Computing x^n

Naive algorithm for computing x^n (RT: O(n)):

 $\label{eq:naivePower} \begin{aligned} & \text{naivePower}(x,n,p) \text{ // Return } x^n mod p \\ & prod \leftarrow 1 \\ & \textbf{for } i \leftarrow 1 \textbf{ to } n \textbf{ do} \end{aligned}$

 $prod \leftarrow (prod * x)\%p$

return prod

DAC algorithms and Recursion

Balaji Raghavachari rbk@utdallas.edu

© Balaji Raghavachari

3

Implementation of ADSA

© Balaji Raghavachari

4

Implementation of ADSA

DAC algorithm

```
\begin{array}{l} \mathsf{Power}(x,n,p) \ // \ \mathsf{RT:} \ O(\log n) \\ \mathbf{if} \ n = 0 \ \mathbf{then} \\ \qquad \qquad \mathbf{return} \quad 1 \\ \mathbf{else} \ \mathbf{if} \ n = 1 \ \mathbf{then} \\ \qquad \qquad \mathbf{return} \quad x \\ \mathbf{else} \\ \qquad \qquad half \leftarrow \mathsf{Power}(x,n/2,p) \\ \qquad \qquad res \leftarrow (half*half)\%p \\ \qquad \qquad \mathbf{if} \ n\%2 = 0 \ \mathbf{then} \\ \qquad \qquad \qquad \qquad \mathbf{return} \quad res \\ \qquad \qquad \mathbf{else} \\ \qquad \qquad \qquad \mathbf{return} \quad (res*x)\%p \\ \end{array}
```

Alternate DAC algorithm

 $\begin{array}{l} \mathsf{Power}(x,n,p) \ // \ \mathsf{RT:} \ O(\log n) \\ \mathbf{if} \ n = 0 \ \mathbf{then} \\ \qquad \qquad \mathbf{return} \ \ 1 \\ \mathbf{else} \ \mathbf{if} \ n = 1 \ \mathbf{then} \\ \qquad \qquad \mathbf{return} \ \ x \\ \mathbf{else} \\ \qquad \qquad res \leftarrow \mathsf{Power}((x*x)\%p, n/2, p) \\ \qquad \mathbf{if} \ n\%2 = 0 \ \mathbf{then} \\ \qquad \qquad \mathbf{return} \ \ res \\ \qquad \mathbf{else} \\ \qquad \qquad \mathbf{return} \ \ (res*x)\%p \end{array}$