Aritmetica al calcolatore

1. **rappresentazione dei numeri finiti**
2. **operazioni sui numeri finiti**
3. **condizionamento di un problema**
4. **stabilità di un algoritmo**
5. **propagazione degli errori**

I numeri e la rappresentazione posizionale

A causa della natura fisica a due stati degli elementi di base che costituiscono la memoria di un calcolatore, indipendentemente dalla tecnologia con cui essi sono costituiti, **si conviene** di rappresentare ogni elemento di base con una cifra binaria (**binary digit** o **bit**).

L'unico alfabeto compreso da una macchina ò l'**alfabeto binario**.

In una macchina o nei manuali capita di vedere sistemi di numerazione differenti da quello decimale.

Perciò conviene ricordare alcuni elementi sulla **rappresentazione posizionale dei numeri**.

Il concetto di numero

NUMERO

ENTITÀ ASTRATTA
“**sette**”
univocamente determinata

RAPPRESENTAZIONE
7, VII, $(111)_2$, ...
molteplice a seconda dei
criteri di rappresentazione adottati.

- RAPPRESENTAZIONE UNARIA: non conveniente perché non si riesce a gestire facilmente numeri grandi, non si riescono a trovare regole per il calcolo.
- RAPPRESENTAZIONE POSIZIONALE: dato un numero naturale $\beta > 1$ (detto base) e una lista di β simboli ordinati (corrispondenti alla rappresentazione dei primi β naturali), ogni numero naturale ò univocamente rappresentabile come combinazione di questi **simboli**.

Duplicità del valore di ogni simbolo

SIMBOLO

VALORE INTRINSECO
(compreso tra 0 e $\beta - 1$)
posizione occupata entro la lista ordinata dei β simboli

VALORE PESO (dato da β^j)
j ò la posizione occupata entro il numero;
equivalente alla potenza della base corrispondente,
contando le posizioni a partire da 0
da destra verso sinistra.

Un esempio (fantasioso) di sistema di numerazione

- Base ***tre***
- Lista ordinata di **tre** simboli: $\clubsuit$ (zero), $\diamondsuit$ (uno), $\heartsuit$ (due)

Un esempio di numero: $\diamondsuit\diamondsuit\clubsuit\heartsuit$

	$\diamondsuit$	$\diamondsuit$	$\clubsuit$	$\heartsuit$
valore intrinseco:	uno	uno	zero	due
valore peso:	3^3	3^2	3^1	3^0

La lista di simboli $\diamondsuit\diamondsuit\clubsuit\heartsuit$ rappresenta **trentotto** nel sistema scelto.

Il convenzionale sistema decimale

Per convenzione si adotta il sistema di numerazione decimale:

- base β uguale a **dieci**
- **dieci** simboli dati dalle *cifre arabiche* 0, 1, 2,. . . , 9.

Ogni numero naturale $N \in \mathbb{N}$ in notazione decimale si esprime come

$$N = d_n 10^n + d_{n-1} 10^{n-1} + \ldots + d_1 10^1 + d_0 10^0 = (d_n d_{n-1} \ldots d_0)_{10}$$

dove $0 \leqslant d_i \leqslant 9$ e $(d_n d_{n-1} ... d_0)_{10}$ si dice forma sintetica.

Ogni cifra d_i ha un valore intrinseco, pari a $\mathrm{ord}(d_i) \equiv d_i$, e un valore peso dato da 10^i.

Esempio: $327 = 3 \cdot 10^2 + 2 \cdot 10 + 7 \cdot 10^0$.

Un sistema di numerazione generale

In generale, in un sistema di numerazione in base $\beta > 1$, ogni numero naturale $N \in \mathbb{N}$ si rappresenta come:

$$N = (d_n d_{n-1} \dots d_0)_\beta = \mathrm{ord}(d_n)\beta^n + \mathrm{ord}(d_{n-1})\beta^{n-1} + \dots + \mathrm{ord}(d_1)\beta^1 + \mathrm{ord}(d_0)\beta^0$$

dove $\mathrm{ord}(d_i)$ ò il valore dell'*i*-esimo naturale.

Se si usano come simboli le cifre arabiche e se $\beta \leqslant 10$, allora $\mathrm{ord}(d_i) = d_i$.

Vale che:

$\mathrm{ord}(d_i)$	VALORE INTRINSECO
β^i	VALORE PESO

$(d_n d_{n-1} \dots d_0)_\beta$ ò la forma sintetica di N in base β.

BASE	SIMBOLI
2	0,1
8	0,1,2,3,4,5,6,7
16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Un esempio

trecentosettantadue:

$$(372)_{10} = 3 \cdot 10^2 + 7 \cdot 10^1 + 2 \cdot 10^0$$
$$(174)_{16} = 1 \cdot 16^2 + 7 \cdot 16^1 + 4 \cdot 16^0$$
$$(564)_8 = 5 \cdot 8^2 + 6 \cdot 8^1 + 4 \cdot 8^0$$
$$(101110100)_2 = 1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + \\ + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$$

duecentoottantasette:

$$(287)_{10} = (100011111)_2 = (10133)_4$$
$$(437)_8 = (11F)_{16} = (8V)_{32}.$$

Scelta della base: pro e contro

SCELTA DELLA BASE
- numero dei simboli
- lunghezza delle stringhe
- complessità dell'aritmetica

Se la base ò ***piccola***:

✓ pochi simboli;

† stringhe lunghe;

✓ aritmetica semplice (le tavole delle operazioni hanno pochi elementi!!).

Esercizio

Se ò noto il numero di cifre della rappresentazione di N in base β_2, posso stimare il numero di cifre della sua rappresentazione in base β_1?

$$N = (a_n a_{n-1} \dots a_0)_{\beta_1} \qquad = (b_m b_{m-1} \dots b_0)_{\beta_2}$$

$$\beta_1^n \leqslant N < \beta_1^{n+1} \qquad \beta_2^m \leqslant N < \beta_2^{m+1}$$

$$\log_{\beta_1}(\beta_1^n) \leqslant \log_{\beta_1}(N) < \log_{\beta_1}(\beta_1^{n+1}) \qquad \log_{\beta_2}(\beta_2^m) \leqslant \log_{\beta_2}(N) < \log_{\beta_2}(\beta_2^{m+1})$$

$$n \leqslant \log_{\beta_1}(N) < n+1 \qquad m \leqslant \log_{\beta_2}(N) < m+1$$

Per la formula del cambiamento di base nei logaritmi,

$$\log_{\beta_1}(N) = \log_{\beta_2}(N) \log_{\beta_1}(\beta_2)$$

approssimando n con $\log_{\beta_1}(N)$ e m con $\log_{\beta_2}(N)$, si ha che

$$\frac{n}{m} \simeq \log_{\beta_1}(\beta_2).$$

Se $\beta_1 = 2$ e $\beta_2 = 10$, $\log_{\beta_1}(\beta_2) = \log_2(10) \simeq 3.32$.

Pertanto, per rappresentare in base 2 un numero intero ci vogliono circa il triplo di cifre rispetto a quelle necessarie per rappresentarlo in base 10.

ESEMPIO. $(287)_{10} = (100011111)_2$

Operazioni aritmetiche nelle differenti basi

Valgono le stesse regole e proprietò formali dell'aritmetica decimale, ma si devono usare tavole diverse da quelle pitagoriche per addizione e moltiplicazione.
La somma può dare un riporto 1; il prodotto può dare un riporto compreso tra 1 e $\beta - 2$.

Base 2

$+$	0	1
0	0	1
1	1	10

$\cdot$	0	1
0	0	0
1	0	1

Base 8

$+$	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

$\cdot$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Caso binario

L'aritmetica binaria ò particolarmente semplice.

Somma: $(25)_{10} + (19)_{10} = (44)_{10}$.

		1	1	0	0	1	$+$
		1	0	0	1	1	$=$
Riporto	1			1	1		
	1	0	1	1	0	0	

Differenza: $(24)_{10} - (13)_{10} = (11)_{10}$.

1	1	0	0	0	$-$
0	1	1	0	1	$=$
0	1	0	1	1	

Caso binario

Prodotto: $(13)_{10} \cdot (14)_{10} = (182)_{10}$.

				1	1	0	1	$\cdot$
				1	1	1	0	$=$
				0	0	0	0	
			1	1	0	1		
		1	1	0	1			
	1	1	0	1				
1	0	1	1	0	1	1	0	

Il prodotto è riportato a somme e traslazioni di “stringhe” binarie.

Quoziente: $(28)_{10} : (9)_{10} = (3)_{10}$ con resto 1.

1	1	1	0	0	1	0	0	1
1	0	0	1		1	1		
0	1	0	1	0				
	1	0	0	1				
	0	0	0	1				

Il quoziente è riportato a differenze e traslazioni di “stringhe” binarie.

La base 2

La scelta della base 2 comporta la manipolazione di lunghe stringhe di numeri ma la complessità dell’aritmetica è bassa.

Le operazioni possono essere realizzate con semplici circuiti elettronici.

Esempio. La somma di due cifre *a* e *b* con riporto *c* fornisce il risultato *s* e il successivo riporto *c*. Il numero delle possibili combinazioni degli impulsi in entrata è basso (2^3).

a	b	*riporto*(c)	s	*riporto*(c)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

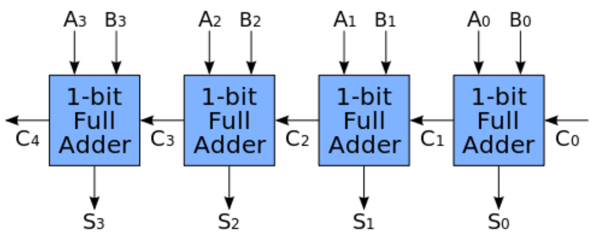

Figura: Esempio di 4 full-adder in cascata, per sommare due parole da 4 bit

Rappresentazione dei numeri reali

In una base $\beta \in \mathbb{N}$, $\beta > 1$ qualsiasi (anche diversa da 10):

- un numero reale minore di 1 può essere rappresentato come somma pesata di potenze negative della base:

$$\alpha = \pm\left(a_1\beta^{-1} + a_2\beta^{-2} + \ldots\right) = \pm 0.a_1a_2\ldots$$

Esempio: $(0.5)_{10} = \left(2^{-1}\right)_{10} = (0.1)_2 = 0 \cdot 2^0 + 1 \cdot 2^{-1}$
Esempio:

$$(0.75)_{10} = \frac{3}{4} = \frac{1}{2} + \frac{1}{4} = 1 \cdot 2^{-1} + 1 \cdot 2^{-2} \quad \Rightarrow \quad (0.75)_{10} = (0.11)_2$$

- La somma può essere anche infinita:

$$(0.1)_{10} = (0.0011001100\ldots)_2 = (0.\overline{0011})_2$$

- per un numero reale qualunque è sufficiente ricordare che ogni reale è la somma della sua parte intera e della sua parte frazionaria:

$$\begin{aligned}\alpha &= \pm\left(d_n\beta^n + d_{n-1}\beta^{n-1} + \ldots + d_1\beta + d_0\beta^0 + a_1\beta^{-1} + a_2\beta^{-2} + \ldots\right)\\ &= \pm\left(d_nd_{n-1}\ldots d_1d_0.a_1a_2\ldots\right)_\beta\end{aligned}$$

parte intera + parte frazionaria

La rappresentazione dei numeri reali

Teorema di rappresentazione dei numeri reali

Sia $\alpha \in \mathbb{R}$, $\alpha \neq 0$.
Fissato un intero $\beta > 1$, α si rappresenta in modo unico come:

$$\begin{aligned}\alpha &= \operatorname{sgn}(\alpha)\left(a_1\beta^{-1} + a_2\beta^{-2} + a_3\beta^{-3} + \ldots\right)\beta^p\\ &= \operatorname{sgn}(\alpha)\sum_{i=1}^{\infty}\left(a_i\beta^{-i}\right)\beta^p\\ &= \operatorname{sgn}(\alpha)m\beta^p\end{aligned}$$

dove

- $\operatorname{sgn}(\alpha) = \pm 1$ (a seconda che $\alpha > 0$ oppure $\alpha < 0$),
- $0 \leqslant a_i \leqslant \beta - 1$, con a_i interi e $a_1 \neq 0$
- p ò un intero;
- può esistere un indice k tale che $a_i = 0\ \forall i \geqslant k$ (*rappresentazione degli interi o dei razionali finiti*),
- ma non esiste un indice k tale che $a_i = \beta - 1\ \forall i \geqslant k$ (*unicitò*).

Conseguenze

- Il numero reale 0 si rappresenta con 0.
- Poiché $\beta > 1$, la serie $\sum_{i=1}^{\infty}(a_i\beta^{-i})$ è convergente:

$$\sum_{i=1}^{\infty}(a_i\beta^{-i}) < \sum_{i=1}^{\infty}(\beta-1)\beta^{-i} = (\beta-1)\sum_{i=1}^{\infty}\frac{1}{\beta^i} = (\beta-1)\frac{1/\beta}{1-1/\beta} = 1$$

(La serie $\sum_{i=1}^{\infty} a^i$ converge per $|a| < 1$ al valore $\frac{a}{1-a}$)

- $m = \sum_{i=1}^{\infty}(a_i\beta^{-i})$ si dice **mantissa** e vale che

$$\frac{1}{\beta} \leqslant m < 1$$

- β^p si dice **parte esponente**; p si dice **esponente o caratteristica**
- “ . ” si dice **punto radice**, $+$ o $-$ si dice **segno** del numero e può essere omesso se il numero ò positivo
- In **forma sintetica** ogni numero reale $\alpha \neq 0$ si rappresenta come

$$\alpha = \pm(.a_1a_2a_3\ldots)_\beta\beta^p$$

Se $a_1 \neq 0$, questa si dice **forma normalizzata**.

Esempio: $(0.75)_{10}\,10^0 = \frac{1}{2}+\frac{1}{4} = 1\cdot 2^{-1} + 1\cdot 2^{-2} = (0.11)_2\,2^0$

Forma mista e forma scientifica

Un numero reale $\alpha \neq 0$ si esprime in notazione posizionale in base $\beta > 1$ nel seguente modo:

- forma scientifica: $\alpha = \pm 0.a_1a_2\ldots\beta^p$; si dice normalizzata se $a_1 \neq 0$.

  In base 2 , in forma normalizzata, $a_1 = 1$ sempre.
- forma mista, senza la parte esponente

$$\alpha = \begin{cases} \pm 0.\underbrace{000\ldots0}_{p\text{ zeri}}\,a_1a_2\ldots & p \leqslant 0 \\ \pm a_1a_2\ldots a_p.a_{p+1}a_{p+2}\ldots & p > 0 \end{cases}$$

Si distingue:

- la **parte intera** $[\alpha]$ (se non nulla, corrisponde a un polinomio in β di grado $p-1$, a sinistra del punto radice: $a_1\beta^{p-1} + a_2\beta^{p-2} + \ldots + a_p\beta^0$)
- la **parte frazionaria** $\alpha - [\alpha]$ (corrispondente a una serie in $1/\beta$ senza termine corrispondente alla potenza nulla)

Se $p > 0$ e $\alpha - [\alpha] = 0$, il numero ò intero.

Esempi

$0.372 \cdot 10^3$	forma scientifica normalizzata
$0.0372 \cdot 10^4$	forma scientifica
$(372)_{10}$	forma mista
$3 \cdot 10^2 + 7 \cdot 10^1 + 2$	polinomio di grado 2 in $\beta = 10$
$(3.141592\ldots)_{10}$	forma mista
3	polinomio di grado 0 in $\beta = 10$
$1 \cdot \frac{1}{10} + 4 \cdot \frac{1}{10^2} + 1 \cdot \frac{1}{10^3} + 5 \cdot \frac{1}{10^4} + 9 \cdot \frac{1}{10^5} + \ldots$	serie
$0.3141592 \cdot 10^1$	forma scientifica normalizzata
$0.3243F\ldots \cdot 16^1$	forma scientifica normalizzata
$(3.243F\ldots)_{16}$	forma mista

Algoritmi di conversione di base

- Conversione di un intero positivo α da base 10 a base $\beta > 1$
- Conversione di un reale α non negativo e minore di 1 ($0 \leqslant \alpha < 1$) da base 10 a base $\beta > 1$
- Conversione di un reale α da base 10 a base $\beta > 1$
- Conversione di un reale α da base $\beta > 1$ a base 10
- Conversione di un reale α da base β_1 a base β_2

Conversione di un intero positivo α da base 10 a base $\beta > 1$

Le incognite del problema sono il numero $m+1$ di cifre della nuova rappresentazione e le cifre stesse $a_m, a_{m-1}, \ldots, a_0$.

Esempio. Conversione di $(123)_{10}$ a base 2.

$$
\begin{aligned}
(123)_{10} &= a_m 2^m + a_{m-1}2^{m-1} + \ldots + a_1 2 + a_0 = (a_m 2^m + a_{m-1}2^{m-1} + \ldots + a_1 2) + a_0 \\
&= (a_m 2^{m-1} + a_{m-1}2^{m-2} + \ldots + a_1)2 + a_0 \\
&= (61)_{10} \times (2)_{10} + 1 \Rightarrow a_0 = 1 \\
(61)_{10} &= (30)_{10} \times (2)_{10} + 1 \Rightarrow a_1 = 1 \\
(30)_{10} &= (15)_{10} \times (2)_{10} + 0 \Rightarrow a_2 = 0 \\
(15)_{10} &= (7)_{10} \times (2)_{10} + 1 \Rightarrow a_3 = 1 \\
(7)_{10} &= (3)_{10} \times (2)_{10} + 1 \Rightarrow a_4 = 1 \\
(3)_{10} &= (1)_{10} \times (2)_{10} + 1 \Rightarrow a_5 = 1 \\
(1)_{10} &= (0)_{10} \times (2)_{10} + 1 \Rightarrow a_6 = 1
\end{aligned}
$$

$m+1 = 6+1$ e il numero convertito vale $(1111011)_2$.

Conversione di un intero positivo α da base 10 a base $\beta > 1$

A partire dall'esempio spieghiamo il **METODO DELLE DIVISIONI SUCCESSIVE**:

$$
\begin{aligned}
\alpha &= (a_m a_{m-1} \ldots a_0)_\beta = a_m\beta^m + a_{m-1}\beta^{m-1} + \ldots + a_1\beta + a_0 \\
&= (a_m\beta^{m-1} + a_{m-1}\beta^{m-2} + \ldots + a_1)\beta + a_0 = \gamma_1\beta + a_0
\end{aligned}
$$

a_0, ossia la cifra meno significativa della rappresentazione cercata, è il resto della divisione intera di α per β.

$$
\begin{aligned}
\gamma_1 &= (a_m\beta^{m-2} + a_{m-1}\beta^{m-3} + \ldots + a_3\beta + a_2)\beta + a_1 = \gamma_2\beta + a_1 \\
\gamma_2 &= (a_m\beta^{m-3} + a_{m-1}\beta^{m-4} + \ldots + a_4\beta + a_3)\beta + a_2 = \gamma_3\beta + a_2 \\
&\vdots \\
\gamma_{m-1} &= a_m\beta + a_{m-1} = \gamma_m\beta + a_{m-1} \\
\gamma_m &= 0\beta + a_m
\end{aligned}
$$

$m+1$ è il numero delle divisioni successive eseguite fino ad avere un quoziente 0. Dopo aver eseguito $m+1$ divisioni, i resti in ordine inverso (rappresentati con i simboli della nuova base) forniscono la rappresentazione del numero.

Metodo delle divisioni successive

Per convertire un intero positivo da base 10 a base $\beta \in \mathbb{N}\backslash\{1\}$, si opera ripetutamente la divisione intera fra il numero da convertire e la base β, conservando i resti nell'ordine in cui vengono generati. Ci si arresta quanto il quoziente della divisione intera diventa zero. Le cifre del valore convertito nella base β si ottengono leggendo i resti in ordine inverso a quello in cui sono stati generati.

Esempio di conversione a base binaria: $(2015)_{10} = (11111011111)_2$

$$
\begin{array}{rll}
2015 : 2 = & 1007 & \text{resto } 1 \\
1007 : 2 = & 503 & \text{resto } 1 \\
503 : 2 = & 251 & \text{resto } 1 \\
251 : 2 = & 125 & \text{resto } 1 \\
125 : 2 = & 62 & \text{resto } 1 \\
62 : 2 = & 31 & \text{resto } 0 \\
31 : 2 = & 15 & \text{resto } 1 \\
15 : 2 = & 7 & \text{resto } 1 \\
7 : 2 = & 3 & \text{resto } 1 \\
3 : 2 = & 1 & \text{resto } 1 \\
1 : 2 = & 0 & \text{resto } 1
\end{array}
$$

Algoritmo delle divisioni successive

Altri esempi di conversione dalla base 10

a base 8:

$$
\begin{array}{rll}
2015 : 8 = & 251 & \text{resto } 7 \\
251 : 8 = & 31 & \text{resto } 3 \\
31 : 8 = & 3 & \text{resto } 7 \\
3 : 8 = & 0 & \text{resto } 3
\end{array}
\quad \Rightarrow \quad (2015)_{10} = (3737)_8
$$

a base 16:

$$
\begin{array}{rll}
2015 : 16 = & 125 & \text{resto } (15)_{10} = (\mathrm{F})_{16} \\
125 : 16 = & 7 & \text{resto } (13)_{10} = (\mathrm{D})_{16} \\
7 : 16 = & 0 & \text{resto } 7
\end{array}
\quad \Rightarrow \quad (2015)_{10} = (7\mathrm{DF})_{16}
$$

a base 23: simboli 0,...,9,A,B,...,L,M

$$
\begin{array}{rll}
2015 : 23 = & 87 & \text{resto } (14)_{10} = (\mathrm{E})_{23} \\
87 : 23 = & 3 & \text{resto } (18)_{10} = (\mathrm{I})_{23} \\
3 : 23 = & 0 & \text{resto } 3
\end{array}
\quad \Rightarrow \quad (2015)_{10} = (3\mathrm{IE})_{23}
$$

Algoritmo delle divisioni successive in pseudocodice

In pseudo-codice:

Dato un intero α e l'insieme di simboli $d_0, d_1, \ldots, d_{\beta-1}$:

$q \leftarrow \alpha$
$c \leftarrow$ '' *# stringa vuota*
while $(q \neq 0)$
  $r \leftarrow \mathrm{rem}(q, \beta)$ *# resto della divisione intera*
  $q \leftarrow \mathrm{fix}(q/\beta)$ *# quoziente della divisione intera*
  $c \leftarrow \mathrm{cat}(d_r, c)$ *# concatenazione di stringhe*
end while
return c

Nota. Il simbolo "#" rappresenta l'inizio di un commento al codice. In Matlab tale funzione è svolta dal simbolo "%".

In **Matlab**: detti `BETA` la base a cui convertire, `ALPHA` il numero intero non negativo da convertire e `d` il vettore dei simboli per le cifre nella base `BETA` (cioè `d(0)`, `d(1)`, `...`, `d(BETA-1)`):

```
q = ALPHA;
s = '';
while (q ~= 0)
  r = q - fix(q/BETA)*BETA; % resto della divisione intera
  q = fix(q/BETA);          % divisione intera
  s = strcat(d(r+1),s);     % concatenazione di stringhe
end
disp(s);
```

Nota. In Matlab gli indici dei vettori partono da 1, non da zero, quindi occore usare `d(r+1)` invece di `d(r)` per ottenere l'r-esima cifra in base `BETA`. Il simbolo "~" rappresenta la negazione logica, dunque "~=" significa "diverso da".

Un metodo alternativo

Esiste un modo più efficiente per eseguire la conversione. Esso consiste dei seguenti passi.
Sia α il numero da convertire in base β e s la locazione che conterrà la sua conversione:

- $s = 0$ inizialmente;
- determinare la più grande potenza β^j della base β che non supera il numero α, contando quante volte questa potenza sta in α; se i è il numero di volte, sommare $i\beta^j$ a s ($s = s + i000\ldots000$ (j zeri));
- togliere da α il numero $i\beta^j$ e ripetere fino a che $\alpha = 0$.

L'algoritmo è particolarmente semplice ed efficiente in base 2 ($i = 1$ sempre).

Per esempio, se $\alpha = 1972$, la potenza di 2 piò grande che non supera il numero vale $1024 = 2^{10}$. Quindi, $s \leftarrow 10000000000$ e $\alpha \leftarrow 1972 - 1024 = 948$. Ripetendo:

$$
\begin{array}{llll}
 & 512 < 948; & \alpha \leftarrow 436, & s \leftarrow 11000000000 \\
 & 256 < 436; & \alpha \leftarrow 180, & s \leftarrow 11100000000 \\
 & 128 < 180; & \alpha \leftarrow 52, & s \leftarrow 11110000000 \\
 & 32 < 52; & \alpha \leftarrow 20, & s \leftarrow 11110100000 \\
 & 16 < 20; & \alpha \leftarrow 4, & s \leftarrow 11110110000 \\
\text{e poiché la conversione di 4 vale 100,} & & & s \leftarrow 11110110100.
\end{array}
$$

Conversione di un reale positivo $\alpha < 1$ da base 10 a base $\beta > 1$

Anche nella conversione di un valore non intero, le incognite del problema sono il numero di cifre della nuova rappresentazione e le cifre stesse $a_1, a_2, \ldots$.

Non si sa a priori se il numero ha rappresentazione finita oppure no, poiché **non è detto che se un numero ha rappresentazione finita in base 10, altrettanto accada in base β.**

Si dimostra che:

> un numero $\alpha > 0$ ha rappresentazione finita in base $\beta \Leftrightarrow$ esistono due interi positivi m ed n tali che $\alpha = \dfrac{m}{\beta^n}$.

Altrimenti il numero nella nuova base ha rappresentazione non finita.

Conversione di un reale positivo $\alpha < 1$ da base 10 a base $\beta > 1$

Esempio. Conversione di 0.3 da base 10 a base 2.

$$
\begin{aligned}
0.3 &= (0.a_1 a_2 a_3 \ldots)_2 = a_1\, 2^{-1} + a_2\, 2^{-2} + \ldots \\
0.3 \cdot 2 &= (a_1\, 2^{-1} + a_2\, 2^{-2} + \ldots) \cdot 2 = a_1 + (a_2\, 2^{-1} + \ldots) \\
&= 0.6 = 0 + 0.6 \Rightarrow a_1 = 0 \\
0.6 \cdot 2 &= 1.2 = 1 + 0.2 \Rightarrow a_2 = 1 \\
0.2 \cdot 2 &= 0.4 = 0 + 0.4 \Rightarrow a_3 = 0 \\
0.4 \cdot 2 &= 0.8 = 0 + 0.8 \Rightarrow a_4 = 0 \\
0.8 \cdot 2 &= 1.6 = 1 + 0.6 \Rightarrow a_5 = 1 \\
0.6 \cdot 2 &= 1.2 = 1 + 0.2 \Rightarrow a_6 = 1
\end{aligned}
$$

Da a_5 in poi il processo si ripete (è un numero periodico). Il numero convertito vale dunque $(0.0\overline{1001})_2$.

Conversione di un reale positivo $\alpha < 1$ da base 10 a base $\beta > 1$

Deriviamo dall'esempio il **METODO DELLE MOLTIPLICAZIONI SUCCESSIVE**:

$$\begin{aligned}\alpha &= (0.a_1 a_2 a_3 \ldots)_\beta \\ &= a_1\beta^{-1} + a_2\beta^{-2} + a_3\beta^{-3} + \ldots\end{aligned}$$

- moltiplicazione di α per la base:

  $$\alpha\beta = a_1 + a_2\beta^{-1} + a_3\beta^{-2} + a_4\beta^{-3}\ldots = a_1 + \eta_1$$

  dove a_1 è la parte intera del risultato e η_1 la parte frazionaria:

  $$\eta_1 = a_2\beta^{-1} + a_3\beta^{-2} + \ldots$$

- ripetere:

  $$\eta_1\beta = a_2 + a_3\beta^{-1} + a_4\beta^{-2} + \ldots = a_2 + \eta_2$$

- ripetere:

  $$\eta_2\beta = a_3 + a_4\beta^{-1} + a_5\beta^{-2} + \ldots = a_3 + \eta_3$$

Ci si arresta perché la parte frazionaria diventa nulla, oppure perché si è raggiunto un numero sufficiente di cifre.

Algoritmo delle moltiplicazioni successive

Per convertire un numero reale $\alpha < 1$ da base 10 a base $\beta \in \mathbb{N}\backslash\{1\}$, si moltiplica il numero α per la nuova base β e si conserva la parte intera del risultato, poi si effettua ripetutamente la stessa operazione sulla parte frazionaria del risultato, finché questa diventa nulla. Le cifre del numero convertito nella nuova base β si ottengono leggendo, dopo il punto radice, le parti intere delle moltiplicazioni, nello stesso ordine in cui sono state generate.

Esempio di conversione a base binaria: $(0.75)_{10} = (0.11)_2$

$0.75 \cdot 2 = 1.5$ parte intera 1
parte frazionaria 0.5

$0.5 \cdot 2 = 1.0$ parte intera 1
parte frazionaria 0

Il primo fattore della seconda riga è la parte frazionaria del risultato della moltiplicazione precedente: $0.5 = 1.5 - 1$.

Algoritmo delle moltiplicazioni successive

Altri esempi di conversione dalla base 10: $\alpha = 0.1$

- a base 2:

$$\begin{array}{lll} 0.1 \cdot 2 = 0.2 & \text{parte intera } 0, & \text{parte fraz. } 0.2 \\ 0.2 \cdot 2 = 0.4 & \text{parte intera } 0, & \text{parte fraz. } 0.4 \\ 0.4 \cdot 2 = 0.8 & \text{parte intera } 0, & \text{parte fraz. } 0.8 \\ 0.8 \cdot 2 = 1.6 & \text{parte intera } 1, & \text{parte fraz. } 0.6 \\ 0.6 \cdot 2 = 1.2 & \text{parte intera } 1, & \text{parte fraz. } 0.2 \\ 0.2 \cdot 2 = 0.4 & \text{parte intera } 0, & \text{parte fraz. } 0.4 \\ 0.4 \cdot 2 = 0.8 & \text{parte intera } 0, & \text{parte fraz. } 0.8 \\ \quad\vdots & & \end{array} \quad \Rightarrow \quad (0.1)_{10} = (0.0\overline{0011})_2$$

- a base 5:

$$\begin{array}{lll} 0.1 \cdot 5 = 0.5 & \text{parte intera } 0, & \text{parte fraz. } 0.5 \\ 0.5 \cdot 5 = 2.5 & \text{parte intera } 2, & \text{parte fraz. } 0.5 \\ 0.5 \cdot 5 = 2.5 & \text{parte intera } 2, & \text{parte fraz. } 0.5 \\ \quad\vdots & & \end{array} \quad \Rightarrow \quad (0.1)_{10} = (0.0\overline{2})_5$$

Osservazioni:

- non è detto che la rappresentazione sia finita in tutte le basi
- arresto: quando la parte frazionaria è nulla oppure quando occorre fissare un numero massimo di cifre della rappresentazione

Algoritmo delle moltiplicazioni successive in pseudocodice

In pseudo-codice:

Dato un numero reale $\alpha < 1$, l'insieme di simboli $d_0, d_1, \ldots, d_{\beta-1}$ e un numero massimo $N_{\max}$ di cifre:

$r \leftarrow \alpha$
$c \leftarrow$ '0.' *# stringa*
$i \leftarrow 0$ *# contatore*
while $(r \neq 0$ **and** $i < N_{\max})$
  $p \leftarrow \mathrm{fix}(r\beta)$ *# parte intera della moltiplicazione*
  $r \leftarrow r\beta - p$ *# parte frazionaria della moltiplicazione*
  $c \leftarrow \mathrm{cat}(c, d_p)$ *# concatenazione di stringhe*
  $i \leftarrow i + 1$ *# incremento del contatore*
end while
return c

In **Matlab**:
detti `BETA` la base a cui convertire, `ALPHA` il numero frazionario da convertire, `N` il numero massimo di cifre frazionarie e `d` il vettore dei simboli per le cifre nella base `BETA` (cioè `d(0), d(1), ..., d(BETA-1)`):

```
p = ALPHA;
s = '0.';
i = 0;
while ((p ~= 0) & (i < N))
  r = fix(p*BETA);      % parte_intera
  p = p*BETA - r;       % parte_frazionaria
  s = strcat(s,d(r+1)); % concatenazione di stringhe
  i = i+1;
end
disp(s)
```

Conversione di un reale α da base 10 a base $\beta > 1$

Per convertire un numero reale α qualsiasi dalla base 10 ad una nuova base $\beta \in \mathbb{N}\backslash\{1\}$ occorre:

1. determinare il segno s di α
2. considerare il valore assoluto $|\alpha|$
3. convertire la parte intera con l'algoritmo delle divisioni successive
4. convertire la parte frazionaria con l'algoritmo delle moltiplicazioni successive
5. scrivere il segno, la conversione della parte intera, il punto radice, la conversione della parte frazionaria.

Esempio: convertire in base 2 il numero $\alpha = (-36.527)_{10}$

1. segno $s = \,'-'$; $|\alpha| = 36.527$;
2. $\left[|\alpha|\right] = 36$: $(36)_{10} = (100100)_2$
3. $|\alpha| - \left[|\alpha|\right] = 0.527$: $(0.527)_{10} = (0.\underbrace{100001\ldots}_{N_{\max}\text{ cifre}})_2$
4. $\alpha = (100100.100001\ldots)_2$

Conversione di un reale da base $\beta > 1$ a base 10

Per convertire da base β a base 10 il numero reale α, ci sono due metodi alternativi:

1. si possono usare gli algoritmi delle divisioni e delle moltiplicazioni successive, **purché si lavori con aritmetica in base β.** Le cifre ottenute si convertono ai simboli di base 10.

   Esempio. $(123.41)_5$

$$(123)_5 : (20)_5 = (3)_5 \quad \text{resto } (13)_5 = (8)_{10}$$
$$(3)_5 : (20)_5 = (0)_5 \quad \text{resto } (3)_5 = (3)_{10}$$

$$(0.41)_5 \cdot (20)_5 = (13.20)_5 \quad \text{parte intera } (13)_5 = (8)_{10}$$
$$(0.20)_5 \cdot (20)_5 = (4.00)_5 \quad \text{parte intera } (4)_5 = (4)_{10}$$

   Conversione: $(38.84)_{10}$. Oppure...

Conversione di un reale da base $\beta > 1$ a base 10

2. si può sfruttare la **rappresentazione posizionale** ($p > 0$):

$$\alpha = \pm(a_1 a_2 \ldots a_p . a_{p+1} a_{p+2} \ldots a_q)_\beta$$
$$= \pm\Bigg(a_1\beta^{p-1} + a_2\beta^{p-2} + \ldots + a_p\beta^0$$
$$+ a_{p+1}\left(\frac{1}{\beta}\right)^1 + a_{p+2}\left(\frac{1}{\beta}\right)^2 + \ldots + a_q\left(\frac{1}{\beta}\right)^{-p+q}\Bigg)$$

Si tratta di calcolare due **polinomi**:
$f(x) = a_1 x^{p-1} + a_2 x^{p-2} + \ldots + a_p$ in $x = \beta$,
$g(x) = a_q x^{-p+q} + a_{q-1} x^{-p+q-1} + \ldots + a_{p+1} x$ in $x = 1/\beta$

$$(\alpha)_{10} = \pm(f(\beta) + g(1/\beta))$$

Esempio. $(123.41)_5 = (1\cdot 5^2 + 2\cdot 5^1 + 3\cdot 5^0) + \left(4\cdot\frac{1}{5^1} + 1\cdot\frac{1}{5^2}\right)$

Dunque occorre calcolare i due polinomi:

$$f(x) = 1x^2 + 2x + 3 \quad \text{in} \quad x = 5 \qquad \text{e} \qquad g(x) = 1x^2 + 4x + 0 \quad \text{in} \quad x = \frac{1}{5}$$

Occorre un algoritmo conveniente per fare il calcolo di un polinomio a coefficienti reali in corrispondenza di un certo valore.

Conversione di un reale α da base β_1 a base β_2

- Si può eseguire la conversione da base β_1 a base β_2 usando l'algoritmo delle divisioni successive e/o delle moltiplicazioni successive con **aritmetica in base** β_1, convertendo le cifre ottenute ai simboli della base β_2. β_2 va espresso in base β_1.

  Esempio. $\alpha = (111001000)_2$. Si converte da base $\beta_1 = 2$ a base $\beta_2 = 7 = (111)_2$.

  $$\begin{array}{rl} 111001000 : 111 = 1000001 & \text{resto } (1)_2 = (1)_7 \\ 1000001 : 111 = 1001 & \text{resto } (10)_2 = (2)_7 \\ 1001 : 111 = 1 & \text{resto } (10)_2 = (2)_7 \\ 1 : 111 = 0 & \text{resto } (1)_2 = (1)_7 \end{array}$$

  $\alpha = (1221)_7$.
- **Si passa attraverso la base 10**: si converte da base β_1 a base 10 (usando la rappresentazione posizionale) e da base 10 a base β_2 (con gli algoritmi delle divisioni e delle moltiplicazioni successive).

  Esempio. $\alpha = (1221)_7$. Conversione da base $\beta_1 = 7$ a base $\beta_2 = 2$.

  $$\alpha = 1\cdot 7^3 + 2\cdot 7^2 + 2\cdot 7^1 + 1\cdot 7^0 = (456)_{10}$$
  $$(456)_{10} = (111001000)_2 \quad \text{DIVISIONI SUCCESSIVE}$$

Conversione di un reale da base β_1 a base β_2 – Casi particolari

- $\beta_1 \to \beta_2 = \beta_1^k$: per rappresentare un numero α reale nella base β_2, si staccano gruppi di k cifre a partire dal punto radice verso destra e verso sinistra, completando eventualmente il primo e l'ultimo gruppo con zeri; ogni gruppo è convertito a un simbolo della base β_2.

  Esempio. $\beta_1 = 2; \beta_2 = 8 = 2^3$ $(k = 3)$.

$$\alpha = (-1101110.01)_2 = (-156.2)_8$$

$$\underbrace{001}_{1}\ \underbrace{101}_{5}\ \underbrace{110}_{6}\ .\ \underbrace{010}_{2}$$

$\beta_1 = 2; \beta_2 = 16 = 2^4 \quad (k = 4)$.

$$\alpha = (-1101110.01)_2 = (-6E.4)_{16}$$

$$\underbrace{0110}_{6}\ \underbrace{1110}_{E}\ .\ \underbrace{0100}_{4}$$

Conversione di un reale da base β_1 a base β_2 – Casi particolari

- $\beta_1 = \beta_2^k \to \beta_2$: viceversa, si espande ogni simbolo della rappresentazione di α in base β_1 sostituendolo con un gruppo di k cifre che sono la conversione del simbolo nella base β_2.

  Esempio. $\beta_1 = 9 = 3^2; \beta_2 = 3 \quad (k = 2)$.

$$\alpha = (37.47)_9 = (1021.1121)_3$$

$$\underbrace{3}_{10}\ \underbrace{7}_{21}\ .\ \underbrace{4}_{11}\ \underbrace{7}_{21}$$

Per facilitare la conversione, si usa la seguente tabella di conversione dei simboli

$$\begin{aligned}
(00)_3 &\to 0_9\\
(01)_3 &\to 1_9\\
(02)_3 &\to 2_9\\
(10)_3 &\to 3_9\\
(11)_3 &\to 4_9\\
(12)_3 &\to 5_9\\
(20)_3 &\to 6_9\\
(21)_3 &\to 7_9\\
(22)_3 &\to 8_9
\end{aligned}$$

Valutazione di un polinomio reale di grado n in $x = \alpha$.

$$p_n(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0 \qquad a_i \in \mathbb{R},\ a_n \neq 0$$

ALGORITMO 1.

$p \leftarrow 1$
$s \leftarrow a_0$
for $i \leftarrow 1$ **to** n
  $p \leftarrow p \cdot \alpha$
  $s \leftarrow p \cdot a_i + s$
end
output: s

La COMPLESSITÀ COMPUTAZIONALE dell'algoritmo (ossia il numero totale di operazioni aritmetiche che devono essere fatte) è $2n$ moltiplicazioni e n addizioni.

Esempio. $p_4(x) = 2x^4 - 3x^2 + 3x - 4 = 2x^4 + 0x^3 - 3x^2 + 3x - 4 \qquad \alpha = -2$
$\boldsymbol{a} = (2, 0, -3, 3, -4)$

inizio :	$p \leftarrow 1;$	$s \leftarrow a_0 = -4;$
step $i = 1$:	$p \leftarrow p \cdot (-2) = -2;$	$s \leftarrow p \cdot a_1 + s = (-2) \cdot 3 + (-4) = -10$
step $i = 2$:	$p \leftarrow p \cdot (-2) = 4;$	$s \leftarrow p \cdot a_2 + s = 4 \cdot (-3) + (-10) = -22$
step $i = 3$:	$p \leftarrow p \cdot (-2) = -8;$	$s \leftarrow p \cdot a_3 + s = (-8) \cdot 0 + (-22) = -22$
step $i = 4$:	$p \leftarrow p \cdot (-2) = 16;$	$s \leftarrow p \cdot a_4 + s = 16 \cdot 2 + (-22) = 10$

totale: 8 moltiplicazioni e 4 addizioni

Schema di Ruffini-Horner

$$p_n(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0 \qquad a_i \in \mathbb{R},\ a_n \neq 0$$

ALGORITMO 2. Si basa sulla seguente riscrittura del polinomio:

$$p_n(x) = \Big(\ldots\Big(\big((a_n x + a_{n-1})x + a_{n-2}\big)x + a_{n-3}\Big)x + \ldots + a_1\Big)x + a_0$$

$s \leftarrow a_n$
for $i \leftarrow n-1$ **to** 0 **step** -1
  $s \leftarrow s \cdot \alpha + a_i$
end
output: s

La COMPLESSITÀ COMPUTAZIONALE è pari a n moltiplicazioni e n addizioni, con un risparmio di n moltiplicazioni.
L'algoritmo prende il nome di **schema di RUFFINI-HORNER**.

Esempio. $p_4(x) = 2x^4 - 3x^2 + 3x - 4$ in $\alpha = -2$. Con $p_4(x) = \Big(\big((2x + 0)x - 3\big)x + 3\Big)x - 4$

inizio :	$s \leftarrow 2$
step $i = 1$:	$s \leftarrow s \cdot (-2) + 0 = -4$
step $i = 2$:	$s \leftarrow s \cdot (-2) - 3 = 5$
step $i = 3$:	$s \leftarrow s \cdot (-2) + 3 = -7$
step $i = 4$:	$s \leftarrow s \cdot (-2) - 4 = 10$

totale: 4 moltiplicazioni e 4 addizioni

Schema di Ruffini-Horner

Lo schema di Ruffini-Horner si può riscrivere in modo da tener conto dei risultati intermedi:

$b_n \leftarrow a_n$
for $i \leftarrow n-1$ **to** 0 **step** -1
    $b_i \leftarrow b_{i+1} \cdot \alpha + a_i$
end
output b_0

In questo caso, i valori b_i, $i = 1, \dots, n$, sono i coefficienti del **polinomio quoziente** $q(x)$ della divisione di $p_n(x)$ per $(x - \alpha)$ e b_0 ò il resto della divisione:

$$p_n(x) = (x - \alpha)(b_n x^{n-1} + b_{n-1} x^{n-2} + \dots + b_1) + b_0$$

Osservazione. b_0 è una costante che rappresenta il resto della divisione di $p_n(x)$ per $x - \alpha$. Infatti, in base al **Teorema di Ruffini**, il valore di un polinomio in α è uguale al resto della divisione di $p_n(x)$ per $x - \alpha$.
La **regola di Ruffini** (divisione sintetica), che calcola i coefficienti del polinomio quoziente e il resto di tale divisione, **coincide con l'algoritmo di Horner**.

Schema di Ruffini-Horner: esempio

$$p_n(x) = (x - \alpha)(b_n x^{n-1} + b_{n-1} x^{n-2} + \dots + b_1) + b_0$$

Esempio:

$$2x^4 - 3x^2 + 3x - 4 = \big(x - (-2)\big)(2x^3 - 4x^2 + 5x - 7) + 10$$

Verifica. Eseguiamo il prodotto di $(x - \alpha)$ per $q(x)$:

$$\begin{aligned}(x - \alpha)&(b_n x^{n-1} + b_{n-1} x^{n-2} + \dots + b_1) + b_0 \\ &= b_n x^n + (b_{n-1} - \alpha b_n) x^{n-1} + (b_{n-2} - \alpha b_{n-1}) x^{n-2} + \dots \\ &\quad + (b_1 - \alpha b_2) x + b_0 - \alpha b_1\end{aligned}$$

Perché ci sia uguaglianza con $p_n(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$ deve essere:

$$\begin{aligned} a_n &= b_n & \text{coefficiente di } x^n &\Leftrightarrow b_n = a_n \\ a_{n-1} &= b_{n-1} - \alpha b_n & \text{coefficiente di } x^{n-1} &\Leftrightarrow b_{n-1} = \alpha b_n + a_{n-1} \\ &\vdots & &\vdots \\ a_0 &= b_0 - \alpha b_1 & \text{coefficiente di } x^0 = 1 &\Leftrightarrow b_0 = \alpha b_1 + a_0 \end{aligned}$$

che vale per come è costruito l'algoritmo.

Schema di Ruffini-Horner

	a_n	a_{n-1}	a_{n-2}	$\dots$	a_1	a_0
α		$b_n\alpha$	$b_{n-1}\alpha$	$\dots$	$b_2\alpha$	$b_1\alpha$
	b_n	b_{n-1}	b_{n-2}	$\dots$	b_1	b_0

Esempio. Calcolo in $x = 2$ del polinomio

$$p_4(x) = x^4 - 5x^3 + 3x^2 - 2x + 1 = \Big(\big((x-5)x+3\big)x-2\Big)x+1$$

	1	-5	3	-2	1
2		2	-6	-6	-16
	1	-3	-3	-8	-15

$$p_4(2) = -15$$
$$p_4(x) = (x-2)(x^3 - 3x^2 - 3x - 8) - 15$$

Applicazione: calcolo di un polinomio e delle sue derivate in $x = \alpha$

Lo schema di Ruffini-Horner permette di valutare un polinomio in $x = \alpha$, calcolando il resto ($r_1 = p_n(\alpha)$) della divisione di $p_n(x)$ per $(x - \alpha)$ e i coefficienti del quoziente di tale divisione ($q_1(x)$) (prima volta):

$$p_n(x) = (x-\alpha)q_1(x) + r_1 \quad \Rightarrow \quad p_n(\alpha) = r_1$$

Poiché vale che:

$$p_n'(x) = q_1(x) + (x-\alpha)q_1'(x)$$

segue che

$$p_n'(\alpha) = q_1(\alpha)$$

Pertanto applicando lo schema di Ruffini-Horner (seconda volta) a $q_1(x)$ si ottengono $q_2(x)$ ed $r_2 = q_1(\alpha) = p_n'(\alpha)$:

$$q_1(x) = (x-\alpha)q_2(x) + r_2 \quad \Rightarrow \quad q_1(\alpha) = r_2 = p_n'(\alpha)$$

e inoltre $q_1'(x) = q_2(x) + (x-\alpha)q_2'(x)$.

Applicazione: calcolo di un polinomio e delle sue derivate in $x = \alpha$

Da $p_n'(x) = q_1(x) + (x - \alpha)q_1'(x)$, vale che:

$$p_n''(x) = 2q_1'(x) + (x - \alpha)q_1''(x)$$

Pertanto $p_n''(\alpha) = 2q_1'(\alpha) = 2q_2(\alpha)$.
Applicando lo schema di Horner a $q_2(x)$ (terza volta), si ottengono $q_3(x)$ e il resto $r_3 = q_2(\alpha) = q_1'(\alpha) = p_n''(\alpha)/2$:

$$q_2(x) = (x - \alpha)q_3(x) + r_3$$

e $q_2'(x) = q_3(x) + (x - \alpha)q_3'(x)$.
Ancora

$$p_n'''(x) = 3q_1''(x) + (x - \alpha)q_1'''(x)$$

Pertanto $p_n'''(\alpha) = 3q_1''(\alpha) = 3 \cdot 2q_2'(\alpha) = 3!q_3(\alpha)$.
Applicando lo schema di Horner (quarta volta) a $q_3(x)$ si ottingono $q_4(x)$ ed $r_4 = q_3(\alpha) = p_n'''(\alpha)/3!$.

In generale, applicando la $(i+1)$-esima volta lo schema di Horner si ottiene $p_n^{(i)}(\alpha)/i! = q_i(\alpha)$.

Esempio

$$p_4(x) = x^4 - 5x^3 + 3x^2 - 2x + 1 = \Big(\big((x-5)x+3\big)x - 2\Big)x + 1$$

1ª volta

$$\begin{array}{c|cccc|c} & 1 & -5 & 3 & -2 & 1 \\ 2 & & 2 & -6 & -6 & -16 \\ \hline & 1 & -3 & -3 & -8 & -15 \end{array}$$

$q_1(x)$: $1\ \ -3\ \ -3\ \ -8$; $r_1 = -15$; $p_4(x)$

2ª volta

$$\begin{array}{c|ccc|c} & 1 & -3 & -3 & -8 \\ 2 & & 2 & -2 & -10 \\ \hline & 1 & -1 & -5 & -18 \end{array}$$

$q_2(x)$: $1\ \ -1\ \ -5$; $r_2 = -18$; $q_1(x)$

3ª volta

$$\begin{array}{c|cc|c} & 1 & -1 & -5 \\ 2 & & 2 & 2 \\ \hline & 1 & 1 & -3 \end{array}$$

$q_3(x)$: $1\ \ 1$; $r_3 = -3$; $q_2(x)$

$$\begin{aligned} p_4(x) = (x-2)q_1(x) + r_1 &= (x-2)\Big((x-2)q_2(x) + r_2\Big) + r_1 \\ &= (x-2)\Big((x-2)\big((x-2)q_3(x) + r_3\big) + r_2\Big) + r_1, \quad q_3(x) = x + 1 \end{aligned}$$

In $x = 2$ si ha dunque:

volte di applicazione dello schema	quoziente	coefficienti quoziente				derivate in α (resti)
1ª	$q_1(x)$	1	-3	-3	-8	$p_4(2) = r_1 = -15$
2ª	$q_2(x)$		1	-1	-5	$p_4'(2) = r_2 = -18$
3ª	$q_3(x)$			1	1	$p_4''(2) = (2!) \cdot r_3 = 2 \cdot (-3) = -6$
4ª	$q_4(x)$				1	$p_4'''(2) = (3!) \cdot r_4 = 6 \cdot 3 = 18$
5ª	$q_5(x)$				0	$p_4^{iv}(2) = (4!) \cdot r_5 = 24 \cdot 1 = 24$