D'où
$$\left(x^{\phi(n)}\right)^t \cdot x = x + x u \varphi$$
$$= x + r\cdot p\cdot u\cdot q$$
$$= x + r\cdot n\cdot q$$
$$\equiv x \pmod{n}. \quad \boxtimes$$

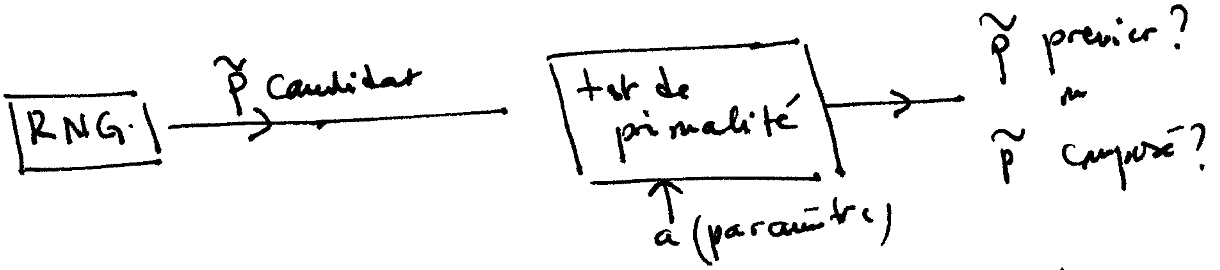
<u>Remarques importantes</u>. Jusqu'à maintenant tout est beau ! Mais pour des raisons d'<u>efficacité</u> et de <u>sécurité</u>, montre

A) Dans l'algorithme RSA1, il faut utiliser un algorithme d'<u>exponentiation modulaire</u> assez rapide (basé sur la <u>décomposition binaire</u> d'un nombre : <u>square and multiply</u>.
Dans RSA2, étape 4, utiliser <u>l'algorithme d'Euclide étendu</u> pour trouver la clé d (voir <u>Rappels</u>).

B) Engendrer des nombres premiers p et q suffisamment grand et équilibré. Et donc on est ramené à des <u>tests de primalité</u>

(2.2) <u>Recherche de nombres premiers assez grands</u>

si le module $n = p\cdot q$ est de taille $|n|_2 = \lceil \log_2 n \rceil \simeq 1024$,
$|p| \simeq |q| = 512$ bits. Comment engendrer les nombres premiers ?



Question : Q1) Combien d'entiers aléatoires qu'on doit engendrer pour dire que $\tilde{p}$ est premier ?

Q2) Rapidité du test de primalité ?