The Perfect Number Theorem

Mark IJbema

September 19, 2015

Abstract

This document presents the formal proof of the Perfect Number Theorem. The result can also be found as number 70 on the list of “top 100 mathematical theorems” [Wie]. This document was produced as result of a B.Sc. Thesis under supervision of Jaap Top and Wim H. Hesselink (University of Groningen) in 2009.

Contents

1 Basics needed 1

2 Sum of divisors function 3

3 Perfect Number Theorem 7

1 Basics needed

theory PerfectBasics
imports Main ~~/src/HOL/Number-Theory/Primes ~~/src/HOL/Algebra/Exponent
begin

lemma setsum-mono2-nat: finite (B::nat set) ==> A <= B ==> \sum A <= \sum B
proof (auto simp add: setsum-mono2)

lemma seteq-imp-setsumeq: A=B ==> \sum A = \sum B by simp

lemma exp-is-max-div:
  assumes m0: m>0 and p: prime p
  shows ~ p dvd (m div (p ^ (exponent p m)))
proof (rule contr)
  assume ~ ~ p dvd (m div (p ^ (exponent p m)))
  hence a:p dvd (m div (p ^ (exponent p m))) by auto
from m0 have p ^ (exponent p m) dvd m by (auto simp add: power-exponent-dvd)
with a have p^(exponent p m) dvd m
  by (metis (full_types) div-dvd-div div-mult-self2-is-id div-triv-right neq0-cone p)
zero-less-prime-power)
with \( p \) have \( m=0 \) by (auto simp add: power-Suc-exponent-Not-dvd)
with \( \bar{n} \) show False by auto
qed

lemma coprime-exponent:
assumes \( p:prime \) and \( m:m>0 \)
shows coprime \( p \) (\( \bar{m} \div (p^\bar{\text{exponent}} p m) \))
proof (rule ccontr)
  assume \( \neg \) coprime \( p \) (\( \bar{m} \div p \neg \text{exponent} p m \))
  hence \( \exists \, q. \, \text{prime} q \& q \text{ dvd} p \& q \text{ dvd} \) \( (\bar{m} \div (p^\bar{\text{exponent}} p m)) \)
  by (metis dvd.dual-order.refl p prime-imp-coprime-nat)
  hence \( \exists \, q. \, q = p \& q \text{ dvd} \) \( (\bar{m} \div (p^\bar{\text{exponent}} p m)) \)
  by (metis one-not-prime-nat p prime-nat-def)
  hence \( \exists \, q. \, p \text{ dvd} \) \( (\bar{m} \div (p^\bar{\text{exponent}} p m)) \) by auto
  hence \( p \text{ dvd} \) \( (\bar{m} \div (p^\bar{\text{exponent}} p m)) \) by auto
  with \( p \ m \) show False by (auto simp add: exp-is-max-div)
qed

lemma add-mult-distrib-three: \( x::nat)\ast((a+b+c)=x\ast a+x\ast b+x\ast c \)
proof
  have \( x::nat)\ast((a+b+c)=x\ast((a+b)+c) \) by auto
  hence \( x\ast(a+b+c)=x\ast(a+b)+x\ast c \) by (metis add-mult-distrib2 add.commute add.left-commute)
  thus \( x\ast(a+b+c)=x\ast a+x\ast b+x\ast c \) by (metis add-mult-distrib2 add.commute add.left-commute)
qed

lemma nat-interval-minus-zero: \( \{0..Suc n\} = \{0\} \) Un \{Suc 0..Suc n\} by auto
lemma nat-interval-minus-zero2:
assumes \( n>0 \)
shows \( \{0..n\} = \{0\} \) Un \{Suc 0..n\} by (auto simp add: nat-interval-minus-zero)

theorem simplify-sum-of-powers: \( x - 1::nat)\ast((\sum i=0 .. n \cdot \cdot i) = x\cdot(n + 1) \)
\(- \, l \) (is \( ?l = ?r) \)
proof (cases)
  assume \( n = 0 \)
  thus \( ?l = x\cdot(n + 1) - 1 \) by auto
next
  assume \( n>0 \)
  hence \( n\geq 0 \) by auto
  have \( \forall l = (x::nat)\ast((\sum i=0 .. n \cdot \cdot i) - (\sum i=0 .. n \cdot \cdot i) \)
  by (metis diff-mult-distrib nat-mult-1)
  also have \( ... = (\sum i=0 .. n \cdot \cdot (Suc i) \) - (\sum i=0 .. n \cdot \cdot i) \)
  by (simp add: setsum-right-distrib)
  also have \( ... = (\sum i=Suc 0 .. Suc n \cdot \cdot i) \) - (\sum i=0 .. n \cdot \cdot i) \)
  by (metis setsum-shift-bounds-cl-Suc-ivl)
  also with \( \bar{n} \)
  have \( ... = ((\sum i=Suc 0 .. n \cdot \cdot (Suc n)) - (x\cdot0 + (\sum i=Suc 0 .. n \cdot \cdot i)) \)

2
by (auto simp add: setsum.union-disjoint nat-interval-minus-zero2)
finally show ?thesis by auto
qed
end

2 Sum of divisors function

theory Sigma
imports PerfectBasics
begin

definition divisors :: nat => nat set where
divisors (m::nat) == {n . n dvd m}
definition sigma :: nat => nat where
sigma m == \( \sum n | n dvd m . n \)

lemma sigma-divisors[auto simp divisors-def]: sigma(n) = \( \sum (divisors(n)) \)

lemma divisors-eq-dvd iff: (a:divisors(n)) = (a dvd n)
by (unfold divisors-def dvd-def)

lemma mult-divisors[auto simp divisors-def]: (a::nat)*b=c===>a: divisors c
by (unfold divisors-def dvd-def,blast)

lemma mult-divisors2[auto simp divisors-def]: (a::nat)*b=c===>b: divisors c
by (unfold divisors-def dvd-def,auto)

lemma divisorsfinite[simp]:
assumes n>0
shows finite (divisors n)
proof -
  from assms have divisors n = \{m . m dvd n & m <= n\}
    by (auto simp only:divisors-def dvd-imp-le)
hence divisors n <= \{m . m<=n\} by auto
thus finite (divisors n)
  by (metis finite-Collect-le-nat finite-subset)
qed

lemma divs-of-zero-UNIV[simp]: divisors(0) = UNIV
by(auto simp add: divisors-def)

lemma sigma0[simp]: sigma(0) = 0
by (simp add: sigma-def)
lemma sigma1[simp]: sigma(1) = 1
by (simp add: sigma-def)

lemma prime-divisors: prime (p::nat) <-> divisors p = \{1,p\} & p>1
by (auto simp add: divisors-def prime-nat-def)

lemma prime-imp-sigma: prime (p::nat) ==> sigma(p) = p+1
proof -
  assume prime (p::nat)
  hence p>1 & divisors(p) = {1,p} by (simp add: prime-divisors)
  hence p>1 & sigma(p) = \sum {1,p} by (auto simp only: sigma-divisors divisors-def)
  thus sigma(p) = p+1 by simp
qed

lemma sigma-third-divisor:
  assumes 1 < a a < n a : divisors n
  shows 1+a+n <= sigma(n)
proof -
  from assms have finite {1,a,n} & finite (divisors n) & {1,a,n} <= divisors n
  by auto
  hence \sum {1,a,n} <= \sum (divisors n) by (simp only: setsum-mono2)
  hence \sum {1,a,n} <= sigma n by (simp add: sigma-divisors)
  with assms show ?thesis by auto
qed

lemma sigma-imp-divisors: sigma(n)=n+1 ==> n>1 & divisors n = \{n,1\}
proof
  assume ass:sigma(n)=n+1
  hence n\neq 0 & n\neq 1
    by (metis Suc-eq-plus1 n-not-Suc-n sigma0 sigma1)
  thus concl: n>1 by simp

show divisors n = \{n,1\}
proof (rule ccontr)
  assume divisors n \neq \{n,1\}
  with concl have divisors n \neq \{n,1\} & 1<n by auto
  moreover
  from ass concl have 1 : divisors(n) & n : divisors n & ~0 : divisors n
    by (simp add: dvd-def divisors-def)
  ultimately
  have (\exists a. a\neq n & a\neq 1 & 1<n & a : divisors n) & 0 ~: divisors n by auto
  hence (\exists a. a\neq n & a\neq 1 & 1<n & a\neq 0 & a : divisors n) by metis
  hence (\exists a. a\neq n & a\neq 1 & 1\neq n & a\neq 0 & finite \{1,a,n\} & finite (divisors n)
    & \{1,a,n\} \leq divisors n) by auto
  hence (\exists a. a\neq n & a\neq 1 & 1\neq n & a\neq 0 & \sum \{1,a,n\} \leq sigma n
    by (metis setsum-mono2-nat sigma-divisors)
  hence (\exists a. a\neq 0 & (1+a+n) \leq sigma n) by auto
  hence 1+n<sigma n by auto
  with ass show False by auto
qed

qed
lemma sigma-imp-prime: sigma(n) = n + 1 ==> prime n
proof
  assume ass: sigma(n) = n + 1
  hence n > 1 & divisors(n) = {1, n} by (metis insert-commute sigma-imp-divisors)
  thus prime n by (simp add: prime-divisors)
qed

lemma pr-pow-div-eq-sm-pr-pow:
  fixes p :: nat
  assumes prime: prime p
  shows \{ d . d dvd p^n \} = \{ p^f | f . f <= n \}
proof
  show \{ p^f | f . f <= n \} <= \{ d . d dvd p^n \}
  proof
    fix x
    assume x: \{ p^f | f . f <= n \}
    hence \exists i . x = p^i & i <= n by auto
    with prime have x dvd p^n
      by (metis le-imp-power-dvd)
    thus x : \{ d . d dvd p^n \} by auto
  qed
  next
  show \{ d . d dvd p^n \} <= \{ p^f | f . f <= n \}
  proof
    fix x
    assume x: \{ d . d dvd p^n \}
    hence x dvd p^n by auto
    with prime obtain i where i <= n & x = p^i using prime-dvd-power-nat-iff
    prime-dvd-power-nat
      by (auto simp only: divides-primepow)
    hence x = p^i & i <= n by auto
    thus x : \{ p^f | f . f <= n \} by auto
  qed
qed

lemma rewrite-sum-of-powers:
  assumes p: (p::nat)>1
  shows (\sum { p^m | m . m<=(n::nat) } ) = (\sum i = 0 .. n . p^i) (is ?l = ?r)
proof
  have ?l = setsum (%x. x) ((op ^ p) m |m . m<= n) by auto
  also have ... = setsum (%x. x) ((op ^ p)"{m . m<= n})
    by (rule seteq-imp-setsumeq auto)
  moreover with p have inj-on (op ^ p) \{ m . m<=n \}
    by (simp add: inj-on-def)
  ultimately have ?l = setsum (op ^ p) \{ m . m<=n \}
    by (simp add: setsum.reindex)
  moreover have \{m::nat . m<=n \} = \{0..n \} by auto
  ultimately show ?l = (\sum i = 0 .. n . p^i) by auto
qed
theorem sigma-primepower:
prime p ==\> (p - 1) * sigma(p (e::nat)) = (p (e+1) - 1)
proof
  assume prime p
  hence sigma(p (e::nat)) = (\sum i=0 .. e . p^i)
    by (simp add: pr-pow-div-eq-sm-pr-pow sigma-def rewrite-sum-of-powers prime-nat-def)
  thus (p - 1) * sigma(p^e) = p^(e+1) - 1 by (simp only: simplify-sum-of-powers)
qed

lemma sigma-prime-power-two: sigma(2^(n::nat)) = 2^(n+1) - 1
proof
  have (2 - 1) * sigma(2^(n::nat)) = 2^(n+1) - 1
    by (auto simp only: sigma-primepower two-is-prime-nat)
  thus \?thesis by simp
qed

lemma prodsums-eq-sumprods:
  fixes p :: nat and m :: nat
  assumes coprime p m
  shows (\sum {p^f | f. f < n & b dvd m}) * (\sum {b^f | f. b dvd m}) = (\sum {p^f * b^f | f. f < n & b dvd m})
proof
  have ALL x f. x dvd m \\> coprime (p^f) x
    by (metis assms coprime-exp-nat gcd-1-nat gcd-nat.absorb-iff1 gcd-commute gcd-semilattice-nat.inf-left-commute)
  thus \?thesis
    by (auto simp add: divides-primepow)
 qed

declare [[simproc add: finite-Collect]]

lemma rewrite-for-sigma-semimultiplicative:
  fixes p :: nat
  assumes prime p
  shows \{p^f * b | f. f < n & b dvd m\} = \{a*b | a b. a dvd (p^n) & b dvd m\}
proof
  show \{p^f * b | f. f < n & b dvd m\} <= \{a*b | a b. a dvd p^n & b dvd m\}
  proof
    fix x
    assume x: \{p^f * b \| f. f < n & b dvd m\}
    then obtain b f where x = p^f * b & f < n & b dvd m by auto
    with (prime p) show x: \{a*b | a b. a dvd p^n & b dvd m\}
      by (auto simp add: divides-primepow)
  qed

next
  show \{a*b | a b. a dvd p^n & b dvd m\} <= \{p^f * b | f. f < n & b dvd m\}
    using (prime p) by auto (metis assms divides-primepow)
qed
lemma div-decomp-comp:
  fixes a :: nat
  shows coprime m n =⇒ a dvd m * n ←→ (∃ b c. a = b * c & b dvd m & c dvd n)
by (auto simp only: division-decomp-nat mult-dvd-mono)

theorem sigma-semimultiplicative:
  assumes p: prime p and cop: coprime p m
  shows sigma (p^n) * sigma m = sigma (p^n * m) (is ?l = ?r)
proof –
  from cop have cop2: coprime (p^n) m
    by (auto simp add: coprime-exp-nat gcd-commute-nat)
  have ?l = (∑ {a. a dvd p^n}) * (∑ {b. b dvd m})
    by (simp add: sigma-def)
  also from p have ... = (∑ {p^f | f. f <= n}) * (∑ {b. b dvd m})
    by (simp add: pr-pow-div-eq-sm-pr-pow)
  also have ... = (∑ {p^f * b | f, b dvd m})
    by (auto simp add: prodsums-eq-sumprods prime-nat-def)
  also have ... = (∑ {a*b | a dvd p^n & b dvd m})
    by (auto simp add: seteq-imp-setsumeq, rule rewrite-for-sigma-semimultiplicative[OF p])
  finally have ?l = (∑ {c. c dvd (p^n*m)})
    by (auto simp add: div-decomp-comp[OF cop2])
  thus ?l = sigma (p^n*m) by (auto simp add: sigma-def)
qed

end

3 Perfect Number Theorem

theory Perfect
imports Sigma
begin

definition perfect :: nat ⇒ bool where
  perfect m = m > 0 & 2*m = sigma m

theorem perfect-number-theorem:
  assumes even: even m and perfect: perfect m
  shows ∃ n. m = 2^n * (2^(n+1) - 1) ∧ prime (2^n * (n+1) - 1)
proof
  from perfect have m0: m > 0 by (auto simp add: perfect-def)
  let ?n = exponent 2 m
  let ?A = m div 2^?n
  let ?np = (2::nat)^(?n+1) - 1
  from even m0 have n1: ?n >= 1 by (simp add: exponent-ge)
  from m0 have 2^?n dvd m by (rule power-exponent-dvd)
  hence m = 2^?n * ?A by (simp only: dvd-mult-div-cancel)
with m0 have mdef: \( m = 2^n \cdot n \cdot A \) & coprime \( 2^n \cdot A \)
by (simp add: coprime-exponent)

moreover with m0 have a0: \(?A > 0\) by (metis nat-0-less-mult-iff)

moreover
{ from perfect have 2\cdot m = sigma(m) by (simp add: perfect-def)
  with mdef have 2 \cdot (n+1) \cdot A = sigma(2^n \cdot A) by auto
}
ultimately have 2 \cdot (n+1) \cdot A = (\?np) \cdot sigma(?A)
by (simp add: sigma-semimultiplicative)

hence formula: 2 \cdot (n+1) \cdot A = (\?np) \cdot sigma(?A)
by (simp only: sigma-prime-power-two)

from n1 have (2::nat)^((n+1)) >= 2\cdot 2 by (simp only: power-increasing)

hence nplarger: \(?np >= 3\) by auto

let \(?B = ?A \div ?np\)

from formula have \(?np \ dvd \ ?A \cdot 2 \cdot (n+1)\)
by (metis mult_commute dvd-def)

then have \(?np \ dvd \ ?A\)
using coprime-minus-one-nat [of \( 2 \cdot (exponent \ 2 \cdot m + 1) \)]
by (auto intro: coprime-dvd-mult-nat)


with a0 have b0: \(?B > 0\) by (metis gr0I mult-is-0)

from nplarger a0 have bsmallera: \(?B < ?A\) by auto

have \(?B = 1\)
proof (rule ccontr)
  assume \( \sim \?B = 1\)
  with b0 bsmallera have 1<?B \(?B?<?A by auto
  moreover from bdef have \(?B : \ divisors \ ?A\) by (rule mult-divisors2)

  ultimately have 1+?B+?A <= sigma ?A by (rule sigma-third-divisor)
  with nplarger have \(?np+1+?A+?B\) <= \(?np\cdot\sigma\ ?A\)
  by (auto simp only: nat-mult-le-cancel1)
  with bdef have \(?np+?A\cdot?np + ?A\cdot1 <= \(?np\cdot\sigma\ ?A\)
  by (simp only: add-mult-distrib-three mult_commute)

  hence \(?np+?A\cdot(?np + 1) <= \(?np\cdot\sigma\ ?A\) by (simp only: add-mult-distrib2)
  with nplarger have 2\cdot(\?n+1)\cdot?A <= \(?np\cdot\sigma\ ?A\) by (simp add: mult_commute)

  with formula show False by auto
qed

with bdef have adef: \( ?A = \?np \) by auto

with formula have \(?np\cdot 2 \cdot (n+1) = (\?np) \cdot \sigma\ ?A\) by auto

with nplarger adef have \(?A + 1 = \sigma\ ?A\) by auto

with a0 have prime ?A by (simp add: sigma-imp-prime)

with mdef adef show m = 2^n \cdot (\?np) \& prime ?np by simp
qed

theorem Euclid-book9-prop36:
assumes $p$: prime $(2^n + 1) - (1 :: \text{nat})$
shows perfect $(2^n)(2^n + 1) - 1$
proof (unfold perfect-def, auto)
from assms show $(2 :: \text{nat}) * 2^n > \text{Suc 0}$ by (auto simp add: prime-nat-def)
next
  have $2 \sim = ((2 :: \text{nat})^2 + 1) - 1$ by simp arith
  then have coprime $(2 :: \text{nat}) (2^n + 1) - 1$
    by (metis primes-coprime-nat two-is-prime-nat)
moreover with $p$ have $2^n + 1 - 1 > (0 :: \text{nat})$
  by (auto simp add: prime-nat-def)
ultimately have $\sigma (2^n * (2^n + 1) - 1) = (\sigma (2^n)) * (\sigma (2^n + 1) - 1)$
  by (metis sigma-semimultiplicative two-is-prime-nat)
also from assms have $\ldots = (\sigma \tau (2^n)) * (2^n + 1)$
  by (auto simp add: prime-imp-sigma)
also have $\ldots = (2^n + 1 - 1) * (2^n + 1)$ by (simp add: sigma-prime-power-two)
finally show $2 * (2^n * (2 * 2^n - \text{Suc 0})) = \sigma (2^n * (2 * 2^n - \text{Suc 0}))$ by auto
qed

end

References