

Lifting the Exponent

Maya Kądziołka

October 13, 2025

Abstract

We formalize the *Lifting the Exponent Lemma*, which shows how to find the largest power of p dividing $a^n \pm b^n$, for a prime p and positive integers a and b . The proof follows [1].

Contents

1	Library additions	1
2	The $p > 2$ case	3
3	The $p = 2$ case	7
theory <i>LTE</i>		
imports		
<i>HOL-Number-Theory.Number-Theory</i>		
begin		

1 Library additions

lemma *cong-sum-mono-neutral-right*:

assumes *finite T*

assumes $S \subseteq T$

assumes *zeros*: $\forall i \in T - S. [g\ i = 0] \pmod n$

shows $[sum\ g\ T = sum\ g\ S] \pmod n$

proof –

have $[sum\ g\ T = (\sum_{x \in T}. if\ x \in S\ then\ g\ x\ else\ 0)] \pmod n$

using *zeros* **by** (*auto intro: cong-sum*)

also have $(\sum_{x \in T}. if\ x \in S\ then\ g\ x\ else\ 0) = (\sum_{x \in S}. if\ x \in S\ then\ g\ x\ else\ 0)$

by (*intro sum.mono-neutral-right; fact?; auto*)

also have $\dots = sum\ g\ S$

by (*auto intro: sum.cong*)

finally show *?thesis*.

qed

lemma *power-odd-inj*:

fixes $a\ b :: 'a :: linordered-idom$

```

    assumes odd k and  $a^k = b^k$ 
    shows  $a = b$ 
  proof (cases  $a \geq 0$ )
    case True
    then have  $b \geq 0$ 
      using assms zero-le-odd-power by metis
    moreover from  $\langle \text{odd } k \rangle$  have  $k > 0$  by presburger
    show ?thesis
      by (rule power-eq-imp-eq-base; fact)
  next
    case False
    then have  $b < 0$ 
      using assms power-less-zero-eq not-less by metis
    from  $\langle a^k = b^k \rangle$  have  $(-a)^k = (-b)^k$ 
      using  $\langle \text{odd } k \rangle$  power-minus-odd by simp
    moreover have  $-a \geq 0$  and  $-b \geq 0$ 
      using  $\langle \neg a \geq 0 \rangle$  and  $\langle b < 0 \rangle$  by auto
    moreover from  $\langle \text{odd } k \rangle$  have  $k > 0$  by presburger
    ultimately have  $-a = -b$  by (rule power-eq-imp-eq-base)
    then show ?thesis by simp
  qed

```

```

lemma power-eq-abs:
  fixes  $a\ b :: 'a::linordered-idom$ 
  assumes  $a^k = b^k$  and  $k > 0$ 
  shows  $|a| = |b|$ 
  proof -
    from  $\langle a^k = b^k \rangle$  have  $|a|^k = |b|^k$ 
      using power-abs by metis
    show  $|a| = |b|$ 
      by (rule power-eq-imp-eq-base; fact?; auto)
  qed

```

```

lemma cong-scale:
   $k \neq 0 \implies [a = b] \text{ (mod } c) \longleftrightarrow [k*a = k*b] \text{ (mod } k*c)$ 
  unfolding cong-def by auto

```

```

lemma odd-square-mod-4:
  fixes  $x :: int$ 
  assumes odd x
  shows  $[x^2 = 1] \text{ (mod } 4)$ 
  proof -
    have  $x^2 - 1 = (x - 1) * (x + 1)$ 
      by (simp add: ring-distrib power2-eq-square)
    moreover from  $\langle \text{odd } x \rangle$  have  $2 \text{ dvd } x - 1$  and  $2 \text{ dvd } x + 1$ 
      by auto
    ultimately have  $4 \text{ dvd } x^2 - 1$ 
      by fastforce
    thus ?thesis

```

by (*simp add: cong-iff-dvd-diff*)
qed

2 The $p > 2$ case

context

fixes $x\ y :: \text{int}$ and $p :: \text{nat}$
assumes *prime* p
assumes $p \text{ dvd } x - y$
assumes $\neg p \text{ dvd } x \quad \neg p \text{ dvd } y$
begin

lemma *decompose-mod-p*:

$[(\sum_{i < n}. y^{(n - \text{Suc } i)} * x^i) = n * x^{(n-1)}] \text{ (mod } p)$
proof –
{
fix i
assume $i < n$
from $\langle p \text{ dvd } x - y \rangle$ **have** $[x = y] \text{ (mod } p)$
by (*simp add: cong-iff-dvd-diff*)
hence $[y^{(n - \text{Suc } i)} * x^i = x^{(n - \text{Suc } i)} * x^i] \text{ (mod } p)$
by (*intro cong-scalar-right cong-pow; rule cong-sym*)
also have $x^{(n - \text{Suc } i)} * x^i = x^{(n - 1)}$
using $\langle i < n \rangle$ **by** (*simp flip: power-add*)
finally have $[y^{(n - \text{Suc } i)} * x^i = x^{(n - 1)}] \text{ (mod } p)$
by *auto*
}
hence $[(\sum_{i < n}. y^{(n - \text{Suc } i)} * x^i) = (\sum_{i < n}. x^{(n-1)})] \text{ (mod } p)$
by (*intro cong-sum; auto*)
thus $[(\sum_{i < n}. y^{(n - \text{Suc } i)} * x^i) = n * x^{(n-1)}] \text{ (mod } p)$
by *simp*
qed

Lemma 1:

lemma *multiplicity-diff-pow-coprime*:

assumes *coprime* $p\ n$
shows *multiplicity* $p\ (x^n - y^n) = \text{multiplicity } p\ (x - y)$
proof –
have *factor*: $x^n - y^n = (\sum_{i < n}. y^{(n - \text{Suc } i)} * x^i) * (x - y)$
by (*simp add: power-diff-sumr2*)
moreover have $\neg p \text{ dvd } (\sum_{i < n}. y^{(n - \text{Suc } i)} * x^i)$
proof
assume $p \text{ dvd } (\sum_{i < n}. y^{(n - \text{Suc } i)} * x^i)$
with *decompose-mod-p* **have** $p \text{ dvd } n * x^{(n-1)}$
using *cong-dvd-iff* **by** *blast*
with $\langle \text{prime } p \rangle$ **have** $p \text{ dvd } n \vee p \text{ dvd } x^{(n-1)}$
by (*simp add: prime-dvd-mult-eq-int*)
moreover from $\langle \text{coprime } p\ n \rangle$ **and** $\langle \text{prime } p \rangle$ **have** $\neg p \text{ dvd } n$
using *coprime-absorb-right not-prime-unit* **by** *auto*

```

ultimately have  $p \mid x^{(n-1)}$ 
  by simp
hence  $p \mid x$ 
  using  $\langle \text{prime } p \rangle$  prime-dvd-power-int prime-nat-int-transfer by blast
with  $\langle \neg p \mid x \rangle$  show False by simp
qed
ultimately show  $\text{multiplicity } p (x^n - y^n) = \text{multiplicity } p (x - y)$ 
  using  $\langle \text{prime } p \rangle$ 
  by (auto intro: multiplicity-prime-elem-times-other)
qed

```

The inductive step:

```

lemma multiplicity-diff-self-pow:
  assumes  $p > 2$  and  $x \neq y$ 
  shows  $\text{multiplicity } p (x^p - y^p) = \text{Suc } (\text{multiplicity } p (x - y))$ 
proof -
  have *:  $\text{multiplicity } p (\sum_{i < p} y^{(p - \text{Suc } i)} * x^i) = 1$ 
  proof (rule multiplicity-eqI)
    have  $[(\sum_{t < p} y^{(p - \text{Suc } t)} * x^t) = p * x^{(p-1)}] \pmod p$ 
      by (rule decompose-mod-p)
    also have  $[p * x^{(p-1)} = 0] \pmod p$ 
      by (simp add: cong-mult-self-left)
    finally show  $(\text{int } p)^1 \mid (\sum_{i < p} y^{(p - \text{Suc } i)} * x^i)$ 
      by (simp add: cong-0-iff)

  from  $\langle p \mid x - y \rangle$  obtain  $k::\text{int}$  where  $kp: x = y + k * p$ 
  by (metis add.commute diff-add-cancel dvd-def mult.commute)

  have  $[y^{(p - \text{Suc } t)} * x^t = y^{(p-1)} + t * k * p * y^{(p-2)}] \pmod{p^2}$  if  $t < p$ 
for  $t$ 
  proof (cases  $t = 0$ )
    case False
    have  $y^{(p - \text{Suc } t)} * x^t = y^{(p - \text{Suc } t)} * (y + k * p)^t$ 
      unfolding kp..
    also have  $\dots = y^{(p - \text{Suc } t)} * (\sum_{i \leq t} (t \text{ choose } i) * (k * p)^i * y^{(t-i)})$ 
      by (simp flip: binomial-ring add: add.commute)
    also have  $[\dots = y^{(p - \text{Suc } t)} * (\sum_{i \leq 1} (t \text{ choose } i) * (k * p)^i * y^{(t-i)})]$ 
      (mod p^2)
      — discard  $i > 1$ 
    proof (intro cong-scalar-left cong-sum-mono-neutral-right; rule)
      fix  $i$ 
      assume  $i \in \{..t\} - \{..1\}$ 
      then have  $i \geq 2$  by simp
      then obtain  $i'$  where  $i = i' + 2$ 
        using add.commute le-Suc-ex by blast
      hence  $(k * p)^i = (k * p)^{i'} * k^2 * p^2$ 
        by (simp add: ac-simps power2-eq-square)
      hence  $[(k * p)^i = 0] \pmod{p^2}$ 
        by (simp add: cong-mult-self-right)

```

```

    thus [(t choose i) * (k*p)^(i-1) * y^(t-i) = 0] (mod p^2)
    by (simp add: cong-0-iff)
  qed (use <t ≠ 0> in auto)
  also have (∑ i≤1. (t choose i) * (k*p)^(i-1) * y^(t-i)) = y^t + t*k*p*y^(t-1)
    by simp
  also have y^(p - Suc t) * ... = y^(p-1) + t*k*p*y^(p-2)
    using <t < p> <t ≠ 0> by (auto simp add: algebra-simps numeral-eq-Suc
simp flip: power-add)
  finally show ?thesis.
qed simp

  hence [(∑ t<p. y^(p - Suc t) * x^t) = (∑ t<p. y^(p-1) + t*k*p*y^(p-2))]
(mod p^2)
  by (auto intro: cong-sum)
  also have (∑ t<p. y^(p-1) + t*k*p*y^(p-2)) = p*y^(p-1) + (∑ t<p. t) *
k*p*y^(p-2)
  by (simp add: sum.distrib sum-distrib-right)
  also have (∑ t<p. t) = p*(p - 1) div 2
  by (simp add: Sum-Ico-nat lessThan-atLeast0)
  finally have [(∑ t<p. y^(p - Suc t) * x^t) = p*y^(p-1) + (p*(p - 1) div 2)
* k*p*y^(p-2)] (mod p^2).
  moreover have [(p*(p - 1) div 2) * k*p*y^(p-2) = 0] (mod p^2)
  proof -
    have [(p * (p - 1) div 2) * p = 0] (mod p^2)
  proof -
    from <p > 2> and <prime p> have odd p
    using prime-odd-nat by blast
    thus ?thesis
  by (metis (no-types, lifting) cong-0-iff div-mult-swap dvd-times-left-cancel-iff
dvd-triv-left le-0-eq linorder-not-less mult.commute odd-pos odd-two-times-div-two-nat
one-add-one power-add power-one-right)
  qed
  hence [int ((p*(p - 1) div 2) * p)*k*y^(p-2) = 0] (mod p^2)
    unfolding cong-0-iff using int-dvd-int-iff by fastforce
  thus ?thesis
    by (simp add: ac-simps)
  qed
  ultimately have [(∑ t<p. y^(p - Suc t) * x^t) = p*y^(p-1)] (mod p^2)
    using cong-add-lcancel-0 cong-trans by blast
  moreover have ¬ p^2 dvd p*y^(p-1)
    using <p > 2> <prime p> <¬ p dvd y> by (simp add: power2-eq-square
prime-dvd-power-int-iff)
  ultimately show ¬ int p^(Suc 1) dvd (∑ t<p. y^(p - Suc t) * x^t)
    by (metis (no-types, lifting) Suc-1 of-nat-power cong-dvd-iff)
  qed
  moreover have multiplicity p (x^p - y^p) = multiplicity p (x - y) + multiplicity
p (∑ i<p. y^(p - Suc i) * x^i)
  apply (unfold power-diff-sumr2, intro prime-elem-multiplicity-mult-distrib)
  using <prime p> <x ≠ y> multiplicity-zero * by auto

```

ultimately show *?thesis* by *simp*
 qed

Theorem 1:

theorem *multiplicity-diff-pow*:

assumes $p > 2$ and $x \neq y$ and $n > 0$

shows $\text{multiplicity } p (x^n - y^n) = \text{multiplicity } p (x - y) + \text{multiplicity } p n$

proof –

obtain k where $n = p^{\text{multiplicity } p n} * k$ and $\neg p \text{ dvd } k$

using $\langle n > 0 \rangle$ $\langle \text{prime } p \rangle$

by (*metis neq0-conv not-prime-unit multiplicity-decompose'*)

have $\text{multiplicity } p (x^{p^a * k} - y^{p^a * k}) = \text{multiplicity } p (x - y) + a$

for a

proof (*induction a*)

case 0

from $\langle \neg p \text{ dvd } k \rangle$ have *coprime* $p k$

using $\langle \text{prime } p \rangle$ by (*intro prime-imp-coprime*)

thus *?case*

by (*simp add: multiplicity-diff-pow-coprime*)

next

case (*Suc a*)

let $?x' = x^{p^a * k}$ and $?y' = y^{p^a * k}$

have $\neg p \text{ dvd } ?x'$ and $\neg p \text{ dvd } ?y'$

using $\langle \neg p \text{ dvd } x \rangle$ $\langle \neg p \text{ dvd } y \rangle$ and $\langle \text{prime } p \rangle$

by (*meson prime-dvd-power prime-nat-int-transfer*)

moreover have $p \text{ dvd } ?x' - ?y'$

using $\langle p \text{ dvd } x - y \rangle$ by (*simp add: power-diff-sumr2*)

moreover have $?x' \neq ?y'$

proof

assume $?x' = ?y'$

moreover have $0 < p^a * k$

using $\langle \text{prime } p \rangle$ $\langle n > 0 \rangle$ n

by (*metis gr0I mult-is-0 power-not-zero prime-gt-0-nat*)

ultimately have $|x| = |y|$

by (*intro power-eq-abs*)

with $\langle x \neq y \rangle$ have $x = -y$

using *abs-eq-iff* by *simp*

with $\langle p \text{ dvd } x - y \rangle$ have $p \text{ dvd } 2*x$

by *simp*

with $\langle \text{prime } p \rangle$ have $p \text{ dvd } 2 \vee p \text{ dvd } x$

by (*metis int-dvd-int-iff of-nat-numeral prime-dvd-mult-iff prime-nat-int-transfer*)

with $\langle p > 2 \rangle$ have $p \text{ dvd } x$

by *auto*

with $\langle \neg p \text{ dvd } x \rangle$ show *False..*

qed

moreover have $p^{\text{Suc } a} * k = p^a * k * p$

by (*simp add: ac-simps*)

ultimately show *?case*

using *LTE.multiplicity-diff-self-pow* [where $x=?x'$ and $y=?y'$, OF $\langle \text{prime } p \rangle$]

```

⟨p > 2⟩
  and Suc.IH
  by (metis add-Suc-right power-mult)
qed
with n show ?thesis by metis
qed

end

```

Theorem 2:

```

corollary multiplicity-add-pow:
  fixes x y :: int and p n :: nat
  assumes odd n
    and prime p and p > 2
    and p dvd x + y and ¬ p dvd x ¬ p dvd y
    and x ≠ -y
  shows multiplicity p (x^n + y^n) = multiplicity p (x + y) + multiplicity p n
proof -
  have [simp]: (-y)^n = -(y^n)
    using ⟨odd n⟩ by (rule power-minus-odd)
  moreover have n > 0
    using ⟨odd n⟩ by presburger
  with assms show ?thesis
    using multiplicity-diff-pow[where x=x and y=-y and n=n]
    by simp
qed

```

3 The $p = 2$ case

Theorem 3:

```

theorem multiplicity-2-diff-pow-4div:
  fixes x y :: int
  assumes odd x odd y and 4 dvd x - y and n > 0 x ≠ y
  shows multiplicity 2 (x^n - y^n) = multiplicity 2 (x - y) + multiplicity 2 n
proof -
  have prime (2::nat) by simp
  then obtain k where n = 2^k * multiplicity 2 n * k and ¬ 2 dvd k
    using ⟨n > 0⟩
    by (metis neq0-conv not-prime-unit multiplicity-decompose')

  have pow2: multiplicity 2 (x^(2^k) - y^(2^k)) = multiplicity 2 (x - y) + k for
    k
  proof (induction k)
  case (Suc k)
  have x^(2^(Suc k)) - y^(2^(Suc k)) = (x^(2^k))^2 - (y^(2^k))^2
    by (simp flip: power-mult algebra-simps)
  also have ... = (x^(2^k) - y^(2^k)) * (x^(2^k) + y^(2^k))
    by (simp add: power2-eq-square algebra-simps)

```

finally have factor: $x^{2^{\text{Suc } k}} - y^{2^{\text{Suc } k}} = (x^{2^k} - y^{2^k}) * (x^{2^k} + y^{2^k})$.
 moreover have *m-plus*: $\text{multiplicity } 2 \ (x^{2^k} + y^{2^k}) = 1$
 proof (rule *multiplicity-eqI*)
 show $2^1 \text{ dvd } x^{2^k} + y^{2^k}$
 using $\langle \text{odd } x \rangle$ and $\langle \text{odd } y \rangle$ by *simp*

 have $[x^{2^k} + y^{2^k} = 2] \ (\text{mod } 4)$
 proof (cases k)
 case 0
 from $\langle \text{odd } y \rangle$ have $[y = 1] \ (\text{mod } 2)$
 using *cong-def* by *fastforce*
 hence $[2 * y = 2] \ (\text{mod } 4)$
 using *cong-scale* [where $k=2$ and $b=1$ and $c=2$, *simplified*] by *force*
 moreover from $\langle 4 \text{ dvd } x - y \rangle$ have $[x - y = 0] \ (\text{mod } 4)$
 by (*simp add: cong-0-iff*)
 ultimately have $[x + y = 2] \ (\text{mod } 4)$
 by (*metis add.commute assms(3) cong-add-lcancel cong-iff-dvd-diff cong-trans mult-2*)
 with $\langle k = 0 \rangle$ show ?thesis by *simp*
 next
 case (*Suc k'*)
 then have $[x^{2^k} = 1] \ (\text{mod } 4)$ and $[y^{2^k} = 1] \ (\text{mod } 4)$
 using $\langle \text{odd } x \rangle$ $\langle \text{odd } y \rangle$
 by (*auto simp add: power-mult power-Suc2 simp del: power-Suc intro: odd-square-mod-4*)
 thus $[x^{2^k} + y^{2^k} = 2] \ (\text{mod } 4)$
 using *cong-add* by *fastforce*
 qed
 thus $\neg 2^{\text{Suc } 1} \text{ dvd } x^{2^k} + y^{2^k}$
 by (*simp add: cong-dvd-iff*)
 qed
 moreover have $x^{2^k} + y^{2^k} \neq 0$
 using *m-plus multiplicity-zero* by *auto*
 moreover have $x^{2^k} - y^{2^k} \neq 0$
 proof
 assume $x^{2^k} - y^{2^k} = 0$
 then have $|x| = |y|$
 by (*intro power-eq-abs, simp, simp*)
 hence $x = y \vee x = -y$
 using *abs-eq-iff* by *auto*
 with $\langle x \neq y \rangle$ have $x = -y$
 by *simp*
 with $\langle 4 \text{ dvd } x - y \rangle$ have $4 \text{ dvd } 2 * x$
 by *simp*
 hence $2 \text{ dvd } x$
 by *auto*
 with $\langle \text{odd } x \rangle$ show *False..*
 qed


```

ultimately have multiplicity 2 (x^2^Suc k - y^2^Suc k) =
  multiplicity 2 (x^2^k - y^2^k) + multiplicity 2 (x^2^k + y^2^k)
  by (unfold factor; intro prime-elem-multiplicity-mult-distrib; auto)
then show ?case
  using m-plus Suc.IH by simp
qed simp

moreover have even-diff: int 2 dvd x^2^multiplicity 2 n - y^2^multiplicity 2 n
  using <odd x> and <odd y> by simp
moreover have odd-parts: ¬ int 2 dvd x^2^multiplicity 2 n - int 2 dvd
y^2^multiplicity 2 n
  using <odd x> and <odd y> by simp+
moreover have coprime: coprime 2 k
  using <¬ 2 dvd k> by simp

show ?thesis
  apply (subst (1) n)
  apply (subst (2) n)
  apply (simp only: power-mult)
  apply (simp only: multiplicity-diff-pow-coprime[OF <prime 2> even-diff odd-parts
coprime, simplified])
  by (rule pow2)
qed

```

Theorem 4:

```

theorem multiplicity-2-diff-even-pow:
  fixes x y :: int
  assumes odd x odd y and even n and n > 0 and |x| ≠ |y|
  shows multiplicity 2 (x^n - y^n) = multiplicity 2 (x - y) + multiplicity 2 (x +
y) + multiplicity 2 n - 1
proof -
  obtain n' where n = 2*n'
  using <even n> by auto
  with <n > 0> have n' > 0 by simp

  moreover have 4 dvd x^2 - y^2
  proof -
    have x^2 - y^2 = (x + y) * (x - y)
    by (simp add: algebra-simps power2-eq-square)
    moreover have 2 dvd x + y and 2 dvd x - y
    using <odd x> and <odd y> by auto
    ultimately show 4 dvd x^2 - y^2 by fastforce
  qed

  moreover have odd (x^2) and odd (y^2)
  using <odd x> <odd y> by auto
  moreover from <|x| ≠ |y|> have x^2 ≠ y^2
  using diff-0 diff-0-right power2-eq-iff by fastforce

```

ultimately have *multiplicity 2* $((x^2)^{n'} - (y^2)^{n'}) = \text{multiplicity 2 } (x^2 - y^2) + \text{multiplicity 2 } n'$
 by (*intro multiplicity-2-diff-pow-4div*)
 also have *multiplicity 2* $((x^2)^{n'} - (y^2)^{n'}) = \text{multiplicity 2 } (x^n - y^n)$
 unfolding $\langle n = 2*n' \rangle$ by (*simp add: power-mult*)
 also have *multiplicity 2* $(x^2 - y^2) = \text{multiplicity 2 } ((x - y) * (x + y))$
 by (*simp add: algebra-simps power2-eq-square*)
 also have $\dots = \text{multiplicity 2 } (x - y) + \text{multiplicity 2 } (x + y)$
 using $\langle |x| \neq |y| \rangle$ by (*auto intro: prime-elem-multiplicity-mult-distrib*)
 also have *multiplicity 2* $n = \text{Suc } (\text{multiplicity 2 } n')$
 unfolding $\langle n = 2*n' \rangle$ using $\langle n' > 0 \rangle$ by (*simp add: multiplicity-times-same*)
 ultimately show *?thesis* by *simp*
 qed
 end

References

- [1] Hossein Parvardi. Lifting The Exponent Lemma (LTE), 2011.
 URL: <https://s3.amazonaws.com/aops-cdn.artofproblemsolving.com/resources/articles/lifting-the-exponent.pdf>.