monotonicity of greatest common divisors

Johan G. F. Belinfante
2007 April 21

```
In[1]:= SetDirectory["l:"]; << goedel92.20a; << tools.m

        :Package Title: goedel92.20a        2007 April 20 at 5:25 a.m.

        It is now:  2007 Apr 21 at 15:18

        Loading Simplification Rules

        TOOLS.M                        Revised 2007 March 25

        weightlimit = 40
```

summary

A monotonicity property of greatest common divisors is derived.

derivation

Lemma.

```
In[2]:= SubstTest[implies, and[subclass[x, y], subclass[y, z]],
          subclass[x, z], z → image[DIV, set[APPLY[GLB[DIV], y]]]] // Reverse

Out[2]= or[not[subclass[x, y]], not[subclass[y, omega]],
          subclass[x, image[DIV, set[APPLY[GLB[DIV], y]]]]] == True

In[3]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Monotonicity Theorem.   The gcd of a set of numbers divides the gcd of any subset.

```
In[4]:= Map[not, SubstTest[and, implies[p3, or[p4, p5]],
            not[implies[and[p1, p2], p4]], {p1 → subclass[x, y], p2 → subclass[y, omega],
             p3 → subclass[x, image[DIV, set[APPLY[GLB[DIV], y]]]],
             p4 → member[pair[APPLY[GLB[DIV], y], APPLY[GLB[DIV], x]], DIV],
             p5 → equal[0, x]}]] // Reverse

Out[4]= or[member[pair[APPLY[GLB[DIV], y], APPLY[GLB[DIV], x]], DIV],
          not[subclass[x, y]], not[subclass[y, omega]]] == True

In[5]:= or[member[pair[APPLY[GLB[DIV], y_], APPLY[GLB[DIV], x_]], DIV],
          not[subclass[x_, y_]], not[subclass[y_, omega]]] := True
```

removing variables

The following temporary lemma makes it easier to eliminate the variables.

```
In[6]:= (member[pair[u, v], composite[inverse[funpart[t]], s, funpart[t]]] // AssertTest) /.
         {s → DIV, t → GLB[DIV]}

Out[6]= member[pair[u, v], composite[inverse[GLB[DIV]], DIV, GLB[DIV]]] ==
         and[member[pair[APPLY[GLB[DIV], u], APPLY[GLB[DIV], v]], DIV],
          subclass[u, omega], subclass[v, omega]]

In[7]:= member[pair[u_, v_], composite[inverse[GLB[DIV]], DIV, GLB[DIV]]] :=
         and[member[pair[APPLY[GLB[DIV], u], APPLY[GLB[DIV], v]], DIV],
          subclass[u, omega], subclass[v, omega]]
```

The variables can now be eliminated in the standard way:

```
In[8]:= Map[composite[Id, complement[#]] &,
         SubstTest[class, pair[u, v], implies[member[pair[u, v], x], member[pair[u, v], y]],
          {x → composite[inverse[S], id[P[omega]]],
           y -> composite[inverse[GLB[DIV]], DIV, GLB[DIV]]}]]

Out[8]= composite[
          intersection[composite[inverse[GLB[DIV]], complement[DIV], GLB[DIV]], inverse[S]],
          id[P[omega]]] == 0

In[9]:= % /. Equal → SetDelayed
```

This can be restated as an inclusion:

```
In[10]:= SubstTest[empty, dif[u, v], {u → composite[inverse[S], id[P[omega]]],
           v -> composite[inverse[GLB[DIV]], DIV, GLB[DIV]]}]

Out[10]= subclass[composite[inverse[S], id[P[omega]]],
           composite[inverse[GLB[DIV]], DIV, GLB[DIV]]] == True

In[11]:= % /. Equal → SetDelayed
```

This result can also be reformulated as follows.

```
In[12]:= SubstTest[implies, subclass[u, v], subclass[image[t, u], image[t, v]],
           {t → cross[GLB[DIV], GLB[DIV]], u -> composite[inverse[S], id[P[omega]]],
            v -> composite[inverse[GLB[DIV]], DIV, GLB[DIV]]}] // Reverse

Out[12]= subclass[composite[GLB[DIV], inverse[S], inverse[GLB[DIV]]], DIV] == True

In[13]:= % /. Equal → SetDelayed
```

Lemma.

```
In[14]:= Map[subclass[#, GLB[DIV]] &, Assoc[GLB[DIV],
            composite[VERTSECT[DIV], id[omega]], inverse[VERTSECT[DIV]]]] // Reverse

Out[14]= subclass[composite[id[omega], inverse[VERTSECT[DIV]]], GLB[DIV]] == True

In[15]:= % /. Equal → SetDelayed
```

Lemma. (Inclusion in the opposite direction.)

```
In[16]:= (SubstTest[implies, subclass[u, v], subclass[image[u, w], image[v, w]],
             {u → cross[x, x], v → cross[y, y]}] // Reverse) /.
          {w → inverse[S], x -> composite[id[omega], inverse[VERTSECT[DIV]]], y -> GLB[DIV]}

Out[16]= subclass[DIV, composite[GLB[DIV], inverse[S], inverse[GLB[DIV]]]] == True

In[17]:= % /. Equal → SetDelayed
```

Main Theorem. (Variable-free formulation of monotonicity.)

```
In[18]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> composite[GLB[DIV], inverse[S], inverse[GLB[DIV]]], v → DIV}]

Out[18]= equal[DIV, composite[GLB[DIV], inverse[S], inverse[GLB[DIV]]]] == True

In[19]:= composite[GLB[DIV], inverse[S], inverse[GLB[DIV]]] := DIV
```

Corollary.

```
In[20]:= composite[GLB[DIV], S, inverse[GLB[DIV]]] // DoubleInverse

Out[20]= composite[GLB[DIV], S, inverse[GLB[DIV]]] == inverse[DIV]

In[21]:= composite[GLB[DIV], S, inverse[GLB[DIV]]] := inverse[DIV]
```