Probabilistic Primality Testing

Daniel Stüwe and Manuel Eberl

October 13, 2025

Abstract

The most efficient known primality tests are *probabilistic* in the sense that they use randomness and may, with some probability, mistakenly classify a composite number as prime – but never a prime number as composite. Examples of this are the Miller–Rabin test, the Solovay–Strassen test, and (in most cases) Fermat's test.

This entry defines these three tests and proves their correctness. It also develops some of the number-theoretic foundations, such as Carmichael numbers and the Jacobi symbol with an efficient executable algorithm to compute it.

Contents

1 Additional Facts about the Legendre Symbol

theory *Legendre-Symbol*
imports
  HOL−Number-Theory.Number-Theory
begin

lemma *basic-cong*[*simp*]:
  fixes $p :: int$
  assumes $2 < p$
  shows $[-1 \neq 1]\ (mod\ p)$
  $[\ 1 \neq -1]\ (mod\ p)$
  $[\ 0 \neq 1]\ (mod\ p)$
  $[\ 1 \neq 0]\ (mod\ p)$
  $[\ 0 \neq -1]\ (mod\ p)$
  $[-1 \neq 0]\ (mod\ p)$
  ⟨*proof*⟩

lemma [*simp*]: $0 < n \Longrightarrow (a\ mod\ 2)\ \hat{}\ n = a\ mod\ 2$ **for** $n :: nat$ **and** $a :: int$
⟨*proof*⟩

lemma *Legendre-in-cong-eq*:
  fixes $p :: int$
  assumes $p > 2$ **and** $b \in \{-1,0,1\}$
  shows $[Legendre\ a\ m = b]\ (mod\ p) \longleftrightarrow Legendre\ a\ m = b$
  ⟨*proof*⟩

lemma *Legendre-p-eq-2*[*simp*]: $Legendre\ a\ 2 = a\ mod\ 2$
  ⟨*proof*⟩

lemma *Legendre-p-eq-1*[*simp*]: $Legendre\ a\ 1 = 0$ ⟨*proof*⟩

lemma *euler-criterion-int*:
  assumes *prime p* **and** $2 < p$
  shows $[Legendre\ a\ p = a\hat{}((nat\ p-1)\ div\ 2)]\ (mod\ p)$
  ⟨*proof*⟩

lemma *QuadRes-neg*[*simp*]: $QuadRes\ (-p)\ a = QuadRes\ p\ a$ ⟨*proof*⟩

lemma *Legendre-neg*[*simp*]: $Legendre\ a\ (-p) = Legendre\ a\ p$ ⟨*proof*⟩

lemma *Legendre-mult*[*simp*]:
  assumes *prime p*
  shows $Legendre\ (a*b)\ p = Legendre\ a\ p * Legendre\ b\ p$
⟨*proof*⟩

lemma *QuadRes-mod*[*simp*]: $p\ dvd\ n \Longrightarrow QuadRes\ p\ (a\ mod\ n) = QuadRes\ p\ a$
  ⟨*proof*⟩

lemma *Legendre-mod*[*simp*]: $p\ dvd\ n \Longrightarrow Legendre\ (a\ mod\ n)\ p = Legendre\ a\ p$
⟨*proof*⟩

lemma *two-cong-0-iff*: $[2 = 0]\ (mod\ p) \longleftrightarrow p = 1 \vee p = 2$ **for** $p :: nat$
⟨*proof*⟩

lemma *two-cong-0-iff-nat*: $[2 = 0]\ (mod\ int\ p) \longleftrightarrow p = 1 \vee p = 2$
⟨*proof*⟩

lemma *two-cong-0-iff-int*: $p > 0 \Longrightarrow [2 = 0]\ (mod\ p) \longleftrightarrow p = 1 \vee p = 2$ **for** p $:: int$
⟨*proof*⟩

lemma *QuadRes-2-2* [*simp*, *intro*]: *QuadRes 2 2*
⟨*proof*⟩

lemma *Suc-mod-eq*[*simp*]: $[Suc\ a = Suc\ b]\ (mod\ 2) = [a = b]\ (mod\ 2)$
⟨*proof*⟩

lemma *div-cancel-aux*: $c\ dvd\ a \Longrightarrow (d + a * b)\ div\ c = (d\ div\ c) + a\ div\ c * b$ **for** $a\ b\ c :: nat$
⟨*proof*⟩

corollary *div-cancel-Suc*: $c\ dvd\ a \Longrightarrow 1 < c \Longrightarrow Suc\ (a * b)\ div\ c = a\ div\ c * b$
⟨*proof*⟩

lemma *cong-aux-eq-1*: $odd\ p \Longrightarrow [(p - 1)\ div\ 2 - p\ div\ 4 = (p\hat{}2 - 1)\ div\ 8]$ $(mod\ 2)$ **for** $p :: nat$
⟨*proof*⟩

lemma *cong-2-pow*[*intro*]: $(-1 :: int)\hat{}a = (-1)\hat{}b$ **if** $[a = b]\ (mod\ 2)$ **for** $a\ b ::$ nat
⟨*proof*⟩

lemma *card-Int*: $card\ (A \cap B) = card\ A - card\ (A - B)$ **if** *finite A*
⟨*proof*⟩

Proofs are inspired by [3].

theorem *supplement2-Legendre*:
fixes $p :: int$
assumes $p > 2$ *prime p*
shows $Legendre\ 2\ p = (-1)\ \hat{}\ (((nat\ p)\hat{}2 - 1)\ div\ 8)$
⟨*proof*⟩

theorem *supplement1-Legendre*:
$prime\ p \Longrightarrow 2 < p \Longrightarrow Legendre\ (-1)\ p = (-1)\hat{}((p-1)\ div\ 2)$
⟨*proof*⟩

lemma *QuadRes-1-right* [*intro*, *simp*]: *QuadRes p 1*

⟨*proof*⟩

lemma *Legendre-1-left* [*simp*]: *prime p* $\Longrightarrow$ *Legendre 1 p* = *1*
⟨*proof*⟩

lemma *cong-eq-0-not-coprime*: *prime p* $\Longrightarrow$ [*a* = *0*] (*mod p*) $\Longrightarrow$ ¬*coprime a p* **for** *a p* :: *int*
⟨*proof*⟩

lemma *not-coprime-cong-eq-0*: *prime p* $\Longrightarrow$ ¬*coprime a p* $\Longrightarrow$ [*a* = *0*] (*mod p*) **for** *a p* :: *int*
⟨*proof*⟩

lemma *prime-cong-eq-0-iff*: *prime p* $\Longrightarrow$ [*a* = *0*] (*mod p*) $\longleftrightarrow$ ¬*coprime a p* **for** *a p* :: *int*
⟨*proof*⟩

lemma *Legendre-eq-0-iff* [*simp*]: *prime p* $\Longrightarrow$ *Legendre a p* = *0* $\longleftrightarrow$ ¬*coprime a p*
⟨*proof*⟩

lemma *Legendre-prod-mset* [*simp*]: *prime p* $\Longrightarrow$ *Legendre* (*prod-mset M*) *p* = ($\prod$ *q*∈#*M*. *Legendre q p*)
⟨*proof*⟩

lemma *Legendre-0-eq-0*[*simp*]: *Legendre 0 p* = *0* ⟨*proof*⟩

lemma *Legendre-values*: *Legendre p q* ∈ {*1*, − *1*, *0*}
⟨*proof*⟩

end

2 Auxiliary Material

theory *Algebraic-Auxiliaries*
imports
HOL−Algebra.Algebra
HOL−Computational-Algebra.Squarefree
HOL−Number-Theory.Number-Theory
begin

hide-const (**open**) *Divisibility.prime*

lemma *sum-of-bool-eq-card*:
assumes *finite S*
shows ($\sum$ *a* ∈ *S*. *of-bool* (*P a*)) = *real* (*card* {*a* ∈ *S* . *P a* })
⟨*proof*⟩

lemma *mod-natE*:
fixes *a n b* :: *nat*

assumes *a mod n* = *b*
shows ∃ *l*. *a* = *n* ∗ *l* + *b*
⟨*proof*⟩

lemma (**in** *group*) *r-coset-is-image*: *H* #> *a* = (λ *x*. *x* ⊗ *a*) ' *H*
⟨*proof*⟩

lemma (**in** *group*) *FactGroup-order*:
assumes *subgroup H G finite H*
shows *order G* = *order* (*G Mod H*) ∗ *card H*
⟨*proof*⟩

corollary (**in** *group*) *FactGroup-order-div*:
assumes *subgroup H G finite H*
shows *order* (*G Mod H*) = *order G div card H*
⟨*proof*⟩

lemma *group-hom-imp-group-hom-image*:
assumes *group-hom G G h*
shows *group-hom G* (*G*⦇*carrier* := *h* ' *carrier G*⦈) *h*
⟨*proof*⟩

theorem *homomorphism-thm*:
assumes *group-hom G G h*
shows *G Mod kernel G* (*G*⦇*carrier* := *h* ' *carrier G*⦈) *h* ≅ *G* ⦇*carrier* := *h* ' *carrier G*⦈
⟨*proof*⟩

lemma *is-iso-imp-same-card*:
assumes *H* ≅ *G*
shows *order H* = *order G*
⟨*proof*⟩

corollary *homomorphism-thm-order*:
assumes *group-hom G G h*
shows *order* (*G*⦇*carrier* := *h* ' *carrier G*⦈) ∗ *card* (*kernel G* (*G*⦇*carrier* := *h* ' *carrier G*⦈) *h*) = *order G*
⟨*proof*⟩

lemma (**in** *group-hom*) *kernel-subset*: *kernel G H h* ⊆ *carrier G*
⟨*proof*⟩

lemma (**in** *group*) *proper-subgroup-imp-bound-on-card*:
assumes *H* ⊂ *carrier G subgroup H G finite* (*carrier G*)
shows *card H* ≤ *order G div 2*
⟨*proof*⟩

lemma *cong-exp-trans*[*trans*]:
[*a* ^ *b* = *c*] (*mod n*) ⟹ [*a* = *d*] (*mod n*) ⟹ [*d* ^ *b* = *c*] (*mod n*)

$[c = a \mathbin{\char`\^} b] \ (mod\ n) \Longrightarrow [a = d] \ (mod\ n) \Longrightarrow [c = d \mathbin{\char`\^} b] \ (mod\ n)$
⟨proof⟩

lemma *cong-exp-mod*[*simp*]:
$[(a\ mod\ n) \ \hat{}\ b = c] \ (mod\ n) \longleftrightarrow [a \ \hat{}\ b = c] \ (mod\ n)$
$[c = (a\ mod\ n) \ \hat{}\ b] \ (mod\ n) \longleftrightarrow [c = a \ \hat{}\ b] \ (mod\ n)$
⟨proof⟩

lemma *cong-mult-mod*[*simp*]:
$[(a\ mod\ n) * b = c] \ (mod\ n) \longleftrightarrow [a * b = c] \ (mod\ n)$
$[a * (b\ mod\ n) = c] \ (mod\ n) \longleftrightarrow [a * b = c] \ (mod\ n)$
⟨proof⟩

lemma *cong-add-mod*[*simp*]:
$[(a\ mod\ n) + b = c] \ (mod\ n) \longleftrightarrow [a + b = c] \ (mod\ n)$
$[a + (b\ mod\ n) = c] \ (mod\ n) \longleftrightarrow [a + b = c] \ (mod\ n)$
$[\sum i{\in}A.\ f\ i\ mod\ n = c] \ (mod\ n) \longleftrightarrow [\sum i{\in}A.\ f\ i = c] \ (mod\ n)$
⟨proof⟩

lemma *cong-add-trans*[*trans*]:
$[a = b + x] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [a = b + y] \ (mod\ n)$
$[a = x + b] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [a = y + b] \ (mod\ n)$
$[b + x = a] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [b + y = a] \ (mod\ n)$
$[x + b = a] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [y + b = a] \ (mod\ n)$
⟨proof⟩

lemma *cong-mult-trans*[*trans*]:
$[a = b * x] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [a = b * y] \ (mod\ n)$
$[a = x * b] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [a = y * b] \ (mod\ n)$
$[b * x = a] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [b * y = a] \ (mod\ n)$
$[x * b = a] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [y * b = a] \ (mod\ n)$
⟨proof⟩

lemma *cong-diff-trans*[*trans*]:
$[a = b - x] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [a = b - y] \ (mod\ n)$
$[a = x - b] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [a = y - b] \ (mod\ n)$
$[b - x = a] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [b - y = a] \ (mod\ n)$
$[x - b = a] \ (mod\ n) \Longrightarrow [x = y] \ (mod\ n) \Longrightarrow [y - b = a] \ (mod\ n)$
for $a :: 'a :: \{unique\text{-}euclidean\text{-}semiring,\ euclidean\text{-}ring\text{-}cancel\}$
⟨proof⟩

lemma *eq-imp-eq-mod-int*: $a = b \Longrightarrow [a = b] \ (mod\ m)$ **for** $a\ b :: int$ ⟨proof⟩
lemma *eq-imp-eq-mod-nat*: $a = b \Longrightarrow [a = b] \ (mod\ m)$ **for** $a\ b :: nat$ ⟨proof⟩

lemma *cong-pow-I*: $a = b \Longrightarrow [x \hat{}a = x \hat{}b](mod\ n)$ ⟨proof⟩

lemma *gre1I*: $(n = 0 \Longrightarrow False) \Longrightarrow (1 :: nat) \leq n$
⟨proof⟩

lemma *gre1I-nat*: $(n = 0 \Longrightarrow False) \Longrightarrow (Suc\ 0 :: nat) \leq n$
⟨*proof*⟩

lemma *totient-less-not-prime*:
assumes $\neg$ *prime n* $1 < n$
shows *totient n* $< n - 1$
⟨*proof*⟩

lemma *power2-diff-nat*: $x \geq y \Longrightarrow (x - y)^2 = x^2 + y^2 - 2 * x * y$ **for** $x\ y :: nat$
⟨*proof*⟩

lemma *square-inequality*: $1 < n \Longrightarrow (n + n) \leq (n * n)$ **for** $n :: nat$
⟨*proof*⟩

lemma *square-one-cong-one*:
assumes $[x = 1](mod\ n)$
shows $[x\hat{}2 = 1](mod\ n)$
⟨*proof*⟩

lemma *cong-square-alt-int*:
$prime\ p \Longrightarrow [a * a = 1]\ (mod\ p) \Longrightarrow [a = 1]\ (mod\ p) \vee [a = p - 1]\ (mod\ p)$
for $a\ p :: 'a :: \{normalization\text{-}semidom, linordered\text{-}idom, unique\text{-}euclidean\text{-}ring\}$
⟨*proof*⟩

lemma *cong-square-alt*:
$prime\ p \Longrightarrow [a * a = 1]\ (mod\ p) \Longrightarrow [a = 1]\ (mod\ p) \vee [a = p - 1]\ (mod\ p)$
for $a\ p :: nat$
⟨*proof*⟩

lemma *square-minus-one-cong-one*:
fixes $n\ x :: nat$
assumes $1 < n$ $[x = n - 1](mod\ n)$
shows $[x\hat{}2 = 1](mod\ n)$
⟨*proof*⟩

lemma *odd-prime-gt-2-int*:
$2 < p$ **if** *odd p prime p* **for** $p :: int$
⟨*proof*⟩

lemma *odd-prime-gt-2-nat*:
$2 < p$ **if** *odd p prime p* **for** $p :: nat$
⟨*proof*⟩

lemma *gt-one-imp-gt-one-power-if-coprime*:
$1 \leq x \Longrightarrow 1 < n \Longrightarrow coprime\ x\ n \Longrightarrow 1 \leq x\ \hat{}\ (n - 1)\ mod\ n$
⟨*proof*⟩

lemma *residue-one-dvd*: $a\ mod\ n = 1 \Longrightarrow n\ dvd\ a - 1$ **for** $a\ n :: nat$
⟨*proof*⟩

lemma *coprimeI-power-mod*:
fixes $x\ r\ n :: nat$
assumes $x\ \hat{}\ r\ mod\ n = 1\ r \neq 0\ n \neq 0$
shows *coprime x n*
⟨*proof*⟩

lemma *prime-dvd-choose*:
assumes $0 < k\ k < p\ prime\ p$
shows $p\ dvd\ (p\ choose\ k)$
⟨*proof*⟩

lemma *cong-eq-0-I*: $(\forall\, i{\in}A.\ [f\ i\ mod\ n = 0]\ (mod\ n)) \Longrightarrow [\sum i{\in}A.\ f\ i = 0]\ (mod\ n)$
⟨*proof*⟩

lemma *power-mult-cong*:
assumes $[x\hat{}n = a](mod\ m)\ [y\hat{}n = b](mod\ m)$
shows $[(x{*}y)\hat{}n = a{*}b](mod\ m)$
⟨*proof*⟩

lemma
fixes $n :: nat$
assumes $n > 1$
shows *odd-pow-cong*: $odd\ m \Longrightarrow [(n - 1)\ \hat{}\ m = n - 1]\ (mod\ n)$
and *even-pow-cong*: $even\ m \Longrightarrow [(n - 1)\ \hat{}\ m = 1]\ (mod\ n)$
⟨*proof*⟩

lemma *cong-mult-uneq'*:
fixes $a :: 'a{::}\{unique\text{-}euclidean\text{-}ring,\ ring\text{-}gcd\}$
assumes *coprime d a*
shows $[b \neq c]\ (mod\ a) \Longrightarrow [d = e]\ (mod\ a) \Longrightarrow [b * d \neq c * e]\ (mod\ a)$
⟨*proof*⟩

lemma *p-coprime-right-nat*: $prime\ p \Longrightarrow coprime\ a\ p = (\neg\ p\ dvd\ a)$ **for** $p\ a :: nat$
⟨*proof*⟩

lemma *squarefree-mult-imp-coprime*:
assumes $squarefree\ (a * b :: 'a :: semiring\text{-}gcd)$
shows *coprime a b*
⟨*proof*⟩

lemma *prime-divisor-exists-strong*:
fixes $m :: int$
assumes $m > 1\ \neg prime\ m$
shows $\exists\, n\ k.\ m = n * k \wedge 1 < n \wedge n < m \wedge 1 < k \wedge k < m$

$\langle proof \rangle$

lemma *prime-divisor-exists-strong-nat*:
  fixes m :: *nat*
  assumes $1 < m$ $\neg$*prime* m
  shows $\exists p\ k.\ m = p * k \wedge 1 < p \wedge p < m \wedge 1 < k \wedge k < m \wedge$ *prime* p
$\langle proof \rangle$

lemma *prime-factorization-eqI*:
  assumes $\bigwedge p.\ p \in\# P \Longrightarrow$ *prime* p *prod-mset* $P = n$
  shows *prime-factorization* $n = P$
  $\langle proof \rangle$

lemma *prime-factorization-prime-elem*:
  assumes *prime-elem* p
  shows *prime-factorization* $p = \{\#$*normalize* $p\#\}$
$\langle proof \rangle$

lemma *size-prime-factorization-eq-Suc-0-iff* [*simp*]:
  fixes n :: $'a$ :: *factorial-semiring-multiplicative*
  shows *size* (*prime-factorization* n) = *Suc* $0 \longleftrightarrow$ *prime-elem* n
$\langle proof \rangle$

lemma *squarefree-prime-elem* [*simp*, *intro*]:
  fixes p :: $'a$ :: *algebraic-semidom*
  assumes *prime-elem* p
  shows *squarefree* p
$\langle proof \rangle$

lemma *squarefree-prime* [*simp*, *intro*]: *prime* $p \Longrightarrow$ *squarefree* p
  $\langle proof \rangle$

lemma *not-squarefree-primepow*:
  assumes *primepow* n
  shows *squarefree* $n \longleftrightarrow$ *prime* n
  $\langle proof \rangle$

lemma *prime-factorization-normalize* [*simp*]:
  prime-factorization (*normalize* n) = *prime-factorization* n
  $\langle proof \rangle$

lemma *one-prime-factor-iff-primepow*:
  fixes n :: $'a$ :: *factorial-semiring-multiplicative*
  shows *card* (*prime-factors* n) = *Suc* $0 \longleftrightarrow$ *primepow* (*normalize* n)
$\langle proof \rangle$

lemma *squarefree-imp-prod-prime-factors-eq*:
  fixes $x :: 'a :: factorial\text{-}semiring\text{-}multiplicative$
  assumes *squarefree x*
  shows $\prod(prime\text{-}factors\ x) = normalize\ x$
⟨*proof*⟩

end

3 The Jacobi Symbol

theory *Jacobi-Symbol*
imports
  Legendre-Symbol
  Algebraic-Auxiliaries
begin

The Jacobi symbol is a generalisation of the Legendre symbol to non-primes [4, 2]. It is defined as

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{k_1} \cdots \left(\frac{a}{p_l}\right)^{k_l}$$

where $(\frac{a}{p})$ denotes the Legendre symbol, a is an integer, n is an odd natural number and $p_1^{k_1} \dots p_l^{k_l}$ is its prime factorisation.
There is, however, a fairly natural generalisation to all non-zero integers for n. It is less clear what a good choice for $n = 0$ is; Mathematica and Maxima adopt the convention that $(\frac{\pm 1}{0}) = 1$ and $(\frac{a}{0}) = 0$ otherwise. However, we chose the slightly different convention $(\frac{a}{0}) = 0$ for *all* a because then the Jacobi symbol is completely multiplicative in both arguments without any restrictions.

definition $Jacobi :: int \Rightarrow int \Rightarrow int$ **where**
  $Jacobi\ a\ n = (if\ n = 0\ then\ 0\ else$
    $(\prod p{\in}\#prime\text{-}factorization\ n.\ Legendre\ a\ p))$

lemma *Jacobi-0-right* [*simp*]: $Jacobi\ a\ 0 = 0$
  ⟨*proof*⟩

lemma *Jacobi-mult-left* [*simp*]: $Jacobi\ (a * b)\ n = Jacobi\ a\ n * Jacobi\ b\ n$
⟨*proof*⟩

lemma *Jacobi-mult-right* [*simp*]: $Jacobi\ a\ (n * m) = Jacobi\ a\ n * Jacobi\ a\ m$
  ⟨*proof*⟩

lemma *prime-p-Jacobi-eq-Legendre*[*intro*!]: $prime\ p \Longrightarrow Jacobi\ a\ p = Legendre\ a\ p$
  ⟨*proof*⟩

lemma *Jacobi-mod* [*simp*]: $Jacobi\ (a\ mod\ m)\ n = Jacobi\ a\ n$ **if** $n\ dvd\ m$
⟨*proof*⟩

lemma *Jacobi-mod-cong*: $[a = b]\ (mod\ n) \Longrightarrow Jacobi\ a\ n = Jacobi\ b\ n$
⟨*proof*⟩

lemma *Jacobi-1-eq-1* [*simp*]: $p \neq 0 \Longrightarrow Jacobi\ 1\ p = 1$
⟨*proof*⟩

lemma *Jacobi-eq-0-not-coprime*:
assumes $n \neq 0$ $\neg coprime\ a\ n$
shows $Jacobi\ a\ n = 0$
⟨*proof*⟩

lemma *Jacobi-p-eq-2'*[*simp*]: $n > 0 \Longrightarrow Jacobi\ a\ (2\hat{}n) = a\ mod\ 2$
⟨*proof*⟩

lemma *Jacobi-prod-mset*[*simp*]: $n \neq 0 \Longrightarrow Jacobi\ (prod\text{-}mset\ M)\ n = (\prod q{\in}\#M.\ Jacobi\ q\ n)$
⟨*proof*⟩

lemma *non-trivial-coprime-neq*:
$1 < a \Longrightarrow 1 < b \Longrightarrow coprime\ a\ b \Longrightarrow a \neq b$ **for** $a\ b :: int$ ⟨*proof*⟩

lemma *odd-odd-even*:
fixes $a\ b :: int$
assumes $odd\ a$ $odd\ b$
shows $even\ ((a*b-1)\ div\ 2) = even\ ((a-1)\ div\ 2 + (b-1)\ div\ 2)$
⟨*proof*⟩

lemma *prime-nonprime-wlog* [*case-names primes nonprime sym*]:
assumes $\bigwedge p\ q.\ prime\ p \Longrightarrow prime\ q \Longrightarrow P\ p\ q$
assumes $\bigwedge p\ q.\ \neg prime\ p \Longrightarrow P\ p\ q$
assumes $\bigwedge p\ q.\ P\ p\ q \Longrightarrow P\ q\ p$
shows $P\ p\ q$
⟨*proof*⟩

lemma *Quadratic-Reciprocity-Jacobi*:
fixes $p\ q :: int$
assumes $coprime\ p\ q$
and $2 < p$ $2 < q$
and $odd\ p$ $odd\ q$
shows $Jacobi\ p\ q * Jacobi\ q\ p =$
$(-\ 1)\ \hat{}(nat\ ((p - 1)\ div\ 2 * ((q - 1)\ div\ 2)))$
⟨*proof*⟩

lemma *Jacobi-values*: $Jacobi\ p\ q \in \{1, -1, 0\}$
⟨*proof*⟩

lemma *Quadratic-Reciprocity-Jacobi'*:
  fixes $p\ q :: int$
  assumes *coprime* $p\ q$
    and $2 < p\ 2 < q$
    and *odd* p *odd* q
  shows $Jacobi\ q\ p = (if\ p\ mod\ 4 = 3 \wedge q\ mod\ 4 = 3\ then\ -1\ else\ 1) * Jacobi\ p\ q$
$\langle proof \rangle$

lemma *dvd-odd-square*: $8\ dvd\ a^2 - 1$ **if** *odd* a **for** $a :: int$
$\langle proof \rangle$

lemma *odd-odd-even'*:
  fixes $a\ b :: int$
  assumes *odd* a *odd* b
  shows $even\ (((a * b)^2 - 1)\ div\ 8) \longleftrightarrow even\ (((a^2 - 1)\ div\ 8) + ((b^2 - 1)\ div\ 8))$
$\langle proof \rangle$

lemma *odd-odd-even-nat'*:
  fixes $a\ b :: nat$
  assumes *odd* a *odd* b
  shows $even\ (((a * b)^2 - 1)\ div\ 8) \longleftrightarrow even\ (((a^2 - 1)\ div\ 8) + ((b^2 - 1)\ div\ 8))$
$\langle proof \rangle$

lemma *supplement2-Jacobi*: $odd\ p \Longrightarrow p > 1 \Longrightarrow Jacobi\ 2\ p = (-\ 1)\ \hat{}\ (((nat\ p)^2 - 1)\ div\ 8)$
$\langle proof \rangle$

lemma *mod-nat-wlog* [*consumes 1*, *case-names modulo*]:
  fixes $P :: nat \Rightarrow bool$
  assumes $b > 0$
  assumes $\bigwedge k.\ k \in \{0..<b\} \Longrightarrow n\ mod\ b = k \Longrightarrow P\ n$
  shows $P\ n$
  $\langle proof \rangle$

lemma *mod-int-wlog* [*consumes 1*, *case-names modulo*]:
  fixes $P :: int \Rightarrow bool$
  assumes $b > 0$
  assumes $\bigwedge k.\ 0 \leq k \Longrightarrow k < b \Longrightarrow n\ mod\ b = k \Longrightarrow P\ n$
  shows $P\ n$
  $\langle proof \rangle$

lemma *supplement2-Jacobi'*:
  assumes *odd* p **and** $p > 1$
  shows $Jacobi\ 2\ p = (if\ p\ mod\ 8 = 1 \vee p\ mod\ 8 = 7\ then\ 1\ else\ -1)$
$\langle proof \rangle$

theorem *supplement1-Jacobi*:
  $odd\ p \Longrightarrow 1 < p \Longrightarrow Jacobi\ (-1)\ p = (-1)\ \hat{}\,(nat\ ((p - 1)\ div\ 2))$
⟨*proof*⟩

theorem *supplement1-Jacobi′*:
  $odd\ n \Longrightarrow 1 < n \Longrightarrow Jacobi\ (-1)\ n = (if\ n\ mod\ 4 = 1\ then\ 1\ else\ -1)$
  ⟨*proof*⟩

lemma *Jacobi-0-eq-0*: $\neg is\text{-}unit\ n \Longrightarrow Jacobi\ 0\ n = 0$
  ⟨*proof*⟩

lemma *is-unit-Jacobi-aux*: $is\text{-}unit\ x \Longrightarrow Jacobi\ a\ x = 1$
  ⟨*proof*⟩

lemma *is-unit-Jacobi*[*simp*]: $Jacobi\ a\ 1 = 1$ $Jacobi\ a\ (-1) = 1$
  ⟨*proof*⟩

lemma *Jacobi-neg-right* [*simp*]:
  $Jacobi\ a\ (-n) = Jacobi\ a\ n$
⟨*proof*⟩

lemma *Jacobi-neg-left*:
  assumes $odd\ n$ $1 < n$
  shows $Jacobi\ (-a)\ n = (if\ n\ mod\ 4 = 1\ then\ 1\ else\ -1) * Jacobi\ a\ n$
⟨*proof*⟩

function *jacobi-code* :: $int \Rightarrow int \Rightarrow int$ **where**
jacobi-code a n = (
      if n = *0 then 0*
  else if n = *1 then 1*
  else if a = *1 then 1*
  else if n < *0 then jacobi-code a* (−*n*)
  else if even n then if even a then 0 else jacobi-code a (*n div 2*)
  else if a < *0 then* (*if n mod 4* = *1 then 1 else* −*1*) ∗ *jacobi-code* (−*a*) *n*
  else if a = *0 then 0*
  else if a ≥ *n then jacobi-code* (*a mod n*) *n*
  else if even a   *then* (*if n mod 8* ∈ {*1*, *7*} *then 1 else* −*1*) ∗ *jacobi-code* (*a div 2*) *n*
  else if coprime a n then (*if n mod 4* = *3* ∧ *a mod 4* = *3 then* −*1 else 1*) ∗ *jacobi-code n a*
  else 0)
  ⟨*proof*⟩
termination
⟨*proof*⟩

lemmas [*simp del*] = *jacobi-code.simps*

lemma *Jacobi-code* [*code*]: *Jacobi a n* = *jacobi-code a n*

$\langle proof \rangle$

lemma *Jacobi-eq-0-imp-not-coprime*:
  assumes $p \neq 0$ $p \neq 1$
  shows $Jacobi\ n\ p = 0 \Longrightarrow \neg coprime\ n\ p$
  $\langle proof \rangle$

lemma *Jacobi-eq-0-iff-not-coprime*:
  assumes $p \neq 0$ $p \neq 1$
  shows $Jacobi\ n\ p = 0 \longleftrightarrow \neg coprime\ n\ p$
$\langle proof \rangle$

end

4 Residue Rings of Natural Numbers

theory *Residues-Nat*
  imports *Algebraic-Auxiliaries*
begin

4.1 The multiplicative group of residues modulo n

definition *Residues-Mult* :: $'a$ :: $\{linordered\text{-}semidom,\ euclidean\text{-}semiring\} \Rightarrow 'a$ *monoid* **where**
  $Residues\text{-}Mult\ p =$
    $(\!|carrier = \{x \in \{1..p\}\ .\ coprime\ x\ p\},\ monoid.mult = \lambda x\ y.\ x * y\ mod\ p,\ one = 1|\!)$

locale *residues-mult-nat* =
  fixes n :: *nat* **and** G
  assumes *n-gt-1*: $n > 1$
  defines $G \equiv Residues\text{-}Mult\ n$
begin

lemma *carrier-eq* [*simp*]: $carrier\ G = totatives\ n$
  and *mult-eq* [*simp*]: $(x \otimes_G y) = (x * y)\ mod\ n$
  and *one-eq* [*simp*]: $\mathbf{1}_G = 1$
  $\langle proof \rangle$

lemma *mult-eq'*: $(\otimes_G) = (\lambda x\ y.\ (x * y)\ mod\ n)$
  $\langle proof \rangle$

sublocale *group* G
$\langle proof \rangle$

sublocale *comm-group*
  $\langle proof \rangle$

lemma *nat-pow-eq* [*simp*]: $x\ [\urcorner]_G\ (k :: nat) = (x\ \widehat{}\ k)\ mod\ n$

⟨*proof*⟩

lemma *nat-pow-eq′*: $([\widehat{\ }]_G) = (\lambda x\ k.\ (x \mathbin{\widehat{}} k)\ mod\ n)$
⟨*proof*⟩

lemma *order-eq*: $order\ G = totient\ n$
⟨*proof*⟩

lemma *order-less*: $\neg prime\ n \Longrightarrow order\ G < n - 1$
⟨*proof*⟩

lemma *ord-residue-mult-group*:
assumes $a \in totatives\ n$
shows $local.ord\ a = Pocklington.ord\ n\ a$
⟨*proof*⟩

end

4.2 The ring of residues modulo n

definition *Residues-nat* :: $nat \Rightarrow nat\ ring$ **where**
$Residues\text{-}nat\ m = (\!|carrier = \{0..<m\},\ monoid.mult = \lambda x\ y.\ (x * y)\ mod\ m,\ one = 1,$
$ring.zero = 0,\ add = \lambda x\ y.\ (x + y)\ mod\ m|\!)$

locale *residues-nat* =
fixes n :: *nat* **and** R
assumes *n-gt-1*: $n > 1$
defines $R \equiv Residues\text{-}nat\ n$
begin

lemma *carrier-eq* [*simp*]: $carrier\ R = \{0..<n\}$
and *mult-eq* [*simp*]: $x \otimes_R y = (x * y)\ mod\ n$
and *add-eq* [*simp*]: $x \oplus_R y = (x + y)\ mod\ n$
and *one-eq* [*simp*]: $\mathbf{1}_R = 1$
and *zero-eq* [*simp*]: $\mathbf{0}_R = 0$
⟨*proof*⟩

lemma *mult-eq′*: $(\otimes_R) = (\lambda x\ y.\ (x * y)\ mod\ n)$
and *add-eq′*: $(\oplus_R) = (\lambda x\ y.\ (x + y)\ mod\ n)$
⟨*proof*⟩

sublocale *abelian-group R*
⟨*proof*⟩

sublocale *comm-monoid R*
⟨*proof*⟩

sublocale *cring R*

⟨*proof*⟩

lemma *Units-eq*: *Units R* = *totatives n*
⟨*proof*⟩

sublocale *units*: *residues-mult-nat n units-of R*
⟨*proof*⟩

lemma *nat-pow-eq* [*simp*]: $x\ [\widehat{\ }]_R\ (k :: nat) = (x \mathbin{\widehat{}} k)\ mod\ n$
⟨*proof*⟩

lemma *nat-pow-eq'*: $([\widehat{\ }]_R) = (\lambda x\ k.\ (x \mathbin{\widehat{}} k)\ mod\ n)$
⟨*proof*⟩

end

4.3 The ring of residues modulo a prime

locale *residues-nat-prime* =
fixes *p* :: *nat* **and** *R*
assumes *prime-p*: *prime p*
defines *R* ≡ *Residues-nat p*
begin

sublocale *residues-nat p R*
⟨*proof*⟩

lemma *carrier-eq'* [*simp*]: *totatives p* = {*0*<..<*p*}
⟨*proof*⟩

lemma *order-eq*: *order* (*units-of R*) = *p* − *1*
⟨*proof*⟩

lemma *order-eq'* [*simp*]: *totient p* = *p* − *1*
⟨*proof*⟩

sublocale *field R*
⟨*proof*⟩

lemma *residues-prime-cyclic*: ∃ *x*∈{*0*<..<*p*}. {*0*<..<*p*} = {*y*. ∃ *i*. *y* = *x* ^ *i mod p*}
⟨*proof*⟩

lemma *residues-prime-cyclic'*: ∃ *x*∈{*0*<..<*p*}. *units.ord x* = *p* − *1*
⟨*proof*⟩

end

4.4 -1 in residue rings

lemma *minus-one-cong-solve-weak*:
  fixes $n\ x :: nat$
  assumes $1 < n\ x \in totatives\ n\ y \in totatives\ n$
    and $[x = n - 1]\ (mod\ n)\ [x * y = 1]\ (mod\ n)$
  shows $y = n - 1$
⟨*proof*⟩

lemma *coprime-imp-mod-not-zero*:
  fixes $n\ x :: nat$
  assumes $1 < n\ coprime\ x\ n$
  shows $0 < x\ mod\ n$
  ⟨*proof*⟩

lemma *minus-one-cong-solve*:
  fixes $n\ x :: nat$
  assumes $1 < n$
    and *eq*: $[x = n - 1]\ (mod\ n)\ [x * y = 1]\ (mod\ n)$
    and *coprime*: $coprime\ x\ n\ coprime\ y\ n$
  shows $[y = n - 1](mod\ n)$
⟨*proof*⟩

corollary *square-minus-one-cong-one'*:
  fixes $n\ x :: nat$
  assumes $1 < n$
  shows $[(n - 1) * (n - 1) = 1](mod\ n)$
  ⟨*proof*⟩

end

5 Additional Material on Quadratic Residues

theory *QuadRes*
imports
  Jacobi-Symbol
  Algebraic-Auxiliaries
begin

Proofs are inspired by [5].

lemma *inj-on-QuadRes*:
  fixes $p :: int$
  assumes $prime\ p$
  shows $inj\text{-}on\ (\lambda x.\ x\hat{\ }2\ mod\ p)\ \{0..(p-1)\ div\ 2\}$
⟨*proof*⟩

lemma *QuadRes-set-prime*:
  assumes $prime\ p$ **and** $odd\ p$
  shows $\{x\ .\ QuadRes\ p\ x \land x \in \{0..<p\}\} = \{x\hat{\ }2\ mod\ p \mid x\ .\ x \in \{0..(p-1)\ div$

2}}
⟨*proof*⟩

corollary *QuadRes-iff*:
  assumes *prime p* **and** *odd p*
  shows (*QuadRes p x* ∧ *x* ∈ {*0*..<*p*}) ⟷ (∃ *a* ∈ {*0*..(*p*−*1*) *div 2*}. *a*^*2 mod p* = *x*)
⟨*proof*⟩

corollary *card-QuadRes-set-prime*:
  fixes *p* :: *int*
  assumes *prime p* **and** *odd p*
  shows *card* {*x*. *QuadRes p x* ∧ *x* ∈ {*0*..<*p*}} = *nat* (*p*+*1*) *div 2*
⟨*proof*⟩

corollary *card-not-QuadRes-set-prime*:
  fixes *p* :: *int*
  assumes *prime p* **and** *odd p*
  shows *card* {*x*. ¬*QuadRes p x* ∧ *x* ∈ {*0*..<*p*}} = *nat* (*p*−*1*) *div 2*
⟨*proof*⟩

lemma *not-QuadRes-ex-if-prime*:
  assumes *prime p* **and** *odd p*
  shows ∃ *x*. ¬*QuadRes p x*
⟨*proof*⟩

lemma *not-QuadRes-ex*:
  1 < *p* ⟹ *odd p* ⟹ ∃ *x*. ¬*QuadRes p x*
⟨*proof*⟩

end

6 Euler Witnesses

theory *Euler-Witness*
imports
  Jacobi-Symbol
  Residues-Nat
  QuadRes
begin

Proofs are inspired by [13, 8, 15, 11].

definition *euler-witness a p* ⟷ [*Jacobi a p* ≠ *a* ^ ((*p* − *1*) *div 2*)] (*mod p*)
abbreviation *euler-liar a p* ≡ ¬ *euler-witness a p*

lemma *euler-witness-mod*[*simp*]: *euler-witness* (*a mod p*) *p* = *euler-witness a p*
  ⟨*proof*⟩

lemma *euler-liar-mod*: *euler-liar* (*a mod p*) *p* = *euler-liar a p* ⟨*proof*⟩

lemma *euler-liar-cong*:
  assumes $[a = b]$ $(mod\ p)$
  shows *euler-liar a p = euler-liar b p*
  ⟨*proof*⟩

lemma *euler-witnessI*:
  $[x \,\widehat{}\, ((n - 1)\ div\ 2) = a]$ $(mod\ int\ n\ ) \Longrightarrow [Jacobi\ x\ (int\ n) \neq a]$ $(mod\ int\ n)$
  $\Longrightarrow$ *euler-witness x n*
  ⟨*proof*⟩

lemma *euler-witnessI2*:
  fixes *a b* :: *int* **and** *k* :: *nat*
  assumes $[a \neq b]$ $(mod\ k)$
  and *k dvd n*
  and *main*: *euler-liar x n* $\Longrightarrow [Jacobi\ x\ n = a]$ $(mod\ k)$
  euler-liar x n $\Longrightarrow [x \,\widehat{}\, ((n - 1)\ div\ 2) = b]$ $(mod\ k)$
  shows *euler-witness x n*
⟨*proof*⟩

lemma *euler-witness-exists-weak*:
  assumes *odd n* ¬*prime n* $2 < n$
  shows ∃ *a. euler-witness a n* ∧ *coprime a n*
⟨*proof*⟩

lemma *euler-witness-exists*:
  assumes *odd n* ¬*prime n* $2 < n$
  shows ∃ *a. euler-witness a n* ∧ *coprime a n* ∧ $0 < a \wedge a < n$
⟨*proof*⟩

lemma *euler-witness-exists-nat*:
  assumes *odd n* ¬*prime n* $2 < n$
  shows ∃ *a. euler-witness (int a) n* ∧ *coprime a n* ∧ $0 < a \wedge a < n$
  ⟨*proof*⟩

lemma *euler-liar-1-p*[*intro*, *simp*]: $p \neq 0 \Longrightarrow$ *euler-liar 1 p*
  ⟨*proof*⟩

lemma *euler-liar-mult*:
  shows *euler-liar y n* $\Longrightarrow$ *euler-liar x n* $\Longrightarrow$ *euler-liar* $(x*y)$ *n*
  ⟨*proof*⟩

lemma *euler-liar-mult'*:
  assumes $1 < n$ *coprime y n*
  shows *euler-liar y n* $\Longrightarrow$ *euler-witness x n* $\Longrightarrow$ *euler-witness* $(x*y)$ *n*
⟨*proof*⟩

lemma *prime-imp-euler-liar*:
  assumes *prime p* $2 < p$ $0 < x$ $x < p$

shows *euler-liar x p*
⟨*proof*⟩

locale *euler-witness-context* =
fixes *p* :: *nat*
assumes *p-gt-1*: $p > 1$ **and** *odd-p*: *odd p*
begin

definition *G* **where** *G* = *Residues-Mult p*

sublocale *residues-mult-nat p G*
⟨*proof*⟩

definition $H = \{x.\ \mathit{coprime}\ x\ p \wedge \mathit{euler\text{-}liar}\ (\mathit{int}\ x)\ p \wedge x \in \{1..<p\}\}$

sublocale *H*: *subgroup H G*
⟨*proof*⟩

lemma *H-finite*: *finite H*
⟨*proof*⟩

lemma *euler-witness-coset*:
assumes *euler-witness x p*
shows $y \in H \ \#>_G x \Longrightarrow$ *euler-witness y p*
⟨*proof*⟩

lemma *euler-liar-coset*:
assumes *euler-liar x p* $x \in$ *carrier G*
shows $y \in H \ \#>_G x \Longrightarrow$ *euler-liar y p*
⟨*proof*⟩

lemma *in-cosets-aux*:
assumes *euler-witness x p* $x \in$ *carrier G*
shows $H \ \#>_G x \in \mathit{rcosets}_G\ H$
⟨*proof*⟩

lemma *H-cosets-aux*:
assumes *euler-witness x p*
shows $(H \ \#>_G x) \cap H = \{\}$
⟨*proof*⟩

lemma *H-rcosets-aux*:
assumes *euler-witness x p* $x \in$ *carrier G*
shows $\{H,\ H \ \#>_G x\} \subseteq \mathit{rcosets}_G\ H$
⟨*proof*⟩

lemma *H-not-eq-coset*:
assumes *euler-witness x p*
shows $H \neq H \ \#>_G x$

⟨*proof*⟩

lemma *finite-cosets-H*: *finite* (*rcosets*$_G$ *H*)
⟨*proof*⟩

lemma *card-cosets-limit*:
assumes *euler-witness x p x* ∈ *carrier G*
shows *2* ≤ *card* (*rcosets*$_G$ *H*)
⟨*proof*⟩

lemma *card-euler-liars-cosets-limit*:
assumes ¬*prime p 2* < *p*
shows *2* ≤ *card* (*rcosets*$_G$ *H*) *card H* < (*p* − *1*) *div 2*
⟨*proof*⟩

lemma *prime-imp-G-is-H*:
assumes *prime p 2* < *p*
shows *carrier G* = *H*
⟨*proof*⟩

end

end

7 Carmichael Numbers

theory *Carmichael-Numbers*
imports
Residues-Nat
begin

A Carmichael number is a composite number n that Fermat's test incorrectly labels as primes no matter which witness a is chosen (except in the case that a shares a factor with n). [9, 14]

definition *Carmichael-number* :: *nat* ⇒ *bool* **where**
Carmichael-number n ⟷ *n* > *1* ∧ ¬*prime n* ∧ (∀ *a*. *coprime a n* ⟶ [*a* ^ (*n* − *1*) = *1*] (*mod n*))

lemma *Carmichael-number-0*[*simp*, *intro*]: ¬*Carmichael-number 0*
⟨*proof*⟩

lemma *Carmichael-number-1*[*simp*, *intro*]: ¬*Carmichael-number 1*
⟨*proof*⟩

lemma *Carmichael-number-Suc-0*[*simp*, *intro*]: ¬*Carmichael-number* (*Suc 0*)
⟨*proof*⟩

lemma *Carmichael-number-not-prime*: *Carmichael-number n* ⟹ ¬*prime n*

⟨*proof*⟩

lemma *Carmichael-number-gt-3*: *Carmichael-number* $n \Longrightarrow n > 3$
⟨*proof*⟩

The proofs are inspired by [9, 12].

lemma *Carmichael-number-imp-squarefree-aux*:
assumes *Carmichael-number n*
assumes n: $n = p \hat{} r * l$ **and** *prime p* $\neg p$ *dvd l*
assumes $r > 1$
shows *False*
⟨*proof*⟩

theorem *Carmichael-number-imp-squarefree*:
assumes *Carmichael-number n*
shows *squarefree n*
⟨*proof*⟩

corollary *Carmichael-not-primepow*:
assumes *Carmichael-number n*
shows $\neg$*primepow n*
⟨*proof*⟩

lemma *Carmichael-number-imp-squarefree-alt-weak*:
assumes *Carmichael-number n*
shows $\exists p\ l.\ (n = p * l) \wedge$ *prime p* $\wedge \neg p$ *dvd l*
⟨*proof*⟩

theorem *Carmichael-number-odd*:
assumes *Carmichael-number n*
shows *odd n*
⟨*proof*⟩

lemma *Carmichael-number-imp-squarefree-alt*:
assumes *Carmichael-number n*
shows $\exists p\ l.\ (n = p * l) \wedge$ *prime p* $\wedge \neg p$ *dvd l* $\wedge\ 2 < l$
⟨*proof*⟩

lemma *Carmichael-number-imp-dvd*:
fixes n :: *nat*
assumes *Carmichael-number*: *Carmichael-number n* **and** *prime p p dvd n*
shows $p - 1$ *dvd* $n - 1$
⟨*proof*⟩

The following lemma is also called Korselt's criterion.

lemma *Carmichael-numberI*:
fixes n :: *nat*
assumes $\neg$ *prime n squarefree n* $1 < n$ **and**
DIV: $\bigwedge p.\ p \in$ *prime-factors n* $\Longrightarrow p - 1$ *dvd* $n - 1$

shows *Carmichael-number n*
⟨*proof*⟩

theorem *Carmichael-number-iff*:
Carmichael-number n $\longleftrightarrow$
$n \neq 1 \wedge \neg prime\ n \wedge squarefree\ n \wedge (\forall p \in prime\text{-}factors\ n.\ p - 1\ dvd\ n - 1)$
⟨*proof*⟩

Every Carmichael number has at least three distinct prime factors.

theorem *Carmichael-number-card-prime-factors*:
assumes *Carmichael-number n*
shows $card\ (prime\text{-}factors\ n) \geq 3$
⟨*proof*⟩

lemma *Carmichael-number-iff'*:
fixes *n* :: *nat*
defines $P \equiv prime\text{-}factorization\ n$
shows *Carmichael-number n* $\longleftrightarrow$
$n > 1 \wedge size\ P \neq 1 \wedge (\forall p \in \# P.\ count\ P\ p = 1 \wedge p - 1\ dvd\ n - 1)$
⟨*proof*⟩

The smallest Carmichael number is 561, and it was found and proven so by Carmichael in 1910 [6].

lemma *Carmichael-number-561*: *Carmichael-number 561* (**is** *Carmichael-number ?n*)
⟨*proof*⟩

end

8 Fermat Witnesses

theory *Fermat-Witness*
imports *Euler-Witness Carmichael-Numbers*
begin

definition *divide-out* :: $'a$ :: *factorial-semiring* $\Rightarrow 'a \Rightarrow 'a \times nat$ **where**
$divide\text{-}out\ p\ x = (x\ div\ p\ \widehat{}\ multiplicity\ p\ x,\ multiplicity\ p\ x)$

lemma *fst-divide-out* [*simp*]: $fst\ (divide\text{-}out\ p\ x) = x\ div\ p\ \widehat{}\ multiplicity\ p\ x$
and *snd-divide-out* [*simp*]: $snd\ (divide\text{-}out\ p\ x) = multiplicity\ p\ x$
⟨*proof*⟩

function *divide-out-aux* :: $'a$ :: *factorial-semiring* $\Rightarrow 'a \times nat \Rightarrow 'a \times nat$ **where**
$divide\text{-}out\text{-}aux\ p\ (x,\ acc) =$
$(if\ x = 0 \vee is\text{-}unit\ p \vee \neg p\ dvd\ x\ then\ (x,\ acc)\ else\ divide\text{-}out\text{-}aux\ p\ (x\ div\ p,\ acc + 1))$
⟨*proof*⟩

termination ⟨*proof*⟩

lemmas [*simp del*] = *divide-out-aux.simps*

lemma *divide-out-aux-correct*:
divide-out-aux p z = (*fst z div p* ^ *multiplicity p* (*fst z*), *snd z* + *multiplicity p* (*fst z*))
⟨*proof*⟩

lemma *divide-out-code* [*code*]: *divide-out p x* = *divide-out-aux p* (*x*, *0*)
⟨*proof*⟩

lemma *multiplicity-code* [*code*]: *multiplicity p x* = *snd* (*divide-out-aux p* (*x*, *0*))
⟨*proof*⟩

lemma *multiplicity-times-same-power*:
assumes $x \neq 0$ ¬*is-unit p* $p \neq 0$
shows *multiplicity p* (*p* ^ *k* ∗ *x*) = *multiplicity p x* + *k*
⟨*proof*⟩

lemma *divide-out-unique-nat*:
fixes *n* :: *nat*
assumes ¬*is-unit p* $p \neq 0$ ¬*p dvd m* **and** *n* = *p* ^ *k* ∗ *m*
shows *m* = *n div p* ^ *multiplicity p n* **and** *k* = *multiplicity p n*
⟨*proof*⟩

context
fixes *a n* :: *nat*
begin

definition *fermat-liar* ⟷ [*a*^(*n*−*1*) = *1*] (*mod n*)
definition *fermat-witness* ⟷ [*a*^(*n*−*1*) ≠ *1*] (*mod n*)

definition *strong-fermat-liar* ⟷
(∀ *k m*. *odd m* ⟶ *n* − *1* = *2*^*k* ∗ *m* ⟶
[*a*^*m* = *1*](*mod n*) ∨ (∃ *i* ∈ {*0*..< *k*}. [*a*^(*2* ^ *i* ∗ *m*) = *n*−*1*] (*mod n*)))

definition *strong-fermat-witness* ⟷ ¬ *strong-fermat-liar*

lemma *fermat-liar-witness-of-composition*[*iff*]:
¬*fermat-liar* = *fermat-witness*
¬*fermat-witness* = *fermat-liar*
⟨*proof*⟩

lemma *strong-fermat-liar-code* [*code*]:
strong-fermat-liar ⟷

$(let\ (m,\ k) = divide\text{-}out\ 2\ (n - 1)$
$in\ [a\hat{\ }m = 1](mod\ n) \vee (\exists\, i \in \{0..< k\}.\ [a\hat{\ }(2\ \hat{\ }\ i * m) = n{-}1]\ (mod\ n)))$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

context
assumes $* : a \in \{1\ ..< n\}$
begin

lemma *strong-fermat-witness-iff*:
strong-fermat-witness =
$(\exists\, k\ m.\ odd\ m \wedge n - 1 = 2\ \hat{\ }\ k * m \wedge [a\ \hat{\ }\ m \neq 1]\ (mod\ n) \wedge$
$(\forall\, i{\in}\{0..{<}k\}.\ [a\ \hat{\ }(2\ \hat{\ }\ i * m) \neq n{-}1]\ (mod\ n)))$
⟨*proof*⟩

lemma *not-coprime-imp-witness*: $\neg coprime\ a\ n \Longrightarrow fermat\text{-}witness$
⟨*proof*⟩

corollary *liar-imp-coprime*: $fermat\text{-}liar \Longrightarrow coprime\ a\ n$
⟨*proof*⟩

lemma *fermat-witness-imp-strong-fermat-witness*:
assumes $1 < n$ *fermat-witness*
shows *strong-fermat-witness*
⟨*proof*⟩

corollary *strong-fermat-liar-imp-fermat-liar*:
assumes $1 < n$ *strong-fermat-liar*
shows *fermat-liar*
⟨*proof*⟩

lemma *euler-liar-is-fermat-liar*:
assumes $1 < n$ *euler-liar a n coprime a n odd n*
shows *fermat-liar*
⟨*proof*⟩

corollary *fermat-witness-is-euler-witness*:
assumes $1 < n$ *fermat-witness coprime a n odd n*
shows *euler-witness a n*
⟨*proof*⟩

end
end

lemma *one-is-fermat-liar*[*simp*]: $1 < n \Longrightarrow fermat\text{-}liar\ 1\ n$
⟨*proof*⟩

lemma *one-is-strong-fermat-liar*[*simp*]: $1 < n \Longrightarrow strong\text{-}fermat\text{-}liar\ 1\ n$

⟨*proof*⟩

lemma *prime-imp-fermat-liar*:
$prime\ p \Longrightarrow a \in \{1 ..< p\} \Longrightarrow fermat\text{-}liar\ a\ p$
⟨*proof*⟩

lemma *not-Carmichael-numberD*:
assumes $\neg Carmichael\text{-}number\ n\ \neg prime\ n$ **and** $1 < n$
shows $\exists\ a \in \{2 ..< n\}\ .\ fermat\text{-}witness\ a\ n \wedge coprime\ a\ n$
⟨*proof*⟩

proposition *prime-imp-strong-fermat-witness*:
fixes $p :: nat$
assumes $prime\ p\ 2 < p\ a \in \{1 ..< p\}$
shows $strong\text{-}fermat\text{-}liar\ a\ p$
⟨*proof*⟩

lemma *ignore-one*:
fixes $P :: \text{-} \Rightarrow nat \Rightarrow bool$
assumes $P\ 1\ n\ 1 < n$
shows $card\ \{a \in \{1..<n\}.\ P\ a\ n\} = 1 + card\ \{a.\ 2 \leq a \wedge a < n \wedge P\ a\ n\}$
⟨*proof*⟩

Proofs in the following section are inspired by [10, 7, 1].

proposition *not-Carmichael-number-imp-card-fermat-witness-bound*:
fixes $n :: nat$
assumes $\neg prime\ n\ \neg Carmichael\text{-}number\ n\ odd\ n\ 1 < n$
shows $(n-1)\ div\ 2 < card\ \{a \in \{1 ..< n\}.\ fermat\text{-}witness\ a\ n\}$
and $(card\ \{a.\ 2 \leq a \wedge a < n \wedge strong\text{-}fermat\text{-}liar\ a\ n\}) < real\ (n - 2)\ /\ 2$
and $(card\ \{a.\ 2 \leq a \wedge a < n \wedge fermat\text{-}liar\ a\ n\}) < real\ (n - 2)\ /\ 2$
⟨*proof*⟩

proposition *Carmichael-number-imp-lower-bound-on-strong-fermat-witness*:
fixes $n :: nat$
assumes *Carmichael-number*: $Carmichael\text{-}number\ n$
shows $(n - 1)\ div\ 2 < card\ \{a \in \{1..<n\}.\ strong\text{-}fermat\text{-}witness\ a\ n\}$
and $real\ (card\ \{a\ .\ 2 \leq a \wedge a < n \wedge strong\text{-}fermat\text{-}liar\ a\ n\}) < real\ (n - 2)$
$/\ 2$
⟨*proof*⟩

corollary *strong-fermat-witness-lower-bound*:
assumes $odd\ n\ n > 2\ \neg prime\ n$
shows $card\ \{a.\ 2 \leq a \wedge a < n \wedge strong\text{-}fermat\text{-}liar\ a\ n\} < real\ (n - 2)\ /\ 2$
⟨*proof*⟩

end

9 A Generic View on Probabilistic Prime Tests

theory *Generalized-Primality-Test*
imports
  HOL−Probability.Probability
  Algebraic-Auxiliaries
begin

definition *primality-test* :: $(nat \Rightarrow nat \Rightarrow bool) \Rightarrow nat \Rightarrow bool\ pmf$ **where**
  primality-test P n =
    (*if* $n < 3 \vee$ *even n then return-pmf* $(n = 2)$ *else*
      do {
        $a \leftarrow$ *pmf-of-set* $\{2..< n\}$;
        return-pmf $(P\ n\ a)$
      })

lemma *expectation-of-bool-is-pmf*: *measure-pmf.expectation M of-bool = pmf M True*
  ⟨*proof*⟩

lemma *eq-bernoulli-pmfI*:
  assumes *pmf p True = x*
  shows *p = bernoulli-pmf x*
⟨*proof*⟩

We require a probabilistic primality test to never classify a prime as composite. It may, however, mistakenly classify composites as primes.

locale *prob-primality-test* =
  fixes $P :: nat \Rightarrow nat \Rightarrow bool$ **and** $n :: nat$
  assumes *P-works*: *odd n* $\Longrightarrow 2 \leq a \Longrightarrow a < n \Longrightarrow$ *prime n* $\Longrightarrow P\ n\ a$
begin

lemma *FalseD*:
  assumes *false*: *False* $\in$ *set-pmf* (*primality-test P n*)
  shows $\neg$ *prime n*
⟨*proof*⟩

theorem *prime*:
  assumes *odd-prime*: *prime n*
  shows *primality-test P n = return-pmf True*
⟨*proof*⟩

end

We call a primality test q-good for a fixed positive real number q if the probability that it mistakenly classifies a composite as a prime is less than q.

locale *good-prob-primality-test* = *prob-primality-test* +

fixes *q* :: *real*
assumes *q-pos*: $q > 0$
assumes *composite-witness-bound*:
$\neg prime\ n \Longrightarrow 2 < n \Longrightarrow odd\ n \Longrightarrow$
$real\ (card\ \{a\ .\ 2 \le a \wedge a < n \wedge P\ n\ a\}) < q * real\ (n - 2)$
begin

lemma *composite-aux*:
assumes $\neg prime\ n$
shows $measure\text{-}pmf.expectation\ (primality\text{-}test\ P\ n)\ of\text{-}bool < q$
⟨*proof*⟩

theorem *composite*:
assumes $\neg prime\ n$
shows $pmf\ (primality\text{-}test\ P\ n)\ True < q$
⟨*proof*⟩

end

end

10 Fermat's Test

theory *Fermat-Test*
imports
Fermat-Witness
Generalized-Primality-Test
begin

definition $fermat\text{-}test = primality\text{-}test\ (\lambda n\ a.\ fermat\text{-}liar\ a\ n)$

The Fermat test is a good probabilistic primality test on non-Carmichael numbers.

locale *fermat-test-not-Carmichael-number* =
fixes *n* :: *nat*
assumes *not-Carmichael-number*: $\neg Carmichael\text{-}number\ n \vee n < 3$
begin

sublocale *fermat-test*: $good\text{-}prob\text{-}primality\text{-}test\ \lambda a\ n.\ fermat\text{-}liar\ n\ a\ n\ 1\ /\ 2$
rewrites $primality\text{-}test\ (\lambda\ a\ n.\ fermat\text{-}liar\ n\ a) = fermat\text{-}test$
⟨*proof*⟩

end

lemma *not-coprime-imp-fermat-witness*:
fixes *n* :: *nat*
assumes $n > 1\ \neg coprime\ a\ n$
shows $fermat\text{-}witness\ a\ n$
⟨*proof*⟩

theorem *fermat-test-prime*:
assumes *prime n*
shows *fermat-test n = return-pmf True*
⟨*proof*⟩

theorem *fermat-test-composite*:
assumes *¬prime n ¬Carmichael-number n ∨ n < 3*
shows *pmf (fermat-test n) True < 1 / 2*
⟨*proof*⟩

For a Carmichael number n, Fermat's test as defined above mistakenly returns 'True' with probability $(\varphi(n) - 1)/(n - 2)$. This probability is close to 1 if n has few and big prime factors; it is not quite as bad if it has many and/or small factors, but in that case, simple trial division can also detect compositeness.

Moreover, Fermat's test only succeeds for a Carmichael number if it happens to guess a number that is not coprime to n. In that case, the fact that we have found a number between 2 and n that is not coprime to n alone is proof that n is composite, and indeed we can even find a non-trivial factor by computing the GCD. This means that for Carmichael numbers, Fermat's test is essentially no better than the very crude method of attempting to guess numbers coprime to n.

This means that, in general, Fermat's test is not very helpful for Carmichael numbers.

theorem *fermat-test-Carmichael-number*:
assumes *Carmichael-number n*
shows *fermat-test n = bernoulli-pmf (real (totient n − 1) / real (n − 2))*
⟨*proof*⟩

end

11 The Miller–Rabin Test

theory *Miller-Rabin-Test*
imports
Fermat-Witness
Generalized-Primality-Test
begin

definition *miller-rabin = primality-test (λn a. strong-fermat-liar a n)*

The test is actually $\frac{1}{4}$ good, but we only show $\frac{1}{2}$, since the former is much more involved.

interpretation *miller-rabin*: *good-prob-primality-test λn a. strong-fermat-liar a n n 1 / 2*

rewrites *primality-test* (λn *a. strong-fermat-liar a n*) = *miller-rabin*
⟨*proof*⟩

end

12 The Solovay–Strassen Test

theory *Solovay-Strassen-Test*
imports
Generalized-Primality-Test
Euler-Witness
begin

definition *solovay-strassen-witness* :: *nat* ⇒ *nat* ⇒ *bool* **where**
solovay-strassen-witness n a =
(*let x* = *Jacobi* (*int a*) (*int n*) *in x* ≠ *0* ∧ [*x* = *int a* ^ ((*n* − *1*) *div 2*)] (*mod n*))

definition *solovay-strassen* :: *nat* ⇒ *bool pmf* **where**
solovay-strassen = *primality-test solovay-strassen-witness*

lemma *prime-imp-solovay-strassen-witness*:
assumes *prime p odd p a* ∈ {*2*..<*p*}
shows *solovay-strassen-witness p a*
⟨*proof*⟩

lemma *card-solovay-strassen-liars-composite*:
fixes *n* :: *nat*
assumes ¬*prime n n* > *2 odd n*
shows *card* {*a* ∈ {*2*..<*n*}. *solovay-strassen-witness n a*} < (*n* − *2*) *div 2*
(**is** *card ?A* < -)
⟨*proof*⟩

interpretation *solovay-strassen*: *good-prob-primality-test solovay-strassen-witness n 1 / 2*
rewrites *primality-test solovay-strassen-witness* = *solovay-strassen*
⟨*proof*⟩

end

References

[1] L. W. after a lecture by Prof. Dr. G. Kemper. Skript: Computer algebra, 2015.

[2] Brilliant.org. Jacobi symbol, 2018. Last accessed 18 November 2018.

[3] Brilliant.org. Law of quadratic reciprocity, 2018. Last accessed 18 November 2018.

[4] Brilliant.org. Legendre symbol, 2018. Last accessed 18 November 2018.

[5] Brilliant.org. Quadratic residues, 2018. Last accessed 18 November 2018.

[6] R. D. Carmichael. Note on a new number theory function. *Bulletin of the American Mathematical Society*, 16(5):232–238, 1910.

[7] K. Conrad. The miller–rabin test, 2018. Last accessed 18 November 2018.

[8] K. Conrad. The solovay–strassen test, 2018. Last accessed 18 November 2018.

[9] G. Jameson. Carmichael numbers and pseudoprimes, 2010. Last accessed 18 November 2018.

[10] B. Kleinberg. The miller-rabin randomized primality test, 2010. Last accessed 18 November 2018.

[11] mathwizard. Solovay–strassen test, 2013. Last accessed 18 November 2018.

[12] A. Nicolas. Carmichael number square free, 2016. Last accessed 18 November 2018.

[13] R. Solovay and V. Strassen. A fast monte-carlo test for primality. *SIAM journal on Computing*, 6(1):84–85, 1977.

[14] wikipedia. Carmichael number, 2013. Last accessed 18 November 2018.

[15] wikipedia. Solovay–strassen primality test, 2013. Last accessed 18 November 2018.