Around Montgomery's trick: A taste of a bit hack

Marc Moreno Maza (after Wei Pan's notes)

University of Western Ontario, London, Ontario (Canada)

CS 4435 - CS 9624

Introduction

- Let a, b, p be number-like objects (integer numbers, univariate polynomials over a field) and such that $p \notin \{-1, 0, 1\}$.

Introduction

- Let a, b, p be number-like objects (integer numbers, univariate polynomials over a field) and such that $p \notin \{-1, 0, 1\}$.
- More formally let a, b, p be elements in an Euclidean domain with p not a unit.

Introduction

- Let a, b, p be number-like objects (integer numbers, univariate polynomials over a field) and such that $p \notin \{-1, 0, 1\}$.
- More formally let a, b, p be elements in an Euclidean domain with p not a unit.
- Computing $\boxed{(a, b, p) \longmapsto (ab) \mod p}$ is a fundamental and challenging operation.

Introduction

- Let a, b, p be number-like objects (integer numbers, univariate polynomials over a field) and such that $p \notin \{-1, 0, 1\}$.
- More formally let a, b, p be elements in an Euclidean domain with p not a unit.
- Computing $\boxed{(a, b, p) \longmapsto (ab) \mod p}$ is a fundamental and challenging operation.
- If a, b, p have large sizes, then FFT-based arithemtic and the fast division trick (S. Cook, 1966) (H. T. Kung, 1974) and (M. Sieveking, 1972) provides a practically efficient solution

Introduction

- Let a, b, p be number-like objects (integer numbers, univariate polynomials over a field) and such that $p \notin \{-1, 0, 1\}$.
- More formally let a, b, p be elements in an Euclidean domain with p not a unit.
- Computing $\boxed{(a, b, p) \longmapsto (ab) \mod p}$ is a fundamental and challenging operation.
- If a, b, p have large sizes, then FFT-based arithemtic and the fast division trick (S. Cook, 1966) (H. T. Kung, 1974) and (M. Sieveking, 1972) provides a practically efficient solution
- If a, b, p have small sizes, say are machine integers, then enter Peter Montgomery and his famous reduction (Math. Computation, vol. 44, pp. 519–521, 1985) improved by Xin Li in his PhD thesis (University of Western Ontario 2009).

The Original Montgomery Trick (1/2)

- Let x, p be integers such that $p \geq 2$. In practice p is a prime. We shall compute $x \mod p$ in an *indirect way*.

The Original Montgomery Trick (1/2)

- Let x, p be integers such that $p \geq 2$. In practice p is a prime. We shall compute $x \mod p$ in an *indirect way*.
- Consider a positive integer $R \geq p$ such that $\gcd(R, p) = 1$. Hence there exists integers R^{-1}, p' such that

$$RR^{-1} - p\,p' = 1 \text{ and } 0 < p' < R.$$

The Original Montgomery Trick (1/2)

- Let x, p be integers such that $p \geq 2$. In practice p is a prime. We shall compute $x \mod p$ in an *indirect way*.
- Consider a positive integer $R \geq p$ such that $\gcd(R, p) = 1$. Hence there exists integers R^{-1}, p' such that

$$RR^{-1} - p\,p' = 1 \;\text{ and }\; 0 < p' < R.$$

- Consider the following two Euclidean divisions:

$$\begin{array}{r|l} x & R \\ \hline d & c \end{array} \quad \text{and} \quad \begin{array}{r|l} dp' & R \\ \hline f & e \end{array}.$$

The Original Montgomery Trick (1/2)

- Let x, p be integers such that $p \geq 2$. In practice p is a prime. We shall compute $x \mod p$ in an *indirect way*.
- Consider a positive integer $R \geq p$ such that $\gcd(R, p) = 1$. Hence there exists integers R^{-1}, p' such that

$$RR^{-1} - p\,p' = 1 \text{ and } 0 < p' < R.$$

- Consider the following two Euclidean divisions:

$$\begin{array}{r|l} x & R \\ \hline d & c \end{array} \quad \text{and} \quad \begin{array}{r|l} dp' & R \\ \hline f & e \end{array}.$$

- Hence we have:

$$x + fp \; = \; cR + d + (dp' - eR)p \; = \; cR + d(1 + pp') - epR.$$

The Original Montgomery Trick (1/2)

- Let x, p be integers such that $p \geq 2$. In practice p is a prime. We shall compute $x \mod p$ in an *indirect way*.
- Consider a positive integer $R \geq p$ such that $\gcd(R, p) = 1$. Hence there exists integers R^{-1}, p' such that

$$RR^{-1} - p\,p' = 1 \quad \text{and} \quad 0 < p' < R.$$

- Consider the following two Euclidean divisions:

$$\begin{array}{r|l} x & R \\ \hline d & c \end{array} \quad \text{and} \quad \begin{array}{r|l} dp' & R \\ \hline f & e \end{array}.$$

- Hence we have:

$$x + fp \;=\; cR + d + (dp' - eR)p \;=\; cR + d(1 + pp') - epR.$$

- Therefore $\boxed{x + fp \text{ writes } qR}$ and thus $\boxed{\frac{x}{R} \equiv q \mod p}$.

The Original Montgomery Trick (2/2)

- Suppose $p > 2$ is a prime and R is a power of 2. Then we have obtained a procedure computing $\frac{x}{R} \mod p$ for $0 \leq x < p^2$, amounting to 2 multiplications, 2 additions and 3 shifts.

The Original Montgomery Trick (2/2)

- Suppose $p > 2$ is a prime and R is a power of 2. Then we have obtained a procedure computing $\frac{x}{R} \mod p$ for $0 \leq x < p^2$, amounting to 2 multiplications, 2 additions and 3 shifts.
- Recall the three divisions:

$$\begin{array}{c|c} x & R \\ \hline d & c \end{array} \quad \text{and} \quad \begin{array}{c|c} dp' & R \\ \hline f & e \end{array} \quad \text{and} \quad \begin{array}{c|c} x + fp & R \\ \hline 0 & q \end{array}$$

The Original Montgomery Trick (2/2)

- Suppose $p > 2$ is a prime and R is a power of 2. Then we have obtained a procedure computing $\frac{x}{R} \mod p$ for $0 \leq x < p^2$, amounting to 2 multiplications, 2 additions and 3 shifts.
- Recall the three divisions:

$$\begin{array}{c|c} x & R \\ \hline d & c \end{array} \quad \text{and} \quad \begin{array}{c|c} dp' & R \\ \hline f & e \end{array} \quad \text{and} \quad \begin{array}{c|c} x + fp & R \\ \hline 0 & q \end{array}$$

- The result is q or $q - p$ since $\boxed{\frac{x}{R} \equiv q \mod p}$ and we have:

$$0 \leq x < p^2 \quad \Rightarrow \quad 0 \leq q < 2p.$$

The Original Montgomery Trick (2/2)

- Suppose $p > 2$ is a prime and R is a power of 2. Then we have obtained a procedure computing $\frac{x}{R} \mod p$ for $0 \le x < p^2$, amounting to 2 multiplications, 2 additions and 3 shifts.
- Recall the three divisions:

$$\begin{array}{c|c} x & R \\ \hline d & c \end{array} \quad \text{and} \quad \begin{array}{c|c} dp' & R \\ \hline f & e \end{array} \quad \text{and} \quad \begin{array}{c|c} x + fp & R \\ \hline 0 & q \end{array}$$

- The result is q or $q - p$ since $\boxed{\frac{x}{R} \equiv q \mod p}$ and we have:

$$0 \le x < p^2 \quad \Rightarrow \quad 0 \le q < 2p.$$

- To compute in $\mathbb{Z}/p\mathbb{Z}$, we map each $a \in \mathbb{Z}/p\mathbb{Z}$ to $aR \in \mathbb{Z}/p\mathbb{Z}$. Then the above procedure gives us $\frac{aRbR}{R} \mod p$, that is, the image of ab in this new representation.

The Improved Montgomery Trick (1/5)

- Suppose $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.

The Improved Montgomery Trick (1/5)

- Suppose $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.
- Let $R := 2^\ell$ and $0 \leq x \leq (p-1)^2$. We get $\frac{x}{R} \mod p$ by:

$$\begin{array}{c|c} x & R \\ \hline r_1 & q_1 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_1 & R \\ \hline r_2 & q_2 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_2 & R \\ \hline 0 & q_3 \end{array}$$

The Improved Montgomery Trick (1/5)

- Suppose $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.
- Let $R := 2^\ell$ and $0 \leq x \leq (p-1)^2$. We get $\frac{x}{R} \mod p$ by:

$$\begin{array}{c|c} x & R \\ \hline r_1 & q_1 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_1 & R \\ \hline r_2 & q_2 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_2 & R \\ \hline 0 & q_3 \end{array}$$

- Using $c2^n \equiv -1 \mod p$ we have:

$$\frac{x}{R} \equiv q_1 + \frac{r_1}{R} \equiv q_1 - q_2 - \frac{r_2}{R} \equiv q_1 - q_2 + q_3 \mod p.$$

The Improved Montgomery Trick (1/5)

- Suppose $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.
- Let $R := 2^\ell$ and $0 \leq x \leq (p-1)^2$. We get $\frac{x}{R} \mod p$ by:

$$\begin{array}{c|c} x & R \\ \hline r_1 & q_1 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_1 & R \\ \hline r_2 & q_2 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_2 & R \\ \hline 0 & q_3 \end{array}$$

- Using $c2^n \equiv -1 \mod p$ we have:

$$\frac{x}{R} \equiv q_1 + \frac{r_1}{R} \equiv q_1 - q_2 - \frac{r_2}{R} \equiv q_1 - q_2 + q_3 \mod p.$$

- The last equality requires a proof. We have:

$$r_2 = c2^n r_1 - q_2 R = c2^n r_1 - q_2 2^\ell.$$

Hence $\boxed{2^n \mid r_2}$ thus $\boxed{2^{2n} \mid c2^n r_2}$ and $\boxed{R \mid c2^n r_2}$.

The Improved Montgomery Trick (2/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.
- Recall $R := 2^\ell$ and $0 \leq x \leq (p-1)^2$. We get $\frac{x}{R} \mod p$ by:

$$\begin{array}{r|l} x & R \\ \hline r_1 & q_1 \end{array} \quad \text{and} \quad \begin{array}{c|l} c2^n r_1 & R \\ \hline r_2 & q_2 \end{array} \quad \text{and} \quad \begin{array}{c|l} c2^n r_2 & R \\ \hline 0 & q_3 \end{array}$$

leading to $\frac{x}{R} \equiv q_1 - q_2 + q_3 \mod p$.

The Improved Montgomery Trick (2/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.
- Recall $R := 2^\ell$ and $0 \leq x \leq (p-1)^2$. We get $\frac{x}{R} \mod p$ by:

$$\begin{array}{c|c} x & R \\ \hline r_1 & q_1 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_1 & R \\ \hline r_2 & q_2 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_2 & R \\ \hline 0 & q_3 \end{array}$$

  leading to $\frac{x}{R} \equiv q_1 - q_2 + q_3 \mod p$.
- Moreover we have:

$$-(p-1) < q_1 - q_2 + q_3 < 2(p-1).$$

  Hence the desired output is either $(q_1 - q_2 + q_3) + p$, or $q_1 - q_2 + q_3$ or $(q_1 - q_2 + q_3) - p$

The Improved Montgomery Trick (2/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil \leq b$ on b-bit machine words.
- Recall $R := 2^\ell$ and $0 \leq x \leq (p-1)^2$. We get $\frac{x}{R} \mod p$ by:

$$\begin{array}{c|c} x & R \\ \hline r_1 & q_1 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_1 & R \\ \hline r_2 & q_2 \end{array} \quad \text{and} \quad \begin{array}{c|c} c2^n r_2 & R \\ \hline 0 & q_3 \end{array}$$

  leading to $\frac{x}{R} \equiv q_1 - q_2 + q_3 \mod p$.
- Moreover we have:

$$-(p-1) < q_1 - q_2 + q_3 < 2(p-1).$$

  Hence the desired output is either $(q_1 - q_2 + q_3) + p$, or $q_1 - q_2 + q_3$ or $(q_1 - q_2 + q_3) - p$
- Indeed $0 \leq x \leq (p-1)^2$ and $p \leq R$ imply

$$q_1 = x \operatorname{quo} R \leq (p-1)^2/R < p - 1.$$

  Next, we have: $q_2 = c2^n r_1 \operatorname{quo} R < c2^n = p - 1$, since $r_1 < R$. Similarly, we have $q_3 < p - 1$.

The Improved Montgomery Trick (3/5)

We describe now the C implementation for 32-bit machine integer assuming that we have at hand the following function:

```
/**
 * Input  : The addresses of two unsigned machine integers a, b
 * Output : Store (a * b) quo 2^32 into a, and
              store (a * b) mod 2^32 into b
 *
 **/

inline void MulHiLoUnsigned (uint32_t *a, uint32_t *b) {

     uint64_t prod;
     prod = (uint64_t)(*a) * (uint64_t)(*b);

     *a = (uint32_t) (prod >> 32);
     *b = (uint32_t) prod;
}
```

The Improved Montgomery Trick (4/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil$. Recall $R := 2^\ell$.
- Let a, b be non-negative 32-bit machine integers less than p. We show how to compute $\frac{ab}{R} \mod p$.

The Improved Montgomery Trick (4/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil$. Recall $R := 2^\ell$.
- Let a, b be non-negative 32-bit machine integers less than p. We show how to compute $\frac{ab}{R} \mod p$.
- $q_1, 2^{32-\ell} r_1 := \texttt{MulHiLoUnsigned}(a, 2^{32-\ell} b)$

The Improved Montgomery Trick (4/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil$. Recall $R := 2^\ell$.
- Let a, b be non-negative 32-bit machine integers less than p. We show how to compute $\frac{ab}{R} \mod p$.
- $q_1, 2^{32-\ell} r_1 := \texttt{MulHiLoUnsigned}(a, 2^{32-\ell} b)$
- $q_2, 2^{32-\ell} r_2 := \texttt{MulHiLoUnsigned}(2^{32-\ell} r_1, 2^n c)$

The Improved Montgomery Trick (4/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil$. Recall $R := 2^\ell$.
- Let a, b be non-negative 32-bit machine integers less than p. We show how to compute $\frac{ab}{R} \mod p$.
- $q_1, 2^{32-\ell} r_1 := \texttt{MulHiLoUnsigned}(a, 2^{32-\ell} b)$
- $q_2, 2^{32-\ell} r_2 := \texttt{MulHiLoUnsigned}(2^{32-\ell} r_1, 2^n c)$
- $q_3 := c \frac{r_2}{2^{\ell-n}}$. The division $\frac{r_2}{2^{\ell-n}}$ is exact and the multiplication $c \frac{r_2}{2^{\ell-n}}$ is correct on 32 bits.

The Improved Montgomery Trick (4/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil$. Recall $R := 2^\ell$.
- Let a, b be non-negative 32-bit machine integers less than p. We show how to compute $\frac{ab}{R} \mod p$.
- $q_1, 2^{32-\ell} r_1 := \mathtt{MulHiLoUnsigned}(a, 2^{32-\ell} b)$
- $q_2, 2^{32-\ell} r_2 := \mathtt{MulHiLoUnsigned}(2^{32-\ell} r_1, 2^n c)$
- $q_3 := c \frac{r_2}{2^{\ell-n}}$. The division $\frac{r_2}{2^{\ell-n}}$ is exact and the multiplication $c \frac{r_2}{2^{\ell-n}}$ is correct on 32 bits.
- Let $A := q_1 - q_2 + q_3$. Then we execute the following code:

```
A += (A >> 31) & p;
A -= p;
A += (A >> 31) & p;
```

The Improved Montgomery Trick (4/5)

- Recall $p > 2$ is a Fourier prime, that is, $p - 1 = c2^n$ and $\ell \leq 2n$ where $\ell = \lceil \log_2(p) \rceil$. Recall $R := 2^\ell$.
- Let a, b be non-negative 32-bit machine integers less than p. We show how to compute $\frac{ab}{R} \mod p$.
- $q_1, 2^{32-\ell} r_1 := \texttt{MulHiLoUnsigned}(a, 2^{32-\ell} b)$
- $q_2, 2^{32-\ell} r_2 := \texttt{MulHiLoUnsigned}(2^{32-\ell} r_1, 2^n c)$
- $q_3 := c \frac{r_2}{2^{\ell-n}}$. The division $\frac{r_2}{2^{\ell-n}}$ is exact and the multiplication $c \frac{r_2}{2^{\ell-n}}$ is correct on 32 bits.
- Let $A := q_1 - q_2 + q_3$. Then we execute the following code:

```
A += (A >> 31) & p;
A -= p;
A += (A >> 31) & p;
```

- Finally we have performed 6 shifts, 5 additions, 2 64-bit multiplications and 1 32-bit multiplication.

The Improved Montgomery Trick (5/5)

- Consider $p = 257 = 1 + 2^8$. Hence $c = 1$, $n = 8$, $\ell = 9$ and $R = 2^9$.
- Take $a = 131$ and $b = 187$.
- Compute $2^{32-\ell} b = 1568669696$.
- Compute $q_1 = 47$ and $2^{32-\ell} r_1 = 3632267264$.
- Compute $q_2 = 216$ and $2^{32-\ell} r_2 = 2147483648$.
- Compute $q_3 = c \frac{r_2}{2^{\ell-n}} = 128$.
- Compute $A = q_1 - q_2 + q_3 = -41$.
- Ajust to get $\frac{ab}{R} \equiv 216 \mod p$.