

# CSCI 124/224

## Discrete Structures II: Fast Modular Exponentiation

Poorvi L. Vora

So far, we have studied encryption functions that are constructed from addition and multiplication  $\text{mod } m$ . These encryption functions, are, however, not secure enough to use today. Among the encryption functions in use today, two of the most popular ones are based on taking the power of an element in  $\mathbb{Z}_m$ . In this section, we study a fast algorithm for finding a power  $\text{mod } m$ . We will not pursue the encryption functions based on exponentiation any further in this course, though. These are taught in more detail in the cryptography courses: CSci 162 (undergraduate Cryptography), CSci 284 (graduate introductory Cryptography) and CSci 381 (graduate Advanced Cryptography).

Consider how one might compute  $x^k \text{ mod } m$  for large values of  $x$ ,  $k$  and  $m$ . If one actually performed  $k - 1$  multiplications:  $x \times x \times \dots \times x$ , that could be a large number of multiplications. On the other hand, one might use the square-and-multiply rule which is considerably more efficient.

Let's consider an example with small enough numbers to compute results by hand:  $5^{13} \text{ mod } 17$ .

$$5^{13} \text{ mod } 17 = ((5^2)^2)^2 \times (5^2)^2 \times 5$$

which involves only 5 multiplications instead of 12. Or another example:

$$5^{15} \text{ mod } 17 = ((5^2)^2)^2 \times (5^2)^2 \times 5^2 \times 5$$

which involves 6 multiplications.

This can be formalized as follows:

*exponentiation(x, b, m) /\* x<sup>b</sup> mod m \*/*

*z := 1*

*/\* b<sub>i</sub> = i<sup>th</sup> bit in binary representation of b \*/*

*for i = l - 1 downto 0*

*z := z<sup>2</sup> mod n*

*if b<sub>i</sub> = 1 z := z × x mod n*

*endfor*

*return(z)*

**Example:**  $5^{51} \text{ mod } 7$ .

$$51 = 110011$$

$i$	$b_i$	$z_{first}$	$z_{second}$
5	1	1	5
4	1	$25 \bmod 7 = 4$	$20 \bmod 7 = 6$
3	0	1	1
2	0	1	1
1	1	1	5
0	1	4	6

**Example:**  $7^{29} \bmod 11$

$$29 = 11101$$

$i$	$b_i$	$z_{first}$	$z_{second}$
4	1	1	7
3	1	5	2
2	1	4	6
1	0	3	3
0	1	9	8