La successió de Fibonacci

Problema 65. Sigui $U = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$.

(i) Demostreu que $U^n = \begin{bmatrix} u_{n+1} & u_n \\ u_n & u_{n-1} \end{bmatrix}$ on les entrades de la matriu són donades pels termes de la successió de Fibonacci.

(ii) Retrobeu el resultat $u_{n+1}u_{n-1} - u_n^2 = (-1)^n$ mitjançant el càlcul de $\det U^n$.

Solució.

(i) Ho demostrarem per inducció (començant amb $n = 1$, i suposant que la successió de Fibonacci comença en $n = 0$).

- Cas $n = 1$. Sabem que $u_0 = 0, u_1 = 1, u_2 = 1$, que coincideix amb les entrades de $U^1 = U$, per tant aquest cas queda confirmat.
- Suposem que és cert per tots els exponents menors o iguals a n. Voldrem veure que també es compleix per $n + 1$. Aplicant la hipòtesi d'inducció tenim que:

$$U^{n+1} = \begin{bmatrix} u_{n+1} & u_n \\ u_n & u_{n-1} \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} u_{n+1} + u_n & u_{n+1} \\ u_n + u_{n-1} & u_n \end{bmatrix}$$

I si ara apliquem la fórmula que descriu els termes de la successió de Fibonacci ($u_{n+1} = u_n + u_{n-1}$) podem reescriure els termes de la matriu per obtenir:

$$U^{n+1} = \begin{bmatrix} u_{n+2} & u_{n+1} \\ u_{n+1} & u_n \end{bmatrix}$$

per tant la proposició queda demostrada per tot $n \in \mathbb{N}$. □

Fixem-nos en que aquest resultat ens pot servir per trobar una forma ràpida per calcular l'element n-éssim de la successió de Fibonnacci, fent anar un algorisme semblant als algorismes d'exponenciació ràpids. Per exemple, si volguessim conèixer un terme en una posició parella de la successió (u_{2k}), podem prendre les matrius U^k i multiplicar-les: $U^k \cdot U^k = U^{2k}$, de forma que:

$$\begin{aligned} u_{2k} &= u_{k+1}u_k + u_k u_{k-1} = u_k(u_{k+1} + u_{k-1}) \\ u_{2k+1} &= u_{k+1}u_{k+1} + u_k u_k = u_{k+1}^2 + u_k^2 \\ u_{2k-1} &= u_k u_k + u_{k-1}u_{k-1} = u_k^2 + u_{k-1}^2 \end{aligned}$$

Per n grans això pot suposar un gran estalvi computacional.

Tot seguit podem veure un codi per SAGE (Python) d'exemple que aplica aquesta idea. Calculant el terme 100000 aquest algorisme ha anat 25 cops més ràpid que l'algorisme usual.

```
def FIB (n):
    if n == 0:
        return 0
    if n <= 2:
        return 1

    Us = [[[1 ,  1] ,[1 ,0]]]

    i = 1
    j = 0
    n2 = n/2
    UU = None
    while i <= n2:
        U = Us[j]
        UU = [[U[0][0]**2+U[0][1]**2 , U[0][1]*(U[0][0]+U[1][1]) ],
              [U[0][1]*(U[0][0]+U[1][1]) , U[0][1]**2+U[1][1]**2]]
        Us.append(UU)
        i*=2
        j+=1

    j-=1

    while n%i != 0:
        ni = int(floor(log(n%i)/log(2)))
        i += 2**ni

        U = Us[ni]
        UU = [[UU[0][0]*U[0][0]+UU[0][1]*U[1][0] ,
               UU[0][0]*U[0][1]+UU[0][1]*U[1][1]] ,
              [UU[1][0]*U[0][0]+UU[1][1]*U[1][0] ,
               UU[1][0]*U[0][1]+UU[1][1]*U[1][1]]]

    return UU[0][1]
```

(ii) Calculem el determinant de U^n :

$$\det \begin{bmatrix} u_{n+1} & u_n \\ u_n & u_{n-1} \end{bmatrix} = u_{n+1}u_{n-1} - u_n^2$$

Podem veure que $u_{n+1}u_{n-1} - u_n^2$ és el determinant de U^n. Seria fàcil agafar aquesta expressió i trobar la igualtat directament, però això ja es va fer a un exercici anterior i ens interessarà més veure-ho directament a partir del determinant.

Veiem que

$$\det \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = -1$$

sabem que el determinant és multiplicatiu ($\det(A \cdot B) = \det(A) \cdot \det(B)$), per tant tenim que $u_{n+1}u_{n-1} - u_n^2 = \det(U^n) = (\det(U))^n = (-1)^n$ com volíem. □