Integer Algorithms: Prime Numbers

\( p \) is prime \( \equiv \)
\( p > 1 \) and \( \forall d \ ((d > 0 \land d \text{ divides } p) \rightarrow (d = 1 \lor d = p)) \)
\( p \) is composite \( \equiv p > 1 \) and \( p \) is not prime.
If \( n \) is composite, then \( \exists d \ (d \text{ divides } n \text{ and } 1 < d \leq \sqrt{n}) \).

\textbf{procedure} prime\( (n: \mathbb{Z} \text{ with } n > 1) \)

\textbf{for} \( d := 2 \text{ to } \lfloor \sqrt{n} \rfloor \)

\textbf{if} \( n \) is divisible by \( d \)

\textbf{then return} false

\textbf{return} true

\( \text{prime}(n) \) performs \( O(\sqrt{n}) \) divisions.

\textbf{Greatest Common Divisor}

\( d \) is the greatest common divisor of \( a \) and \( b \) \( \equiv \)
\( d \) is the largest integer that divides \( a \) and \( b \).
\( a \mod m = r \equiv \)
\( m > 0, 0 \leq r < m, \text{ and } \exists q (a = mq + r). \)
If \( a = bq + r \), then \( \gcd(a, b) = \gcd(b, r) \).

\textbf{procedure} gcd\( (a, b: \mathbb{Z}^+) \)

\textbf{while} \( b \neq 0 \)

\( r := a \mod b \)
\( a := b \)
\( b := r \)

\textbf{return} \( a \)

\( \gcd(a, b) \) performs \( O(\log(a + b)) \mod \) operations.
Representation of Integers

If $b > 1$ and $n \geq 1$, $n$ can be uniquely represented by:

$$n = \sum_{i=0}^{k} a_i b^i$$

where $k \geq 0$, $0 \leq a_i < b$, and $a_k > 0$

**procedure** base_expansion ($n$, $b$: $\mathbb{Z}^+$ with $b > 1$)

```plaintext
k := 0
while $n \neq 0$
    $a_k := n \mod b$
    $n := \lfloor n/b \rfloor$
    $k := k + 1$
return ($a_{k-1}, \ldots, a_0$)
```

`base_expansion(n, b)` performs $O(\log_b n)$ mods and divisions, e.g., $n$ can be represented with $O(\log n)$ bits.

---

Binary Addition

Binary addition adds one column at a time.

**procedure** add($a$, $b$: $\mathbb{Z}^+$)

```plaintext
(a_{n-1}, \ldots, a_0) := base_expansion(a, 2)
(b_{n-1}, \ldots, b_0) := base_expansion(b, 2)
carry := 0
for $j := 0$ to $n - 1$
    $s_j := (a_j + b_j + carry) \mod 2$
    carry := $\lfloor(a_j + b_j + carry)/2 \rfloor$
$s_n := carry$
return ($s_n, \ldots, s_0$)
```

`add(a, b)` performs $O(n)$ bit operations, where $a$ and $b$ are represented with $n$ bits.
Binary Multiplication

Binary multiplication performs multiple additions.

**procedure** multiply\((a, b: \mathbb{Z}^+)\)

\[
(a_{n-1}, \ldots, a_0) := base\_expansion(a, 2) \\
(b_{n-1}, \ldots, b_0) := base\_expansion(b, 2) \\
p := 0 \\
\text{for } j := 0 \text{ to } n - 1 \\
\quad \text{if } b_j = 1 \text{ then } p := add(p, a) \\
\quad \text{shift } a \text{ one place adding one 0} \\
\text{return } p
\]

\(multiply(a, b)\) performs \(O(n^2)\) bit operations, where \(a\) and \(b\) are represented with \(n\) bits.