

Mersenne primes and the Lucas–Lehmer test

Manuel Eberl

March 17, 2025

Abstract

This article provides formal proofs of basic properties of Mersenne numbers, i. e. numbers of the form $2^n - 1$, and especially of Mersenne primes. In particular, an efficient, verified, and executable version of the Lucas–Lehmer test is developed. This test decides primality for Mersenne numbers in time polynomial in n .

Contents

1	Auxiliary material	2
1.1	Auxiliary number-theoretic material	2
1.2	Auxiliary algebraic material	3
2	The Lucas–Lehmer test	4
2.1	General properties of Mersenne numbers and Mersenne primes	4
2.2	The Lucas–Lehmer sequence	6
2.3	The ring $\mathbb{Z}[\sqrt{3}]$	6
2.4	The ring $(\mathbb{Z}/m\mathbb{Z})[\sqrt{3}]$	7
2.5	$\mathbb{Z}[\sqrt{3}]$ as a subring of $\mathbb{R}$	9
2.6	The canonical homomorphism $\mathbb{Z}[\sqrt{3}] \rightarrow (\mathbb{Z}/m\mathbb{Z})[\sqrt{3}]$	10
2.7	Correctness of the Lucas–Lehmer test	11
2.8	A first executable version Lucas–Lehmer test	12
3	Efficient code for testing Mersenne primes	12
3.1	Efficient computation of remainders modulo a Mersenne number	13
3.2	Efficient code for the Lucas–Lehmer sequence	14
3.3	Code for the Lucas–Lehmer test	14
3.4	Examples	15

1 Auxiliary material

```
theory Lucas-Lehmer-Auxiliary
imports
  HOL-Algebra.Ring
  Probabilistic-Prime-Tests.Jacobi-Symbol
begin
```

1.1 Auxiliary number-theoretic material

```
lemma congD:  $[a = b] \text{ (mod } n) \implies a \text{ mod } n = b \text{ mod } n$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma eval-coprime:
   $(b :: 'a :: euclidean-semiring-gcd) \neq 0 \implies \text{coprime } a \ b \longleftrightarrow \text{coprime } b \ (a \text{ mod } b)$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma two-power-odd-mod-12:
  assumes odd  $n > 1$ 
  shows  $[2^n = 8] \text{ (mod } (12 :: nat))$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma Legendre-3-right:
  fixes  $p :: nat$ 
  assumes  $p$ : prime  $p > 3$ 
  shows  $p \text{ mod } 12 \in \{1, 5, 7, 11\}$  and Legendre  $p \ 3 = (\text{if } p \text{ mod } 12 \in \{1, 7\}$ 
   $\text{then } 1 \text{ else } -1)$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma Legendre-3-left:
  fixes  $p :: nat$ 
  assumes  $p$ : prime  $p > 3$ 
  shows Legendre  $3 \ p = (\text{if } p \text{ mod } 12 \in \{1, 11\} \text{ then } 1 \text{ else } -1)$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma supplement2-Legendre':
  assumes prime  $p \neq 2$ 
  shows Legendre  $2 \ p = (\text{if } p \text{ mod } 8 = 1 \vee p \text{ mod } 8 = 7 \text{ then } 1 \text{ else } -1)$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma little-Fermat-nat:
  fixes  $a :: nat$ 
  assumes prime  $p \neg p \text{ dvd } a$ 
  shows  $[a^p = a] \text{ (mod } p)$ 
   $\langle \text{proof} \rangle$ 
```

```
lemma little-Fermat-int:
  fixes  $a :: int$  and  $p :: nat$ 
  assumes prime  $p \neg p \text{ dvd } a$ 
  shows  $[a^p = a] \text{ (mod } p)$ 
```

$\langle \text{proof} \rangle$

lemma *prime-dvd-choose*:

assumes $0 < k \ k < p \ \text{prime } p$

shows $p \ \text{dvd } (p \ \text{choose } k)$

$\langle \text{proof} \rangle$

lemma *prime-natD*:

assumes $\text{prime } (p :: \text{nat}) \ a \ \text{dvd } p$

shows $a = 1 \ \vee \ a = p$

$\langle \text{proof} \rangle$

lemma *not-prime-imp-ex-prod-nat*:

assumes $m > 1 \ \neg \ \text{prime } (m :: \text{nat})$

shows $\exists n \ k. \ m = n * k \wedge 1 < n \wedge n < m \wedge 1 < k \wedge k < m$

$\langle \text{proof} \rangle$

1.2 Auxiliary algebraic material

lemma (*in group*) *ord-eqI-prime-factors*:

assumes $\bigwedge p. \ p \in \text{prime-factors } n \implies x \ [\wedge] \ (n \ \text{div } p) \neq 1 \ \text{and } x \ [\wedge] \ n = 1$

assumes $x \in \text{carrier } G \ n > 0$

shows $\text{group.ord } G \ x = n$

$\langle \text{proof} \rangle$

lemma (*in monoid*) *pow-nat-eq-1-imp-unit*:

fixes $n :: \text{nat}$

assumes $x \ [\wedge] \ n = 1 \ \text{and } n > 0 \ \text{and } [\text{simp}]: x \in \text{carrier } G$

shows $x \in \text{Units } G$

$\langle \text{proof} \rangle$

lemma (*in cring*) *finsum-reindex-bij-betw*:

assumes *bij-betw* $h \ S \ T \ g \in T \rightarrow \text{carrier } R$

shows $\text{finsum } R \ (\lambda x. \ g \ (h \ x)) \ S = \text{finsum } R \ g \ T$

$\langle \text{proof} \rangle$

lemma (*in cring*) *finsum-reindex-bij-witness*:

assumes *witness*:

$\bigwedge a. \ a \in S \implies i \ (j \ a) = a$

$\bigwedge a. \ a \in S \implies j \ a \in T$

$\bigwedge b. \ b \in T \implies j \ (i \ b) = b$

$\bigwedge b. \ b \in T \implies i \ b \in S$

$\bigwedge b. \ b \in S \implies g \ b \in \text{carrier } R$

assumes *eq*:

$\bigwedge a. \ a \in S \implies h \ (j \ a) = g \ a$

shows $\text{finsum } R \ g \ S = \text{finsum } R \ h \ T$

$\langle \text{proof} \rangle$

lemma (*in cring*) *binomial*:

```

fixes  $n :: \text{nat}$ 
assumes  $[simp]: x \in \text{carrier } R \ y \in \text{carrier } R$ 
shows  $(x \oplus y) [\wedge] n = (\bigoplus_{i \in \{..n\}}. \text{add-pow } R \ (n \text{ choose } i) \ (x [\wedge] i \otimes y [\wedge] (n - i)))$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma (in cring) binomial-finite-char:
  fixes  $p :: \text{nat}$ 
  assumes  $[simp]: x \in \text{carrier } R \ y \in \text{carrier } R$  and  $\text{add-pow } R \ p \ \mathbf{1} = \mathbf{0}$  prime  $p$ 
  shows  $(x \oplus y) [\wedge] p = x [\wedge] p \oplus y [\wedge] p$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma (in ring-hom-cring) hom-add-pow-nat:
   $x \in \text{carrier } R \implies h \ (\text{add-pow } R \ (n::\text{nat}) \ x) = \text{add-pow } S \ n \ (h \ x)$ 
 $\langle \text{proof} \rangle$ 

```

end

2 The Lucas–Lehmer test

```

theory Lucas-Lehmer
imports
  Lucas-Lehmer-Auxiliary
  HOL-Algebra.Ring
  Probabilistic-Prime-Tests.Jacobi-Symbol
  Pell.Pell
begin

```

2.1 General properties of Mersenne numbers and Mersenne primes

We mostly follow the proofs given on Wikipedia [4, 3] in the following sections.

We first show some basic theorems about Mersenne numbers and Mersenne primes in general, beginning with this: Mersenne primes are the only primes of the form $a^n - 1$ for $n > 1$.

```

lemma prime-power-minus-oneD:
  fixes  $a \ n :: \text{nat}$ 
  assumes prime  $(a^n - 1)$ 
  shows  $n = 1 \vee a = 2$ 
 $\langle \text{proof} \rangle$ 

```

Next, we show that if a prime q divides a Mersenne number $2^p - 1$ with an odd prime exponent p , then q must be of the form $q = 1 + 2kp$ for some $k > 0$.

```

lemma prime-dvd-mersenneD:
  fixes  $p \ q :: \text{nat}$ 

```

```

assumes prime  $p$   $p \neq 2$  prime  $q$   $q \text{ dvd } (2^p - 1)$ 
shows  $[q = 1] \pmod{2 * p}$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma prime-dvd-mersenneD':
  fixes  $p\ q :: \text{nat}$ 
  assumes prime  $p$   $p \neq 2$  prime  $q$   $q \text{ dvd } (2^p - 1)$ 
  shows  $\exists k > 0. q = 1 + 2 * k * p$ 
 $\langle \text{proof} \rangle$ 

```

A Mersenne number is any number of the form $2^p - 1$ for a natural number p . To make things a bit more pleasant, we additionally exclude $2^2 - 1$, i.e. we require $p > 2$. It can be shown that p is then always an odd prime.

```

locale mersenne-prime =
  fixes  $p\ M :: \text{nat}$ 
  defines  $M \equiv 2^p - 1$ 
  assumes p-gt-2:  $p > 2$  and prime: prime  $M$ 
begin

```

```

lemma M-gt-6:  $M > 6$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma M-odd: odd  $M$ 
 $\langle \text{proof} \rangle$ 

```

```

theorem p-prime: prime  $p$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma p-odd: odd  $p$ 
 $\langle \text{proof} \rangle$ 

```

We now first show a few more properties of Mersenne primes regarding congruences and the Legendre symbol.

```

lemma M-cong-7-mod-12:  $[M = 7] \pmod{12}$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma Legendre-3-M: Legendre  $3\ M = -1$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma M-cong-7-mod-8:  $[M = 7] \pmod{8}$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma Legendre-2-M: Legendre  $2\ M = 1$ 
 $\langle \text{proof} \rangle$ 

```

```

lemma M-not-dvd-24:  $\neg M \text{ dvd } 24$ 
 $\langle \text{proof} \rangle$ 

```

```

end

```

2.2 The Lucas–Lehmer sequence

We now define the Lucas–Lehmer sequence $a_{n+1} = a_n^2 - 2$. The starting value we will always use is $a_0 = 4$.

primrec *gen-lucas-lehmer-sequence* :: *int* $\Rightarrow$ *nat* $\Rightarrow$ *int* **where**

gen-lucas-lehmer-sequence *a* 0 = *a*
| *gen-lucas-lehmer-sequence* *a* (Suc *n*) = *gen-lucas-lehmer-sequence* *a* $n^2 - 2$

lemma *gen-lucas-lehmer-sequence-Suc'*:

gen-lucas-lehmer-sequence *a* (Suc *n*) = *gen-lucas-lehmer-sequence* ($a^2 - 2$) *n*
⟨proof⟩

lemmas *gen-lucas-lehmer-code* [code] =

gen-lucas-lehmer-sequence.simps(1) *gen-lucas-lehmer-sequence-Suc'*

For $a_0 = 4$, the recurrence has the closed form $a_{4,n} = \omega^{2^n} + \bar{\omega}^{2^n}$ with $\omega = 2 + \sqrt{3}$ and $\bar{\omega} = 2 - \sqrt{3}$.

lemma *gen-lucas-lehmer-sequence-4-closed-form1*:

real-of-int (*gen-lucas-lehmer-sequence* 4 *n*) = $(2 + \text{sqrt } 3)^{2^n} + (2 - \text{sqrt } 3)^{2^n}$
⟨proof⟩

lemma *gen-lucas-lehmer-sequence-4-closed-form2*:

gen-lucas-lehmer-sequence 4 *n* = round $((2 + \text{sqrt } 3)^{2^n} + (2 - \text{sqrt } 3)^{2^n})$
⟨proof⟩

lemma *gen-lucas-lehmer-sequence-4-closed-form3*:

gen-lucas-lehmer-sequence 4 *n* = $\lceil (2 + \text{sqrt } 3)^{2^n} \rceil$
⟨proof⟩

2.3 The ring $\mathbb{Z}[\sqrt{3}]$

To relate this sequence to Mersenne primes, we now first need to define the ring $\mathbb{Z}[\sqrt{3}]$, which is a subring of $\mathbb{R}$. This ring can be seen as the lattice on $\mathbb{R}$ that is freely generated by 1 and $\sqrt{3}$.

It is, however, more convenient to explicitly describe it as a ring structure over the set $\mathbb{Z} \times \mathbb{Z}$ with a corresponding injective homomorphism $\mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{R}$.

definition *lucas-lehmer-add'* :: *int* $\times$ *int* $\Rightarrow$ *int* $\times$ *int* $\Rightarrow$ *int* $\times$ *int* **where**

lucas-lehmer-add' = $(\lambda(a,b) (c,d). (a + c, b + d))$

definition *lucas-lehmer-mult'* :: *int* $\times$ *int* $\Rightarrow$ *int* $\times$ *int* $\Rightarrow$ *int* $\times$ *int* **where**

lucas-lehmer-mult' = $(\lambda(a,b) (c,d). (a * c + 3 * b * d, a * d + b * c))$

definition *lucas-lehmer-ring* :: (*int* $\times$ *int*) *ring* **where**

lucas-lehmer-ring =
 (| *carrier* = UNIV,
 monoid.mult = *lucas-lehmer-mult'*,

```

one = (1, 0),
ring.zero = (0, 0),
add = lucas-lehmer-add'

```

lemma *carrier-lucas-lehmer-ring* [simp]: *carrier lucas-lehmer-ring = UNIV*
 ⟨proof⟩

lemma *cring-lucas-lehmer-ring* [intro]: *cring (lucas-lehmer-ring)*
 ⟨proof⟩

2.4 The ring $(\mathbb{Z}/m\mathbb{Z})[\sqrt{3}]$

We shall also need the ring $(\mathbb{Z}/m\mathbb{Z})[\sqrt{3}]$, which is obtained from $\mathbb{Z}[\sqrt{3}]$ by reducing each component separately modulo m . This essentially identifies any two points that are a multiple of m apart and then all those that are a multiple of $m\sqrt{3}$ apart.

definition *lucas-lehmer-mult* :: *nat* $\Rightarrow$ *nat* $\times$ *nat* $\Rightarrow$ *nat* $\times$ *nat* $\Rightarrow$ *nat* $\times$ *nat* **where**
lucas-lehmer-mult $m = (\lambda(a,b) (c,d). ((a * c + 3 * b * d) \bmod m, (a * d + b * c) \bmod m))$

definition *lucas-lehmer-add* :: *nat* $\Rightarrow$ *nat* $\times$ *nat* $\Rightarrow$ *nat* $\times$ *nat* $\Rightarrow$ *nat* $\times$ *nat* **where**
lucas-lehmer-add $m = (\lambda(a,b) (c,d). ((a + c) \bmod m, (b + d) \bmod m))$

definition *lucas-lehmer-ring-mod* :: *nat* $\Rightarrow$ (*nat* $\times$ *nat*) *ring* **where**
lucas-lehmer-ring-mod $m =$
 ⟨carrier = $\{..<m\} \times \{..<m\}$,
 monoid.mult = *lucas-lehmer-mult* m ,
 one = (1, 0),
 ring.zero = (0, 0),
 add = *lucas-lehmer-add* m ⟩

lemma *lucas-lehmer-add-in-carrier*: $m > 0 \implies \text{lucas-lehmer-add } m \ x \ y \in \{..<m\} \times \{..<m\}$
 ⟨proof⟩

lemma *lucas-lehmer-mult-in-carrier*: $m > 0 \implies \text{lucas-lehmer-mult } m \ x \ y \in \{..<m\} \times \{..<m\}$
 ⟨proof⟩

lemma *lucas-lehmer-add-cong*:
 [fst (*lucas-lehmer-add* $m \ x \ y$) = fst x + fst y] (mod m)
 [snd (*lucas-lehmer-add* $m \ x \ y$) = snd x + snd y] (mod m)
 ⟨proof⟩

lemma *lucas-lehmer-mult-cong*:
 [fst (*lucas-lehmer-mult* $m \ x \ y$) = fst x * fst y + 3 * snd x * snd y] (mod m)
 [snd (*lucas-lehmer-mult* $m \ x \ y$) = fst x * snd y + snd x * fst y] (mod m)

$\langle \text{proof} \rangle$

lemma *lucas-lehmer-add-neutral* [simp]:
 assumes $\text{fst } x < m \text{ and } x < m$
 shows $\text{lucas-lehmer-add } m \ (0, 0) \ x = x$
 and $\text{lucas-lehmer-add } m \ x \ (0, 0) = x$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-mult-neutral* [simp]:
 assumes $\text{fst } x < m \text{ and } x < m$
 shows $\text{lucas-lehmer-mult } m \ (\text{Suc } 0, 0) \ x = x$
 and $\text{lucas-lehmer-mult } m \ x \ (\text{Suc } 0, 0) = x$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-add-commute*: $\text{lucas-lehmer-add } m \ x \ y = \text{lucas-lehmer-add } m \ y \ x$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-mult-commute*: $\text{lucas-lehmer-mult } m \ x \ y = \text{lucas-lehmer-mult } m \ y \ x$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-add-assoc*:
 assumes $m: m > 0$
 shows $\text{lucas-lehmer-add } m \ x \ (\text{lucas-lehmer-add } m \ y \ z) =$
 $\text{lucas-lehmer-add } m \ (\text{lucas-lehmer-add } m \ x \ y) \ z$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-mult-assoc*:
 assumes $m: m > 0$
 shows $\text{lucas-lehmer-mult } m \ x \ (\text{lucas-lehmer-mult } m \ y \ z) =$
 $\text{lucas-lehmer-mult } m \ (\text{lucas-lehmer-mult } m \ x \ y) \ z$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-distrib-right*:
 assumes $m: m > 1$
 shows $\text{lucas-lehmer-mult } m \ (\text{lucas-lehmer-add } m \ x \ y) \ z =$
 $\text{lucas-lehmer-add } m \ (\text{lucas-lehmer-mult } m \ x \ z) \ (\text{lucas-lehmer-mult } m \ y \ z)$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-distrib-left*:
 assumes $m > 1$
 shows $\text{lucas-lehmer-mult } m \ z \ (\text{lucas-lehmer-add } m \ x \ y) =$
 $\text{lucas-lehmer-add } m \ (\text{lucas-lehmer-mult } m \ z \ x) \ (\text{lucas-lehmer-mult } m \ z \ y)$
 $\langle \text{proof} \rangle$

lemma *cring-lucas-lehmer-ring-mod* [intro]:
 assumes $m > 1$
 shows $\text{cring } (\text{lucas-lehmer-ring-mod } m)$

<proof>

Since 0 is clearly not a unit in the ring and its carrier has size m^2 , the number of units is strictly less than m^2 .

lemma *card-lucas-lehmer-Units:*

assumes $m > 1$

shows $\text{card } (\text{Units } (\text{lucas-lehmer-ring-mod } m)) < m^2$

<proof>

Consider now the case of a prime modulus m : Since $\mathbb{Z}/m\mathbb{Z} = \text{GF}(m)$ is a field, any element of $\mathbb{Z}/m\mathbb{Z}$ is a unit in $(\mathbb{Z}/m\mathbb{Z})[\sqrt{3}]$.

lemma *int-in-Units-lucas-lehmer-ring-mod:*

assumes *prime* p

assumes $x > 0$ $x < p$

shows $(x, 0) \in \text{Units } (\text{lucas-lehmer-ring-mod } p)$

<proof>

2.5 $\mathbb{Z}[\sqrt{3}]$ as a subring of $\mathbb{R}$

We now define the homomorphism from $\mathbb{Z}[\sqrt{3}]$ into the reals:

definition *lucas-lehmer-to-real* :: $\text{int} \times \text{int} \Rightarrow \text{real}$ **where**

lucas-lehmer-to-real = $(\lambda(a,b). \text{real-of-int } a + \text{real-of-int } b * \text{sqrt } 3)$

context

begin

interpretation *cring lucas-lehmer-ring* *<proof>*

lemma *minus-lucas-lehmer-ring*: $\ominus_{\text{lucas-lehmer-ring}} x = (\text{case } x \text{ of } (a, b) \Rightarrow (-a, -b))$

<proof>

lemma *lucas-lehmer-to-real-simps1*:

lucas-lehmer-to-real $(a, b) = \text{of-int } a + \text{of-int } b * \text{sqrt } 3$

lucas-lehmer-to-real $(x \oplus_{\text{lucas-lehmer-ring}} y) =$

lucas-lehmer-to-real $x + \text{lucas-lehmer-to-real } y$

lucas-lehmer-to-real $(x \otimes_{\text{lucas-lehmer-ring}} y) =$

lucas-lehmer-to-real $x * \text{lucas-lehmer-to-real } y$

lucas-lehmer-to-real $(\ominus_{\text{lucas-lehmer-ring}} x) = -\text{lucas-lehmer-to-real } x$

lucas-lehmer-to-real $(\mathbf{0}_{\text{lucas-lehmer-ring}}) = 0$

lucas-lehmer-to-real $(\mathbf{1}_{\text{lucas-lehmer-ring}}) = 1$

<proof>

lemma *lucas-lehmer-to-add-pow-nat*:

lucas-lehmer-to-real $([n] \cdot_{\text{lucas-lehmer-ring}} x) = \text{of-nat } n * \text{lucas-lehmer-to-real } x$

<proof>

lemma *lucas-lehmer-to-add-pow-int*:

lucas-lehmer-to-real ($[n] \cdot \text{lucas-lehmer-ring } x$) = *of-int* $n * \text{lucas-lehmer-to-real } x$
 <proof>

lemma *lucas-lehmer-to-real-power*:

lucas-lehmer-to-real ($x \text{ } [\wedge]_{\text{lucas-lehmer-ring}} (n :: \text{nat})) = \text{lucas-lehmer-to-real } x \wedge^n$
 <proof>

lemmas *lucas-lehmer-to-real-simps* =

lucas-lehmer-to-real-simps1 *lucas-lehmer-to-real-power*
lucas-lehmer-to-add-pow-nat *lucas-lehmer-to-add-pow-int*

end

lemma *lucas-lehmer-to-real-inj*: *inj* *lucas-lehmer-to-real*

<proof>

2.6 The canonical homomorphism $\mathbb{Z}[\sqrt{3}] \rightarrow (\mathbb{Z}/m\mathbb{Z})[\sqrt{3}]$

Next, we show that reduction modulo m is indeed a homomorphism.

definition *lucas-lehmer-hom* :: $\text{nat} \Rightarrow (\text{int} \times \text{int}) \Rightarrow (\text{nat} \times \text{nat})$ **where**

lucas-lehmer-hom $m = (\lambda(x,y). (\text{nat } (x \bmod m), \text{nat } (y \bmod m)))$

lemma *lucas-lehmer-hom-cong*:

$[fst\ x = fst\ y] \ (mod\ int\ m) \implies [snd\ x = snd\ y] \ (mod\ int\ m) \implies$
lucas-lehmer-hom $m\ x = \text{lucas-lehmer-hom } m\ y$
 <proof>

lemma *lucas-lehmer-hom-cong'*:

$[a = b] \ (mod\ int\ m) \implies [c = d] \ (mod\ int\ m) \implies$
lucas-lehmer-hom $m\ (a, c) = \text{lucas-lehmer-hom } m\ (b, d)$
 <proof>

context

fixes $m :: \text{nat}$

assumes $m: m > 1$

begin

lemma *lucas-lehmer-hom-in-carrier*: *lucas-lehmer-hom* $m\ x \in \{..<m\} \times \{..<m\}$

<proof>

lemma *lucas-lehmer-hom-add*:

lucas-lehmer-hom $m\ (\text{lucas-lehmer-add}'\ x\ y) =$
lucas-lehmer-add $m\ (\text{lucas-lehmer-hom } m\ x)\ (\text{lucas-lehmer-hom } m\ y)$
 <proof>

lemma *lucas-lehmer-hom-mult*:

lucas-lehmer-hom $m\ (\text{lucas-lehmer-mult}'\ x\ y) =$
lucas-lehmer-mult $m\ (\text{lucas-lehmer-hom } m\ x)\ (\text{lucas-lehmer-hom } m\ y)$

<proof>

lemma *lucas-lehmer-hom-1* [*simp*]: *lucas-lehmer-hom* m $(1, 0) = (1, 0)$
<proof>

lemma *ring-hom-lucas-lehmer-hom*:
lucas-lehmer-hom $m \in \text{ring-hom } \text{lucas-lehmer-ring } (\text{lucas-lehmer-ring-mod } m)$
<proof>

end

2.7 Correctness of the Lucas–Lehmer test

In this section, we will prove that the Lucas–Lehmer test is both a necessary and sufficient condition for the primality of a Mersenne number of the form $2^p - 1$ for an odd prime p . The proof that shall be given here is rather explicit and heavily draws from the Wikipedia article on the Lucas–Lehmer test [3].

A shorter and more high-level proof of a more general statement can be obtained using more theory on finite fields (in particular the field $\text{GF}(q^2)$) (cf. e. g. Rödseth [2]).

definition *lucas-lehmer-test* **where**
lucas-lehmer-test $p = (p > 2 \wedge$
 $(2 \wedge p - 1) \text{ dvd } \text{gen-lucas-lehmer-sequence } 4 \ (p - 2))$

We can now prove that any Mersenne number $2^p - 1$ for p prime that passes the Lucas–Lehmer test is prime. We follow the simple argument given by Bruce [1], which is also given on Wikipedia [3].

theorem *lucas-lehmer-sufficient*:
assumes *prime* p *odd* p
assumes $(2 \wedge p - 1) \text{ dvd } \text{gen-lucas-lehmer-sequence } 4 \ (p - 2)$
shows *prime* $(2 \wedge p - 1 :: \text{nat})$
<proof>

Next, we show that any Mersenne prime passes the Lucas–Lehmer test. We again follow the rather explicit proof outlined on Wikipedia [3], which is a simplified (but less general and less abstract) version of the proof by Rödseth [2].

theorem (**in** *mersenne-prime*) *lucas-lehmer-necessary*:
 $(2 \wedge p - 1) \text{ dvd } \text{gen-lucas-lehmer-sequence } 4 \ (p - 2)$
<proof>

corollary *lucas-lehmer-correct*:
 $\text{prime } (2 \wedge p - 1 :: \text{nat}) \longleftrightarrow$
 $\text{prime } p \wedge (p = 2 \vee (2 \wedge p - 1) \text{ dvd } \text{gen-lucas-lehmer-sequence } 4 \ (p - 2))$
<proof>

corollary *lucas-lehmer-correct'*:

$\text{prime } (2 \wedge p - 1 :: \text{nat}) \longleftrightarrow \text{prime } p \wedge (p = 2 \vee \text{lucas-lehmer-test } p)$
 $\langle \text{proof} \rangle$

2.8 A first executable version Lucas–Lehmer test

The following is an implementation of the Lucas–Lehmer test using modular arithmetic on the integers. This is not the most efficient implementation – the modular arithmetic can be replaced by much cheaper bitwise operations, and we will do that in the next section.

primrec *gen-lucas-lehmer-sequence'* :: $\text{int} \Rightarrow \text{int} \Rightarrow \text{nat} \Rightarrow \text{int}$ **where**

$\text{gen-lucas-lehmer-sequence}' \ m \ a \ 0 = a$
 $|\ \text{gen-lucas-lehmer-sequence}' \ m \ a \ (\text{Suc } n) = \text{gen-lucas-lehmer-sequence}' \ m \ ((a \wedge 2 - 2) \bmod m) \ n$

lemma *gen-lucas-lehmer-sequence'-Suc'*:

$\text{gen-lucas-lehmer-sequence}' \ m \ a \ (\text{Suc } n) = (\text{gen-lucas-lehmer-sequence}' \ m \ a \ n \wedge 2 - 2) \bmod m$
 $\langle \text{proof} \rangle$

lemma *gen-lucas-lehmer-sequence'-correct*:

assumes $a \in \{0..<m\}$
shows $\text{gen-lucas-lehmer-sequence}' \ m \ a \ n = \text{gen-lucas-lehmer-sequence } a \ n \bmod m$
 $\langle \text{proof} \rangle$

lemma *lucas-lehmer-test-code-arithmetic* [code]:

$\text{lucas-lehmer-test } p = (p > 2 \wedge$
 $\text{gen-lucas-lehmer-sequence}' (2 \wedge p - 1) \ 4 \ (p - 2) = 0)$
 $\langle \text{proof} \rangle$

lemma *mersenne-prime-iff*: $\text{mersenne-prime } p \longleftrightarrow p > 2 \wedge \text{prime } (2 \wedge p - 1 :: \text{nat})$

$\langle \text{proof} \rangle$

lemma *mersenne-prime-code* [code]:

$\text{mersenne-prime } p \longleftrightarrow \text{prime } p \wedge \text{lucas-lehmer-test } p$
 $\langle \text{proof} \rangle$

end

3 Efficient code for testing Mersenne primes

theory *Lucas-Lehmer-Code*

imports

Lucas-Lehmer

HOL-Library.Code-Target-Numeral

Native-Word.Code-Target-Int-Bit
begin

3.1 Efficient computation of remainders modulo a Mersenne number

We have $k = k \bmod 2^n + k \operatorname{div} 2^n \pmod{(2^n - 1)}$, and $k \bmod 2^n = k \& (2^n - 1)$ and $k \operatorname{div} 2^n = k \gg n$. Therefore, we can reduce k modulo $2^n - 1$ using only bitwise operations, addition, and bit shifts.

lemma *cong-mersenne-number-int*:

fixes $k :: \text{int}$
shows $[k \bmod 2^n + k \operatorname{div} 2^n = k] \pmod{(2^n - 1)}$
 $\langle \text{proof} \rangle$

We encapsulate a single reduction step in the following operation. Note, however, that the result is not, in general, the same as $k \bmod (2^n - 1)$. Multiple reductions might be required in order to reduce it below 2^n , and a multiple of $2^n - 1$ can be reduced to $2^n - 1$, which is invariant to further reduction steps.

definition *mersenne-mod* $:: \text{int} \Rightarrow \text{nat} \Rightarrow \text{int}$ **where**

mersenne-mod $k\ n = k \bmod 2^n + k \operatorname{div} 2^n$

lemma *mersenne-mod-code* $[code]$:

mersenne-mod $k\ n = \text{take-bit } n\ k + \text{drop-bit } n\ k$
 $\langle \text{proof} \rangle$

lemma *cong-mersenne-mod*: $[mersenne-mod\ k\ n = k] \pmod{(2^n - 1)}$

$\langle \text{proof} \rangle$

lemma *mersenne-mod-nonneg* $[simp]$: $k \geq 0 \implies mersenne-mod\ k\ n \geq 0$

$\langle \text{proof} \rangle$

lemma *mersenne-mod-less*:

assumes $k \leq 2^m\ m \geq n$
shows $mersenne-mod\ k\ n < 2^n + 2^{m-n}$
 $\langle \text{proof} \rangle$

lemma *mersenne-mod-less'*:

assumes $k \leq 5 * 2^n$
shows $mersenne-mod\ k\ n < 2^n + 5$
 $\langle \text{proof} \rangle$

It turns out that for our use case, a single reduction is not enough to reduce the number in question enough (or at least I was unable to prove that it is). We therefore perform two reduction steps, which is enough to guarantee that our numbers are below $2^n + 4$ before and after every step in the Lucas–Lehmer sequence.

Whether one or two reductions are performed is not very important anyway, since the dominant step is the squaring anyway.

definition *mersenne-mod2* :: *int* $\Rightarrow$ *nat* $\Rightarrow$ *int* **where**
mersenne-mod2 *k n* = *mersenne-mod* (*mersenne-mod* *k n*) *n*

lemma *cong-mersenne-mod2*: [*mersenne-mod2* *k n* = *k*] (*mod* ($2^n - 1$))
 <proof>

lemma *mersenne-mod2-nonneg* [*simp*]: $k \geq 0 \implies \text{mersenne-mod2 } k \ n \geq 0$
 <proof>

lemma *mersenne-mod2-less*:
assumes $n > 2$ **and** $k \leq 2^{(2 * n + 2)}$
shows $\text{mersenne-mod2 } k \ n < 2^n + 5$
 <proof>

Since we subtract 2 at one point, the intermediate results can become negative. This is not a problem since our reduction modulo $2^p - 1$ happens to make them positive again immediately.

lemma *mersenne-mod-nonneg-strong*:
 <*mersenne-mod* *a p* ≥ 0 > **if** < $-(2^p) + 1 < a$ >
 <proof>

lemma *mersenne-mod2-nonneg-strong*:
assumes $a > -(2^p) + 1$
shows $\text{mersenne-mod2 } a \ p \geq 0$
 <proof>

3.2 Efficient code for the Lucas–Lehmer sequence

primrec *gen-lucas-lehmer-sequence''* :: *nat* $\Rightarrow$ *int* $\Rightarrow$ *nat* $\Rightarrow$ *int* **where**
gen-lucas-lehmer-sequence'' *p a* 0 = *a*
 | *gen-lucas-lehmer-sequence''* *p a* (*Suc n*) =
 gen-lucas-lehmer-sequence'' *p* (*mersenne-mod2* ($a^2 - 2$) *p*) *n*

lemma *gen-lucas-lehmer-sequence''-correct*:
assumes [*a* = *a'*] (*mod* ($2^p - 1$))
shows [*gen-lucas-lehmer-sequence''* *p a n* = *gen-lucas-lehmer-sequence* *a' n*]
 (*mod* ($2^p - 1$))
 <proof>

lemma *gen-lucas-lehmer-sequence''-bounds*:
assumes $a \geq 0$ $a < 2^p + 5$ $p > 2$
shows $\text{gen-lucas-lehmer-sequence'' } p \ a \ n \in \{0..<2^p + 5\}$
 <proof>

3.3 Code for the Lucas–Lehmer test

lemmas [*code del*] = *lucas-lehmer-test-code-arithmetic*

```

lemma lucas-lehmer-test-code [code]:
  lucas-lehmer-test p =
    (2 < p ∧ (let x = gen-lucas-lehmer-sequence'' p 4 (p - 2) in x = 0 ∨ x =
      (push-bit p 1) - 1))
  ⟨proof⟩

```

3.4 Examples

Note that for some reason, the clever bit-arithmetic version of the Lucas–Lehmer test is actually much slower than the one using integer arithmetic when using PolyML, and even more so when using the built-in evaluator in Isabelle (which also uses PolyML with a slightly different setup).

I do not quite know why this is the case, but it is likely because of inefficient implementations of bit arithmetic operations in PolyML and/or the code generator setup for it.

When running with GHC, the bit-arithmetic version is *much* faster.

```

value filter mersenne-prime [0..100]

```

```

lemma prime (2521 - 1 :: nat)
  ⟨proof⟩

```

```

lemma prime (24253 - 1 :: nat)
  ⟨proof⟩

```

```

end

```

References

- [1] J. W. Bruce. A really trivial proof of the Lucas-Lehmer test. *The American Mathematical Monthly*, 100(4):370–371, 1993.
- [2] Ö. J. Rödseth. A note on primality tests for $n = h \cdot 2^n - 1$. *BIT Numerical Mathematics*, 34(3):451–454, Sep 1994.
- [3] Wikipedia contributors. Lucas–Lehmer primality test — Wikipedia, the free encyclopedia, 2020. [Online; accessed 17 Jan 2020].
- [4] Wikipedia contributors. Mersenne prime — Wikipedia, the free encyclopedia, 2020. [Online; accessed 17 Jan 2020].