Euclid’s lemma

Johan G. F. Belinfante
2009 March 27

In[1]:=
```
SetDirectory["l:"]; << goedel.09mar26b; << tools.m
```

```
:Package Title: goedel.09mar26b            2009 March 26 at 2:25 p.m.

It is now:  2009 Mar 27 at 12:57

Loading Simplification Rules

TOOLS.M                                    Revised 2009 February 18

weightlimit = 40
```

---

summary

Proposition 30 in Book 7 of Euclid’s *Elements* states that a prime divisor of a product of two natural numbers must divide at least one of the two factors.

In[2]:=
```
"Sir Thomas L. Heath, The Thirteen Books of Euclid's Elements, Second
Edition, Dover Publications, New York, 1956.  See volume 2, page 331.";
```

In this notebook Art Quaife’s derivation of this theorem, popularly called Euclid's Lemma, has been translated into the language of the **GOEDEL** program.

In[3]:=
```
"Art Quaife, Automated Development of Fundamental Mathematical
Theories, Appendix 3. Theorems Proved in Peano's Arithmetic,
Kluwer Academic Publishers, Dordrecht, 1992. Cf. pp. 202-206";
```


---

relatively prime natural numbers

Linear differences are the key idea in the derivation of Euclid’s lemma. Two natural numbers **x** and **y** are **relatively prime** if their gcd is **1 = set[0]**, which implies **1∈ ld[x, y]**. In this section it is shown that if **x** divides a product **y z** and if **x** is relatively prime to **y**, then **x** divides **z**.   If **x** and **y** are relatively prime, then either **x** is nonzero or **y = 1**. The derivation of this precursor of Euclid’s lemma considers each of these two cases separately. In the case that **x** is nonzero, Quaife’s theorems **(LD17)** and **(LD18)** about linear differences are used. The case that **y =1** is entirely trivial.

Lemma. A special case of Quaife’s theorem **(LD17)**.

```
In[4]:= SubstTest[implies,
          and[member[nat[z], ld[nat[u], nat[v]]], member[pair[nat[x], nat[u]], DIV],
           member[pair[nat[x], nat[v]], DIV]], member[pair[nat[x], nat[z]], DIV],
          {u -> natmul[nat[x], nat[z]], v -> natmul[nat[y], nat[z]]}] // Reverse

Out[4]= or[member[pair[nat[x], nat[z]], DIV],
          not[member[nat[z], ld[natmul[nat[x], nat[z]], natmul[nat[y], nat[z]]]]],
          not[member[pair[nat[x], natmul[nat[y], nat[z]]], DIV]]] == True

In[5]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

Lemma.

```
In[6]:= Map[not,
          SubstTest[and, implies[p2, or[p3, p4]], implies[and[p2, p3], p5], implies[p5, p6],
           implies[and[p1, p6], p7], implies[and[p1, p4], p7], not[implies[and[p1, p2], p7]],
           {p1 → member[pair[nat[x], natmul[nat[y], nat[z]]], DIV], p2 →
             equal[APPLY[GLB[DIV], set[nat[x], nat[y]]], set[0]], p3 → not[equal[0, nat[x]]],
            p4 → equal[nat[y], set[0]], p5 → member[set[0], ld[nat[x], nat[y]]],
            p6 → member[nat[z], ld[natmul[nat[x], nat[z]], natmul[nat[y], nat[z]]]],
            p7 → member[pair[nat[x], nat[z]], DIV]}]] // Reverse

Out[6]= or[member[pair[nat[x], nat[z]], DIV],
          not[equal[APPLY[GLB[DIV], set[nat[x], nat[y]]], set[0]]],
          not[member[pair[nat[x], natmul[nat[y], nat[z]]], DIV]]] == True

In[7]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

All the **nat** wrappers are superfluous.

Theorem. Quaife's theorem **(GCD15)**. Precursor of Euclid's lemma. If a natural number **x** divides a product **y z** of natural numbers, and if **x** and **y** are relatively prime, then **x** divides **z**.

```
In[8]:= SubstTest[implies, and[equal[x, nat[u]], equal[y, nat[v]], equal[z, nat[w]]],
           or[member[pair[x, z], DIV], not[equal[APPLY[GLB[DIV], set[x, y]], set[0]]],
            not[member[pair[x, natmul[y, z]], DIV]]], {u → x, v → y, w → z}] // Reverse // MapNotNot

Out[8]= or[member[pair[x, z], DIV], not[equal[APPLY[GLB[DIV], set[x, y]], set[0]]],
          not[member[pair[x, natmul[y, z]], DIV]]] == True

In[9]:= or[member[pair[x_, z_], DIV], not[equal[APPLY[GLB[DIV], set[x_, y_]], set[0]]],
          not[member[pair[x_, natmul[y_, z_]], DIV]]] := True
```

prime divisors

In this section, the results of the preceding section are used to derive Euclid's lemma.

Lemma.

```
In[10]:= (Map[not, SubstTest[and, implies[p0, p2], implies[and[p1, p2], or[p3, p4]],
             implies[and[p0, p4], p5], not[implies[and[p0, p1, p2], or[p3, p5]]],
             {p0 -> equal[t, APPLY[GLB[DIV], set[nat[x], nat[y]]]], p1 → member[nat[x], PRIMES],
              p2 -> member[pair[t, nat[x]], DIV], p3 → equal[t, set[0]],
              p4 → equal[t, nat[x]], p5 → member[pair[nat[x], nat[y]], DIV]}]] //
           Reverse) /. t → APPLY[GLB[DIV], set[nat[x], nat[y]]]

Out[10]= or[equal[APPLY[GLB[DIV], set[nat[x], nat[y]]], set[0]],
           member[pair[nat[x], nat[y]], DIV], not[member[nat[x], PRIMES]]] == True

In[11]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

The **nat** wrapper on **x** is not needed.

Theorem.

```
In[12]:= SubstTest[implies, equal[x, nat[t]], or[equal[APPLY[GLB[DIV], set[x, nat[y]]], set[0]],
           member[pair[x, nat[y]], DIV], not[member[x, PRIMES]]], t → x] // Reverse

Out[12]= or[equal[APPLY[GLB[DIV], set[x, nat[y]]], set[0]],
           member[pair[x, nat[y]], DIV], not[member[x, PRIMES]]] == True

In[13]:= or[equal[APPLY[GLB[DIV], set[x_, nat[y_]]], set[0]],
           member[pair[x_, nat[y_]], DIV], not[member[x_, PRIMES]]] := True
```

The other **nat** wrapper could be replaced by a numberhood literal, if desired.

Theorem. Quaife's Theorem **(PR9)**. If **x** is a prime and **y** is a natural number, then either **x** divides **y**, or they are relatively prime.

```
In[14]:= SubstTest[implies, equal[y, nat[t]], or[equal[APPLY[GLB[DIV], set[x, y]], set[0]],
           member[pair[x, y], DIV], not[member[x, PRIMES]]], t → y] // Reverse

Out[14]= or[equal[APPLY[GLB[DIV], set[x, y]], set[0]],
           member[pair[x, y], DIV], not[member[x, PRIMES]], not[member[y, omega]]] == True

In[15]:= or[equal[APPLY[GLB[DIV], set[x_, y_]], set[0]], member[pair[x_, y_], DIV],
           not[member[x_, PRIMES]], not[member[y_, omega]]] := True
```

Lemma.

```
In[16]:= Map[not, SubstTest[and, implies[p1, or[p3, p4]],
           implies[and[p2, p3], p5], not[implies[and[p1, p2], or[p4, p5]]],
           {p1 → member[nat[x], PRIMES], p2 → member[pair[nat[x], natmul[nat[y], nat[z]]], DIV],
            p3 → equal[APPLY[GLB[DIV], set[nat[x], nat[y]]], set[0]],
            p4 -> member[pair[nat[x], nat[y]], DIV],
            p5 -> member[pair[nat[x], nat[z]], DIV]}]] // Reverse

Out[16]= or[member[pair[nat[x], nat[y]], DIV],
           member[pair[nat[x], nat[z]], DIV], not[member[nat[x], PRIMES]],
           not[member[pair[nat[x], natmul[nat[y], nat[z]]], DIV]]] == True

In[17]:= (% /. {x → x_, y → y_, z → z_}) /. Equal → SetDelayed
```

None of the **nat** wrappers are needed.

Corollary. Quaife's theorem **(PR10)**. Euclid's lemma. If a prime divides a product, it divides one of the factors.

```
In[18]:= SubstTest[implies, and[equal[x, nat[u]], equal[y, nat[v]], equal[z, nat[w]]],
           or[member[pair[x, y], DIV], member[pair[x, z], DIV], not[member[x, PRIMES]],
            not[member[pair[x, natmul[y, z]], DIV]]]], {u → x, v → y, w → z}] // MapNotNot // Reverse

Out[18]= or[member[pair[x, y], DIV], member[pair[x, z], DIV],
          not[member[x, PRIMES]], not[member[pair[x, natmul[y, z]], DIV]]] == True

In[19]:= or[member[pair[x_, y_], DIV], member[pair[x_, z_], DIV],
          not[member[x_, PRIMES]], not[member[pair[x_, natmul[y_, z_]], DIV]]] := True
```