por **António Saraiva**[1]
[Centro de Matemática do Porto]

Algoritmo de Potenciação

Para potências elevadas, são claros os ganhos computacionais de usar algoritmos de potenciação em etapas. Desenvolve-se aqui um desses algoritmos.

Dado um real x e um natural n, para calcular x^n podemos usar o método seguinte: escreva-se n na base 2; na representação binária de n, substitua-se cada 0 pela letra Q e cada 1, com excepção do primeiro que é ignorado nesta transcrição, por QX; na palavra que daqui resulta, cada Q significa *elevar ao quadrado* e cada X representa a operação de *multiplicação por* x.

Por exemplo, se $n = 21$, n representa-se na base 2 por $10101_{(2)}$; desta representação binária obtemos a palavra $QQXQQX$, que indica que, para calcular x^{21}, devemos "quadrar, quadrar, multiplicar por x, quadrar, quadrar e multiplicar por x"; ou seja, devemos calcular sucessivamente x^2, x^4, x^5, x^{10}, x^{20} e x^{21}.

O procedimento delineado acima tem semelhanças com o algoritmo de Horner para determinar uma dízima de um natural de que se conhece a representação binária finita. De facto, se $N = a_r a_{r-1} \cdots a_1 a_{0(2)}$, onde $r \geq 0$, $a_r = 1$ e $a_i \in \{0, 1\}$ para todo o i, então este algoritmo traduz-se pelo esquema

	a_r	a_{r-1}	$\cdots$	a_1	a_0
2					

tabela que é completada do seguinte modo:

	a_r	a_{r-1}	a_{r-2}	$\cdots$	a_1	a_0
2	a_r	b_{r-1}	b_{r-2}	$\cdots$	b_1	b_0

onde

$$b_{r-1} = 2 \times a_r + a_{r-1}$$

$$b_{r-2} = 2 \times b_{r-1} + a_{r-2} = 2^2 \times a_r + 2 \times a_{r-1} + a_{r-2}$$

$$\vdots$$

$$b_0 = 2^r \times a_r + 2^{r-1} \times a_{r-1} + \cdots + 2 \times a_1 + a_0$$

Ora, se associarmos a N a palavra

$$X^{a_r} QX^{a_{r-1}} Q \cdots QX^{a_1} QX^{a_0}$$

e tivermos em conta que $a_r = 1$, podemos lê-la como

$$(x)\,(QX^{a_{r-1}} Q \cdots QX^{a_1} QX^{a_0})$$

[1]Trabalho realizado com a orientação de Maria Carvalho (FCUP) no âmbito de uma bolsa da Fundação para a Ciência e a Tecnologia de iniciação à investigação.

e interpretar a sua aplicação a x como a composição em sequência das operações $Q, X^{a_{r-1}}, Q, ..., Q, X^{a_0}$:

$$x = x^{a_r} \to x^{2a_r} \to x^{2a_r+a_{r-1}} \to x^{2^2a_r+2a_{r-1}} \cdots \to x^{2^r a_{r+2}{}^{r-1}a_{r-1}+\cdots+2a_1+a_0} = x^N.$$

1. Notação

Se F é um símbolo funcional, escreveremos $(\alpha)F$ para designar a aplicação de F a α. Sempre que a notação se simplificar sem prejuízo da clareza do argumento, omitiremos os parênteses, escrevendo apenas αF. E, ao contrário do que é usual, descreveremos a composição de funções pela direita: assim $(\alpha)(FG)$ representa a acção de F seguida da de G na variável α.

Consideremos o operador $Q: \mathbb{R} \to \mathbb{R}$, $(\alpha)Q = \alpha^2$ e, para cada real x, a aplicação de multiplicação por x, $X: \mathbb{R} \to \mathbb{R}$, $(\alpha)X = \alpha \times x$. Por convenção, para cada x real, $x^0 = 1$. Dado um natural m e qualquer operador $T: \mathbb{R} \to \mathbb{R}$, a composta de T consigo mesmo m vezes, $T \circ T \circ \cdots \circ T$, será denotada por T^m. Como usualmente, $T^0 \equiv$ *Identidade*. Note-se que, no algoritmo de potenciação, quando, para calcular x^N, se usa uma palavra formada com as letras Q e QX, estamos de facto a determinar a imagem de x pelo operador composição de certos iterados de Q e de X. Por convenção, a palavra vazia I será interpretada como a *Identidade* de $\mathbb{R}$.

Finalmente relembremos que todo o número natural n pode ser representado numa base b, onde b é um inteiro maior ou igual a dois. Mais precisamente, existe um único $m \in \mathbb{Z}^+$ e existem (dígitos únicos) $a_0, a_1, \cdots, a_m$ de $\{0, 1, \cdots, b-1\}$ com $a_m \neq 0$, tais que

$$n = a_m \times b^m + a_{m-1} \times b^{m-1} + \cdots + a_1 \times b + a_0;$$

a representação de n na base b é então $a_m\, a_{m-1} \cdots a_1\, a_{0(b)}$. Assim, na base tradicional (a decimal), os dígitos podem ser um dos dez algarismos habituais 0, 1, 2, $\cdots$, 9; neste caso é usual omitir o índice na representação, escrevendo-se apenas $n = a_m\, a_{m-1} \cdots a_1\, a_0$. Já na base $b = 2$, a representação (binária) de n consiste numa sequência finita de 0's e 1's, sendo o dígito mais à esquerda um 1. Por exemplo, se n é representado na base 10 por 27, então, na base 2, $n = 11011_{(2)}$ pois

$$27 = 16 + 8 + 2 + 1 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1.$$

2. Justificação do algoritmo

Sejam x um real e N um natural. Mostraremos, por indução em N, que o algoritmo produz efectivamente x^N. O caso $N = 1$ é imediato: $1 = 1_{(2)}$ e, como na transcrição desta representação binária numa palavra o primeiro dígito 1 é ignorado, obtemos a palavra vazia I; logo, o resultado do algoritmo é $(x)\, Id$, que é x^1.

Fixemos agora um real x e suponha-se que, para um dado $N \in \mathbb{Z}^+$, o algoritmo dá como resultado x^N. Seja $a_r\, a_{r-1} \cdots a_1\, a_{0(2)}$ a representação finita de N na base 2, onde $r \geq 0$, $a_r = 1$ e $a_i \in \{0, 1\}$ para todo o i. Temos as duas alternativas seguintes:

(I) $a_r = a_{r-1} = \cdots = a_1 = a_0 = 1$, isto é, $N = 111 \cdots 1_{(2)} = 2^{r+1} - 1$.

Neste caso $N+1 = 2^{r+1} = 1\,0\,0\cdots 0_{(2)}$, representação que tem $r+1$ zeros; a primeira e segunda etapas do algoritmo aplicadas a esta escrita binária conduzem à palavra Q^{r+1}; e portanto o algoritmo produz $(x)Q^{r+1}$. Ora, mostra-se facilmente por indução finita que

Lema 1. $\forall \alpha \in \mathbb{R}\, \forall m \in \mathbb{Z}_0^+ \;\; (\alpha)Q^m = \alpha^{2^m}$.

Prova: Fixemos um real α. Para $m=0$, a igualdade é imediata uma vez que, por convenção, $(\alpha)Q^0 = (\alpha)\,Id = \alpha = \alpha^1 = \alpha^{2^0}$. Suponhamos que a igualdade é válida para um natural m fixado. Então

$$(\alpha)Q^{m+1} = [(\alpha)Q^m]Q = (\alpha^{2^m})Q = (\alpha^{2^m})^2 = \alpha^{2^{m+1}}.\blacksquare$$

Resulta deste Lema que $(x)Q^{r+1} = x^{2^{r+1}}$, ou seja, $(x)Q^{r+1} = x^{N+1}$.

(II) *Pelo menos um dos a_i's é zero, para algum* $i \in \{0, 1, \cdots, r-1\}$.

Neste caso, seja $m \in \{0, 1, \cdots, r-1\}$ o menor índice tal que $a_m = 0$. Assim, na representação binária de N, há um dígito nulo na m-ésima casa seguido por m dígitos iguais a 1:

$$N = a_r \cdots a_{m+1}\, 0\, 1\, 1\, 1 \cdots 1_{(2)}.$$

E portanto,

$$N+1 = a_r \cdots a_{m+1} 1\, 0\, 0\, 0 \cdots 0_{(2)}$$

onde o dígito 1 assinalado é seguido por m zeros. A transcrição de $a_r \cdots a_{m+1}$, como dita o algoritmo, dá-nos uma palavra (eventualmente vazia) que corresponde a um operador P (eventualmente a *Identidade*). Além disso, aplicando o algoritmo a N obtemos a palavra $PQ(QX)^m$; analogamente, a $N+1$ corresponde a palavra $PQXQ^m$. Ora,

$$(x)\,[PQXQ^m] = (xP)(QXQ^m)$$

e

Proposição 1. $\forall m \in \mathbb{Z}_0^+ \;\; QXQ^m = Q(QX)^m X$.

Prova: Fixemos um real α e um inteiro não-negativo m. Temos

$$(\alpha)QXQ^m = [(\alpha)Q]XQ^m = (\alpha^2)XQ^m = (\alpha^2 \,.\, x)Q^m$$

e, pelo Lema 1,

$$(\alpha^2 .\, x)Q^m = (\alpha^2 \,.\, x)^{2^m}$$

ou seja, $(\alpha)QXQ^m = \alpha^{2^{m+1}} . x^{2^m}$. Além disso,

$$(\alpha)Q(QX)^m X = [\alpha^2 (QX)^m] \,.\, x$$

e

Lema 2. $\forall\beta\in\mathbb{R}\,\forall m\in\mathbb{Z}_0^+\ (\beta)(QX)^m=\beta^{2^m}.\,x^{2^m-1}$.
Prova: Consideremos um real β. Se $m=0$, a igualdade é imediata:

$$(\beta)(QX)^0=\beta=\beta^{2^0}.\,x^{2^0-1}.$$

Suponhamos agora, por indução, que a igualdade é válida para um natural m fixado. Então

$$(\beta)(QX)^{m+1}=[(\beta)(QX)^m]\,(QX)=(\beta^{2^m}x^{2^m-1})^2.\,x=\beta^{2^{m+1}}.\,x^{2^{m+1}-1}.\blacksquare$$

Retomemos a prova da Proposição. Deduzimos já que, para todo o natural m e todo o real α,

$$(\alpha)QXQ^m=\alpha^{2^{m+1}}.\,x^{2^m}$$

e

$$(\alpha)Q(QX)^mX=[\alpha^2(QX)^m].\,x.$$

Pelo Lema 2, o segundo membro desta última igualdade pode reescrever-se como

$$[\alpha^2(QX)^m].\,x=[(\alpha^2)^{2^m}.\,x^{2^m-1}].\,x$$

logo

$$(\alpha)Q(QX)^mX=\alpha^{2^{m+1}}.\,x^{2^m}$$

e portanto

$$(\alpha)QXQ^m=(\alpha)Q(QX)^mX.\blacksquare$$

Voltemos ao caso (II) do algoritmo. Já sabemos que

$$(x)\,[PQXQ^m]=(xP)(QXQ^m);$$

resulta agora da Proposição que

$$(xP)(QXQ^m)=(xP)\,[Q(QX)^mX];$$

além disso, uma vez que a composição é operação associativa,

$$(xP)\,[Q(QX)^mX]=[(x)(PQ(QX)^m)]X;$$

e, por hipótese de indução, $(x)(PQ(QX)^m)=x^N$; logo

$$[(x)(PQ(QX)^m)]X=(x^N).\,x=x^{N+1}.$$

3. Vantagens computacionais

O algoritmo de potenciação aqui analisado é naturalmente mais vantajoso, numa perspectiva computacional, se a potência é elevada. A economia nos cálculos deve-se ao uso eficiente de potências já calculadas. Por exemplo, o modo menos rápido de se determinar 13^{53} é o de multiplicar 13 por si mesmo 52 vezes. Mas, tendo em conta que $53 = 32 + 16 + 4 + 1$, temos

$$13^{53} = 13 \times 13^4 \times 13^{16} \times 13^{32} = (((((13^2)\,13)^2)^2\,13)^2)^2\,13 = 13\,(QXQQXQQX)$$

o que requer o uso da operação de *elevar ao quadrado* cinco vezes (para se obterem 13^2, 13^4, 13^8, 13^{16} e 13^{32}) além de três *multiplicações por* 13 (ou seja, 13×13^4, $13 \times 13^4 \times 13^{16}$ e $13 \times 13^4 \times 13^{16} \times 13^{32}$), num total de 8 multiplicações em vez das 52 acima.

O algoritmo é mais eficiente quando os expoentes têm representações binárias com mais dígitos iguais a zero. Por exemplo, no cálculo de x^{256} precisamos de 8 multiplicações, uma vez que $256 = 2^8 = 100000000_{(2)}$, e portanto

$$x^{256} = x^{2^8} = (((((((x^2)^2)^2)^2)^2)^2)^2)^2$$

o que corresponde a 8 operações de elevar ao quadrado. Mas, para determinar x^{255}, precisamos de 14 multiplicações, das quais 7 são operações de elevar ao quadrado e as restantes 7 são multiplicações finais:

$$255 = 2^8 - 1 = 11111111_{(2)}$$
$$x^{255} = x \times x^2 \times x^4 \times x^8 \times x^{16} \times x^{32} \times x^{64} \times x^{128} = (((((((x^2)\,x)^2\,x)^2\,x)^2\,x)^2\,x)^2\,x)^2\,x.$$

No que se segue iremos comprovar que, dados $x \neq 0$ e $N \in \mathbb{Z}^+$, no cálculo de x^N, usando o algoritmo de potenciação, o número $\mathfrak{m}$ de multiplicações efectuadas é sempre menor ou igual a $N-1$. Note-se que $N-1$ é o número de multiplicações utilizadas para obter x^N se fizermos o produto de x por si mesmo até se chegar à potência N.

Comecemos por observar a seguinte tabela:

N	$N_{(2)}$	*Palavra*	$\mathfrak{m}$	N - 1
1	1	I	0	0
2	10	Q	1	1
3	11	QX	2	2
4	100	Q^2	2	3
5	101	Q^2X	3	4
6	110	QXQ	3	5
7	111	$(QX)^2$	4	6

Quando $N = 1$, 2 ou 3, o valor de $\mathfrak{m}$ é $N-1$, enquanto que, para $4 \leq N \leq 7 = 2^3 - 1$, $\mathfrak{m}$ é estritamente menor que $N-1$. Na verdade esta vantagem é válida para $N \geq 4$. Verifiquemos esta afirmação para os casos não inscritos na

tabela ($N \geq 8 = 2^3 = 1000_{(2)}$).

Suponha-se que $N = a_r a_{r-1} \cdots a_1 a_{0(2)}$, onde $r \geq 3$, $a_r = 1$ e $a_i \in \{0, 1\}$ para $i = 0, 1, \cdots, r$. Claramente $\mathfrak{m}$ é menor ou igual ao número de multiplicações efectuadas quando aplicamos o algoritmo ao natural $11 \cdots 1_{(2)}$ que tem $r+1$ dígitos iguais a 1. Ora a palavra que resulta da transcrição desta representação binária é $(QX)^r$, e portanto o correspondente número de multiplicações é $2r$. Ou seja, em geral, $\mathfrak{m} \leq 2r$.

Por outro lado, é fácil verificar por indução que, para todo o inteiro r maior ou igual a 3, se tem $r \leq 2^{r-1} - 1$ ou, equivalentemente, $2r \leq 2^r - 2$. Ora, como $N \geq 2^r$, temos $N - 1 \geq 2^r - 1$, e portanto

$$\mathfrak{m} \leq 2r \leq 2^r - 2 < 2^r - 1 \leq N - 1.$$

Podemos acrescentar que, dado $N = a_r a_{r-1} \cdots a_1 a_{0(2)}$, onde $r \geq 0$, $a_r = 1$ e $a_i \in \{0, 1\}$ para $i = 0, 1, \cdots, r$, se tem

$$\mathfrak{m} = r + \sum_{i=0}^{r-1} a_i.$$

4. Generalização

O uso da base 2 nas secções anteriores não é essencial; podem estabelecer-se, por argumento idêntico, algoritmos de potenciação que utilizam a representação dos naturais noutras bases. Como anteriormente, o processo pode ser descrito pelas três etapas seguintes: dados $b \in \mathbb{Z}^+$, $b \geq 2$, $x \in \mathbb{R}$ e $N \in \mathbb{Z}^+$,

(1) escreva-se N na base b, digamos $N = c_r c_{r-1} \cdots c_1 c_{0(b)}$, onde $r \geq 0$, $c_r > 0$ e $c_i \in \{0, 1, \cdots, b-1\}$ para todo o i;

(2) na representação anterior, substitua-se cada c_i por X^{c_i}, sendo $X^0 = Id$, e coloque-se entre $X^{c_{i+1}}$ e X^{c_i} a letra E: obtemos assim a transcrição

$$X^{c_r} E X^{c_{r-1}} E \cdots E X^{c_1} E X^{c_0};$$

(3) a palavra de (2) é agora interpretada do seguinte modo: cada E significa elevar à potência b; cada X^{c_i} corresponde a multiplicar por x^{c_i};

(4) finalmente aplique-se esta leitura ao número 1, da esquerda para a direita, fazendo actuar sucessivamente X^{c_r}, E, $X^{c_{r-1}}$, E, ..., X^{c_0}:

$$1 \to x^{c_r} \to x^{bc_r} \to x^{bc_r + c_{r-1}} \to x^{b^2 c_r + bc_{r-1}} \cdots \to x^{b^r c_r + b^{r-1} c_{r-1} + \ldots + bc_1 + c_0} = x^N.$$

Exemplos:

(a) $$102 = 10210_{(3)} + XEX^0EX^2EXEX^0 \rightleftharpoons XE^2X^2EXE$$
$$1 \to x \to x^3 \to x^9 \to x^{11} \to x^{33} \to x^{34} \to x^{102}.$$

(b) $$102 = 204_{(7)} + X^2EX^0EX^4 \rightleftharpoons X^2E^2X^4$$
$$1 \to x^2 \to x^{14} \to x^{98} \to x^{102}.$$

Referências

[1] **Robert M. Young**, *Excursions in Calculus*, Dolciani Mathematical Expositions, 13 (1992) MAA.