

**Le Rho de Pollard (Nombres de quelques chiffres).** Soit le nombre  $m$  composé, donc on veut trouver les facteurs. Il faut tout d'abord calculer une suite du type  $u_{k+1} = f(u_k) \pmod m$  de grande période, de sorte que les  $u_k$  soient distincts le plus longtemps possible.

0	1	2	...	p	p+1	p+2	...	kp	kp+1	kp+2	...	m-1
		$u_i$	$u_l$		$u_k$			$u_h$		$u_j$		

TABLEAU 1 – Distribution des multiples de  $p$  modulo  $m$ .

Ensuite, l'idée est que si  $p$  est un facteur de  $m$ , les  $u_k$  distincts modulo  $m$  le seront moins souvent modulo  $p$  (voir Tableau 1). Dans ce cas, si  $u_i = u_j \pmod p$  alors le pgcd de  $m$  et de  $u_i - u_j$  vaut  $kp$  et est donc un facteur non trivial de  $m$ . Si les  $u_i$  sont bien tous distincts, le calcul se termine alors en au plus  $p$  étapes. Une première version de l'algorithme consiste à produire des  $u_i$  et à chaque nouvel élément de calculer les pgcd avec tous les précédents  $u_k$ . Cette version présente deux défauts : tout d'abord il faut stocker de l'ordre de  $p$  éléments ; en outre, il faut faire  $j^2$  calculs de pgcd si  $i$  et  $j > i$  sont les plus petits indices tels que  $u_i = u_j \pmod p$ . La deuxième astuce de Pollard est d'utiliser la détection de cycle de Floyd. Il s'agit de stocker uniquement les  $u_k$  tels que  $k$  soit une puissance de 2. En effet, puisque les  $u_k$  sont générés par une fonction, si  $u_i = u_j$ , alors pour tout  $h \geq 0$ ,  $u_{i+h} = u_{j+h}$  et un cycle se crée modulo  $p$ , même si il n'est pas visible directement.

En ne stockant que des puissances de 2, le cycle sera donc détecté seulement pour  $u_{2^a} = u_{2^a+j-i}$  avec  $2^{a-1} < i \leq 2^a$ , comme on en voit l'illustration sur la figure ??.

Or,  $2^a + j - i < 2i + j - i = i + j < 2j$ . Donc on effectue au plus le double d'étapes nécessaires. Un seul pgcd est calculé à chaque étape et un seul élément supplémentaire est stocké au long de l'algorithme. Cela donne l'algorithme extrêmement simple donné ci-dessous.

---

**Algorithme 1** Factorisation de Pollard avec détection de cycle de Floyd, sous forme de rho.

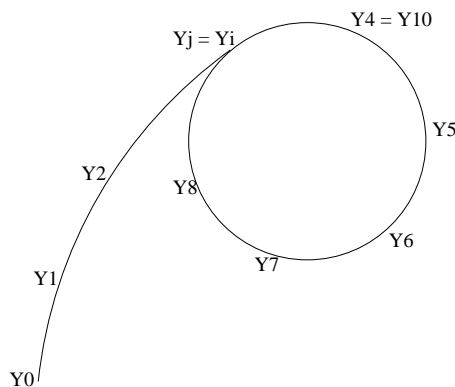
---

**Require:** Un entier  $m$ , composé.

**Ensure:**  $p$ , un facteur non trivial de  $m$ .

```

 $p \leftarrow 1$ ;
Générer aléatoirement  $y$ ;
 $k \leftarrow 0$ ;
while ( $p == 1$ ) do
  if  $k$  est une puissance de 2 then
     $x \leftarrow y$ ;
  end if
   $y \leftarrow f(y) \pmod m$ ;
   $p \leftarrow \text{pgcd}(y - x, m)$ ;
  Incréments  $k$ ;
end while
Renvoyer  $p$ .
```



Si on prend  $f(u) = u^2 + 1$ , de sorte que les  $u_i$  soient bien tous distincts modulo  $m$  par exemple, il faudra au pire  $2p$  itérations si  $p$  est le plus petit facteur de  $m$ , mais beaucoup moins en général :

**Théorème 1.** *L'algorithme Rho de Pollard a plus d'une chance sur deux de se terminer en  $O(\sqrt{p})$  étapes.*

**Preuve.** La preuve est encore du type de celle du célèbre paradoxe des anniversaires. Si  $k$  valeurs distinctes  $u_i$  sont tirées au hasard, il y a  $A_p^k$  combinaisons qui ne donnent pas de collisions entre les  $u_i$  sur un total de  $p^k$ . Pour que la probabilité de trouver un facteur soit plus grande que  $1/2$  il faut donc d'après le théorème des anniversaires que  $k > 0.5 + 1.18\sqrt{p}$ .  $\square$

En pratique, cet algorithme factorise en quelques secondes les nombres de 1 à environ 25 chiffres (avec des facteurs de 12 ou 13 chiffres, cela donne environ 10 millions d'itérations !) et devient très rapidement inutilisable pour des facteurs au-delà de 15 chiffres.