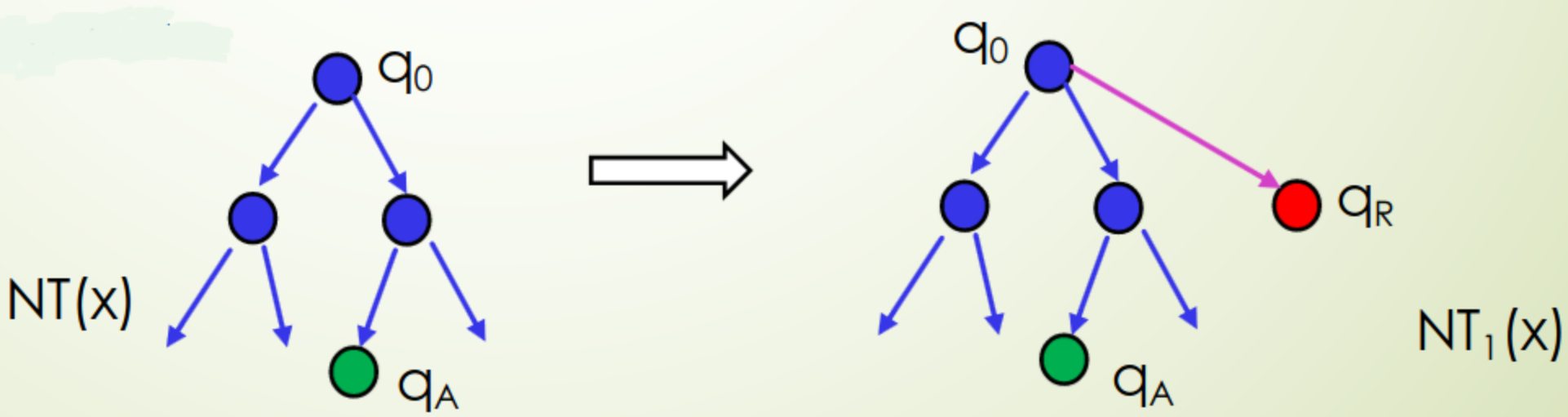


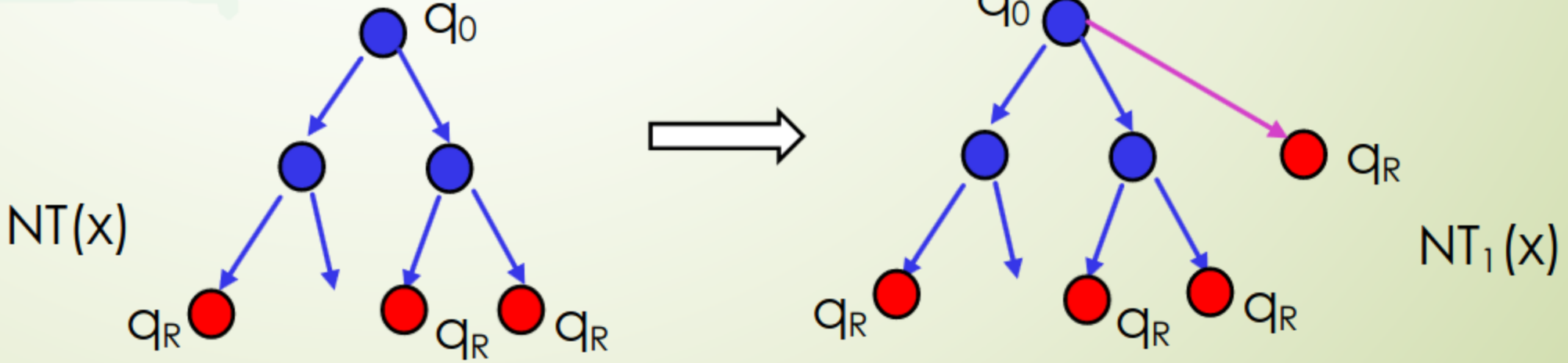
5)

6)



7)


q0
NT(x)
qA
q0
qR
qA
NT1(x)


NT(x)
q0
qR
qR
qR
q0
qR
qR
qR
qR
NT1(x)

x_1	x_2	x_3	f
vero	vero	vero	vero
vero	vero	falso	vero
vero	falso	vero	vero
vero	falso	falso	vero
falso	vero	vero	falso
falso	vero	falso	vero
falso	falso	vero	vero
falso	falso	falso	falso

Input: stringa $x_1 x_2 \ldots x_n$ memorizzata nell'array N, con $N[i] = x_i$ per $i = 1, \ldots, n$.
Costanti: l'insieme delle quintuple $P = \{\langle q_{1_1}, s_{1_1}, s_{1_2}, q_{1_2}, m_1\rangle, \langle q_{2_1}, s_{2_1}, s_{2_2}, q_{2_2}, m_2\rangle, \ldots, \langle q_{k_1}, s_{k_1}, s_{k_2}, q_{k_2}, m_k\rangle\}$ che descrivono la macchina di Turing NT.

```
1   q ← q_0;
2   t ← 1;
3   primaCella ← t;
4   ultimaCella ← n;
5   while (q ≠ q_A ∧ q ≠ q_R) do begin
6       Ψ ← {⟨q_{i_1}, s_{i_1}, s_{i_2}, q_{i_2}, m_i⟩ ∈ P : q_{i_1} = q ∧ s_{i_1} = N[t]};
7       if (Ψ ≠ ∅) then begin
8           scegli ⟨q_{i_1}, s_{i_1}, s_{i_2}, q_{i_2}, m_i⟩ ∈ Ψ;
9           N[t] ← s_{i_2};
10          q ← q_{i_2};
11          t ← t + m_i;
12      end
13      if (t < primaCella) then begin
14          primaCella ← t;
15          N[t] ← □;
16      end
17      if (t > ultimaCella) then begin
18          ultimaCella ← t;
19          N[t] ← □;
20      end
21  end
22  Output: q.
```

Input: un insieme di variabili booleane $X = \{x_1, x_2, \ldots, x_n\}$ e una funzione booleana f definita sulle variabili in X.

1 $i \leftarrow 1$;
2 **while** $(i \leq n)$ **do begin**
3 **scegli** $a(x_i) \in \{\texttt{vero}, \texttt{falso}\}$;
4 $i \leftarrow i + 1$;
5 **end**
6 $i \leftarrow 1$;
7 **while** $(i \leq n)$ **do begin**
8 sostituisci in f ogni occorrenza di x_i con $a(x_i)$, e ogni occorrenza di $\neg x_i$ con $\neg a(x_i)$
9 $i \leftarrow i + 1$;
10 **end**
11 **if** $(f(a(X)) = \texttt{vero})$ **then** $q \leftarrow q_A$;
12 **else** $q \leftarrow q_R$;
13 **Output:** q.

Input: un grafo non orientato $G = (V, E)$, con $V = \{v_1, v_2, \ldots, v_n\}$ e un intero $k \in \mathbb{N}$.

```
1   V' ← ∅;
2   i ← 1;
3   while (i ≤ n) do begin
4       scegli se inserire v_i in V';
5       i ← i + 1;
6   end
7   trovata ← vero;
8   i ← 1;
9   while (i ≤ n − 1) do begin
10      j ← i + 1;
11      while (j ≤ n) do begin
12          if (v_i ∈ V' ∧ v_j ∈ V' ∧ (v_i, v_j) ∉ E) then
13              trovata ← falso;
14          j ← j + 1;
15      end
16      i ← i + 1;
17  end
18  if (trovata = vero ∧ |V'| ≥ k) then q ← q_A;
19  else q ← q_R;
20  Output: q.
```

```
B ← Tf( |x| );          calcola la lunghezza della parola che deve scegliere
i ← 1;
while ( i ≤ B ) do begin
      scegli y[ i ] nell'insieme  {0,1};
      i ← i+1;
end
y ← y[1]y[2] … y[B];
```

$A \leftarrow T_g(|x|)$; calcola la lunghezza della computazione che deve simulare
$i \leftarrow 1$;
while ($i \leq A$) **do begin**
    simula l'esecuzione della i-esima istruzione eseguita da T(x,y);
    if (T è entrata in q_A) **then accetta** e termina;
    else $i \leftarrow i+1$;
end