sums of non-negative rational numbers

Johan G. F. Belinfante
2012 October 26

```
In[1]:= SetDirectory["l:"]; << goedel.12oct24a

        :Package Title: goedel.12oct24a              2012 October 24 at 7:50 p.m.

        Loading takes about sixteen minutes, half that time due to builtin pauses.

        It is now:  2012 Oct 26 at 2:27

        Loading Simplification Rules

        TOOLS.M is now incorporated in the GOEDEL program as of 2010 September 3

        weightlimit = 40

        Loading completed.

        It is now:  2012 Oct 26 at 2:42
```

summary

The sum of non-negative rational numbers is non-negative.

temporary definitions

Some temporary abbreviations are introduced, not only to save some writing, but to increase transparency.

The sum of integer-valued functions is denoted **funadd[x, y]**.

```
In[2]:= funadd[x_, y_] := composite[INTADD,
          intersection[composite[inverse[FIRST], x], composite[inverse[SECOND], y]]]
```

The union of the first and third quadrants of the integer plane $\mathbf{Z} \times \mathbf{Z}$ is denoted **u13**.

```
In[3]:= u13 := union[cart[image[INVERSE, range[PLUS]], image[INVERSE, range[PLUS]]],
          cart[range[PLUS], range[PLUS]]]
```

The following are temporary abbreviations for the sets of non-negative and non-positive rationals.

```
In[4]:= POSRATS := intersection[RATS, P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]

In[5]:= NEGRATS := intersection[RATS,
          P[complement[cart[image[INVERSE, range[PLUS]], complement[range[PLUS]]]]]]
```

funadd

Theorem. An inclusion: **funadd[rat[x], rat[y]] ⊂ U[RATS]**.

```
In[6]:= SubstTest[implies, and[subclass[u, w], subclass[v, w]],
          subclass[funadd[u, v], funadd[w, w]], {u → rat[x], v → rat[y], w → U[RATS]}] // Reverse

Out[6]= subclass[composite[INTADD, intersection[
            composite[inverse[FIRST], rat[x]], composite[inverse[SECOND], rat[y]]]],
          union[cart[intersection[Z, complement[set[id[omega]]]], Z],
           cart[set[id[omega]], set[id[omega]]]]] == True

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

Temporary Lemma.

```
In[8]:= SubstTest[subclass, funadd[rat[x], rat[y]],
          intersection[u, v], {u → u13, v → U[RATS]}] // Reverse

Out[8]= subclass[composite[INTADD, intersection[
            composite[inverse[FIRST], rat[x]], composite[inverse[SECOND], rat[y]]]],
          union[cart[intersection[complement[set[id[omega]]], image[INVERSE, range[PLUS]]],
            image[INVERSE, range[PLUS]]],
           cart[intersection[complement[set[id[omega]]], range[PLUS]], range[PLUS]],
           cart[set[id[omega]], set[id[omega]]]]] ==
         subclass[composite[INTADD, intersection[composite[inverse[FIRST], rat[x]],
            composite[inverse[SECOND], rat[y]]]],
          union[cart[image[INVERSE, range[PLUS]], image[INVERSE, range[PLUS]]],
           cart[range[PLUS], range[PLUS]]]]

In[9]:= subclass[composite[INTADD, intersection[
            composite[inverse[FIRST], rat[x_]], composite[inverse[SECOND], rat[y_]]]],
          union[cart[intersection[complement[set[id[omega]]], image[INVERSE, range[PLUS]]],
            image[INVERSE, range[PLUS]]],
           cart[intersection[complement[set[id[omega]]], range[PLUS]], range[PLUS]],
           cart[set[id[omega]], set[id[omega]]]]] :=
         subclass[composite[INTADD, intersection[composite[inverse[FIRST], rat[x]],
            composite[inverse[SECOND], rat[y]]]],
          union[cart[image[INVERSE, range[PLUS]], image[INVERSE, range[PLUS]]],
           cart[range[PLUS], range[PLUS]]]]
```

Key lemma.

```
In[11]:= SubstTest[implies,
           and[subclass[u, v], invariant[v, range[PLUS]]], invariant[u, range[PLUS]],
           {u → funadd[rat[x], rat[y]], v → intersection[u13, U[RATS]]}] // Reverse

Out[11]= or[not[subclass[composite[INTADD, intersection[
               composite[inverse[FIRST], rat[x]], composite[inverse[SECOND], rat[y]]]],
             union[cart[image[INVERSE, range[PLUS]], image[INVERSE, range[PLUS]]],
              cart[range[PLUS], range[PLUS]]]]],
           subclass[image[INTADD, composite[rat[y], id[range[PLUS]], inverse[rat[x]]]],
            range[PLUS]]] == True

In[12]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma. An inclusion for non-negative rationals:

```
In[13]:= SubstTest[implies, and[subclass[u, w], subclass[v, w]],
           subclass[funadd[u, v], funadd[w, w]], {u → rat[x], v → rat[y], w → u13}] // Reverse

Out[13]= or[not[subclass[image[rat[x], range[PLUS]], range[PLUS]]],
           not[subclass[image[rat[y], range[PLUS]], range[PLUS]]],
           subclass[composite[INTADD, intersection[
              composite[inverse[FIRST], rat[x]], composite[inverse[SECOND], rat[y]]]],
            union[cart[image[INVERSE, range[PLUS]], image[INVERSE, range[PLUS]]],
             cart[range[PLUS], range[PLUS]], cart[set[id[omega]], Z]]]] == True

In[14]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem. If **x** and **y** are non-negative, then **funadd[x, y]** leaves **range[PLUS]** invariant.

```
In[18]:= Map[not, SubstTest[and, implies[p1, p2],
            implies[p2, p3], implies[p3, p4], not[implies[p1, p4]],
           {p1 → and[invariant[rat[x], range[PLUS]], invariant[rat[y], range[PLUS]]],
            p2 → subclass[composite[INTADD, intersection[
                composite[inverse[FIRST], rat[x]], composite[inverse[SECOND], rat[y]]]],
              union[cart[image[INVERSE, range[PLUS]], image[INVERSE, range[PLUS]]],
               cart[range[PLUS], range[PLUS]], cart[set[id[omega]], Z]]],
            p3 -> subclass[composite[INTADD, intersection[composite[inverse[FIRST], rat[x]],
                composite[inverse[SECOND], rat[y]]]], union[cart[image[INVERSE, range[PLUS]],
                image[INVERSE, range[PLUS]]], cart[range[PLUS], range[PLUS]]]],
            p4 → invariant[funadd[rat[x], rat[y]], range[PLUS]]}]] // Reverse

Out[18]= or[not[subclass[image[rat[x], range[PLUS]], range[PLUS]]],
           not[subclass[image[rat[y], range[PLUS]], range[PLUS]]], subclass[image[INTADD,
             composite[rat[y], id[range[PLUS]], inverse[rat[x]]]], range[PLUS]]] == True

In[19]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

---

simplification rules for ratadd

Some simplification rules are derived for the sum of two rationals.

Theorem. The sum commutes with **INVERSE**.

```
In[20]:= SubstTest[composite, INVERSE, rat[t], t → ratadd[rat[x], rat[y]]] // Reverse

Out[20]= composite[INVERSE, ratadd[rat[x], rat[y]]] == composite[ratadd[rat[x], rat[y]], INVERSE]

In[21]:= composite[INVERSE, ratadd[rat[x_], rat[y_]]] :=
          composite[ratadd[rat[x], rat[y]], INVERSE]
```

Corollary.

```
In[22]:= ImageComp[INVERSE, ratadd[rat[x], rat[y]], z]

Out[22]= image[ratadd[rat[x], rat[y]], image[INVERSE, z]] ==
          image[INVERSE, image[ratadd[rat[x], rat[y]], z]]

In[23]:= image[ratadd[rat[x_], rat[y_]], image[INVERSE, z_]] :=
          image[INVERSE, image[ratadd[rat[x], rat[y]], z]]
```

Theorem.

```
In[24]:= SubstTest[subclass, image[INVERSE, image[rat[t], x]],
           y, t → ratadd[rat[u], rat[v]]] // Reverse

Out[24]= subclass[image[INVERSE, image[ratadd[rat[u], rat[v]], x]], y] ==
          subclass[image[ratadd[rat[u], rat[v]], x], image[INVERSE, y]]

In[25]:= subclass[image[INVERSE, image[ratadd[rat[u_], rat[v_]], x_]], y_] :=
          subclass[image[ratadd[rat[u], rat[v]], x], image[INVERSE, y]]
```

main theorem

Lemma. The sum of rationals is either non-negative or non-positive.

```
In[26]:= SubstTest[or, invariant[rat[t], range[PLUS]],
           invariant[composite[rat[t], INVERSE], range[PLUS]],
           t → ratadd[rat[x], rat[y]]] // Reverse

Out[26]= or[subclass[image[ratadd[rat[x], rat[y]], range[PLUS]], image[INVERSE, range[PLUS]]],
           subclass[image[ratadd[rat[x], rat[y]], range[PLUS]], range[PLUS]]] == True

In[27]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma. If the rational sum of two rationals conducts **u** to **v**, then so does their functional sum.

```
In[33]:= SubstTest[implies,
           and[subclass[s, t], subclass[image[t, u], v]], subclass[image[s, u], v],
           {s -> funadd[rat[x], rat[y]], t → ratadd[rat[x], rat[y]]}] // Reverse

Out[33]= or[not[subclass[image[ratadd[rat[x], rat[y]], u], v]],
           subclass[image[INTADD, composite[rat[y], id[u], inverse[rat[x]]]], v]] == True
```

```
In[34]:= (% /. {u → u_, v → v_, x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[36]:= SubstTest[subclass, image[rat[t], range[PLUS]],
           set[id[omega]], t → ratadd[rat[x], rat[y]]] // Reverse

Out[36]= subclass[image[ratadd[rat[x], rat[y]], range[PLUS]], set[id[omega]]] ==
          equal[cart[Z, set[id[omega]]], ratadd[rat[x], rat[y]]]

In[37]:= subclass[image[ratadd[rat[x_], rat[y_]], range[PLUS]], set[id[omega]]] :=
          equal[cart[Z, set[id[omega]]], ratadd[rat[x], rat[y]]]
```

Lemma.

```
In[40]:= SubstTest[implies, and[not[subclass[domain[t], set[id[omega]]]], subclass[t, rat[w]]],
           equal[hull[RATS, t], rat[w]],
           {t → funadd[rat[x], rat[y]], w → cart[Z, set[id[omega]]]}] // Reverse

Out[40]= or[equal[cart[Z, set[id[omega]]], ratadd[rat[x], rat[y]]], not[
            subclass[image[INTADD, composite[rat[y], inverse[rat[x]]]], set[id[omega]]]]] == True

In[41]:= or[equal[cart[Z, set[id[omega]]], ratadd[rat[x_], rat[y_]]], not[subclass[
              image[INTADD, composite[rat[y_], inverse[rat[x_]]]], set[id[omega]]]]] := True
```

Lemma.

```
In[45]:= SubstTest[implies, conduct[ratadd[rat[x], rat[y]], u, v],
           conduct[funadd[rat[x], rat[y]], u, v],
           {u → range[PLUS], v → image[INVERSE, range[PLUS]]}] // Reverse

Out[45]= or[not[subclass[image[ratadd[rat[x], rat[y]], range[PLUS]],
              image[INVERSE, range[PLUS]]]], subclass[
            image[INVERSE, image[INTADD, composite[rat[y], id[range[PLUS]], inverse[rat[x]]]]],
            range[PLUS]]] == True

In[46]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[50]:= Map[implies[#,
            subclass[image[INTADD, composite[rat[y], id[range[PLUS]], inverse[rat[x]]]],
              set[id[omega]]]] &, SubstTest[subclass, t,
           intersection[u, v], {t -> image[funadd[rat[x], rat[y]], range[PLUS]],
            u → range[PLUS], v → image[INVERSE, range[PLUS]]}]]

Out[50]= or[not[subclass[image[INTADD, composite[rat[y], id[range[PLUS]], inverse[rat[x]]]],
              range[PLUS]]], not[subclass[image[INVERSE, image[INTADD,
                composite[rat[y], id[range[PLUS]], inverse[rat[x]]]]], range[PLUS]]],
           subclass[image[INTADD, composite[rat[y], id[range[PLUS]], inverse[rat[x]]]],
            set[id[omega]]]] == True

In[51]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

new

Lemma.

```
In[67]:= ImageComp[INVERSE, funadd[rat[x], rat[y]], z]

Out[67]= image[INTADD, composite[rat[y], id[image[INVERSE, z]], inverse[rat[x]]]] ==
          image[INVERSE, image[INTADD, composite[rat[y], id[z], inverse[rat[x]]]]]

In[68]:= image[INTADD, composite[rat[y_], id[image[INVERSE, z_]], inverse[rat[x_]]]] :=
          image[INVERSE, image[INTADD, composite[rat[y], id[z], inverse[rat[x]]]]]
```

Lemma.

```
In[74]:= SubstTest[subclass, image[INVERSE, x], image[INVERSE, y], y → set[id[omega]]] // Reverse

Out[74]= subclass[image[INVERSE, x], set[id[omega]]] ==
          subclass[intersection[x, P[cart[V, V]]], set[id[omega]]]

In[75]:= subclass[image[INVERSE, x_], set[id[omega]]] :=
          subclass[intersection[x, P[cart[V, V]]], set[id[omega]]]
```

Lemma.

```
In[76]:= Map[implies[subclass[#, set[id[omega]]],
             subclass[range[funadd[rat[x], rat[y]]], set[id[omega]]]] &,
          SubstTest[image, t, union[u, v], t → funadd[rat[x], rat[y]]] /.
           {u → range[PLUS], v → image[INVERSE, range[PLUS]]}]

Out[76]= or[not[subclass[image[INTADD,
              composite[rat[y], id[range[PLUS]], inverse[rat[x]]]], set[id[omega]]]],
           subclass[image[INTADD, composite[rat[y], inverse[rat[x]]]], set[id[omega]]]] == True

In[77]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Main Theorem. The sum of non-negative rationals is non-negative.

```
In[82]:= Map[not, SubstTest[and, implies[p1, p4], implies[p2, p3],
            implies[p3, p5], (*implies[and[p4,p5],p6],*) implies[p6, p7],
            implies[p7, p8], implies[p8, not[p2]], not[implies[p1, not[p2]]],
            {p1 → and[subclass[image[rat[x], range[PLUS]], range[PLUS]],
                subclass[image[rat[y], range[PLUS]], range[PLUS]]],
             p2 → not[subclass[image[ratadd[rat[x], rat[y]], range[PLUS]], range[PLUS]]], p3 →
              subclass[image[ratadd[rat[x], rat[y]], range[PLUS]], image[INVERSE, range[PLUS]]],
             p4 → subclass[image[funadd[rat[x], rat[y]], range[PLUS]], range[PLUS]], p5 →
              subclass[image[funadd[rat[x], rat[y]], range[PLUS]], image[INVERSE, range[PLUS]]],
             p6 → subclass[image[funadd[rat[x], rat[y]], range[PLUS]], set[id[omega]]],
             p7 -> subclass[image[INTADD, composite[rat[y], inverse[rat[x]]]], set[id[omega]]],
             p8 → equal[ratadd[rat[x], rat[y]], cart[Z, set[id[omega]]]]}]] // Reverse

Out[82]= or[not[subclass[image[rat[x], range[PLUS]], range[PLUS]]],
           not[subclass[image[rat[y], range[PLUS]], range[PLUS]]],
           subclass[image[ratadd[rat[x], rat[y]], range[PLUS]], range[PLUS]]] == True

In[84]:= or[not[subclass[image[rat[x_], range[PLUS]], range[PLUS]]],
           not[subclass[image[rat[y_], range[PLUS]], range[PLUS]]],
           subclass[image[ratadd[rat[x_], rat[y_]], range[PLUS]], range[PLUS]]] := True
```

Corollary. (Eliminate the **rat** wrappers.)

```
In[85]:= SubstTest[implies, and[equal[x, rat[u]], equal[y, rat[v]]],
           or[not[subclass[image[x, range[PLUS]], range[PLUS]]],
            not[subclass[image[y, range[PLUS]], range[PLUS]]],
            subclass[image[ratadd[x, y], range[PLUS]], range[PLUS]]], {u → x, v → y}] // Reverse

Out[85]= or[not[member[x, RATS]], not[member[y, RATS]],
           not[subclass[image[x, range[PLUS]], range[PLUS]]],
           not[subclass[image[y, range[PLUS]], range[PLUS]]],
           subclass[image[ratadd[x, y], range[PLUS]], range[PLUS]]] == True

In[87]:= or[not[member[x_, RATS]], not[member[y_, RATS]],
           not[subclass[image[x_, range[PLUS]], range[PLUS]]],
           not[subclass[image[y_, range[PLUS]], range[PLUS]]],
           subclass[image[ratadd[x_, y_], range[PLUS]], range[PLUS]]] := True
```

simplification rules

Lemma.

```
In[96]:= SubstTest[member, APPLY[funpart[t], PAIR[x, y]], V, t → RATADD] // Reverse

Out[96]= member[ratadd[x, y], V] == and[member[x, RATS], member[y, RATS]]

In[97]:= member[ratadd[x_, y_], V] := and[member[x, RATS], member[y, RATS]]
```

Theorem.

In[101]:=
```
ImageComp[RATADD, id[cart[RATS, V]], cart[x, y]] // Reverse
```

Out[101]=
```
image[RATADD, cart[intersection[RATS, x], y]] == image[RATADD, cart[x, y]]
```

In[102]:=
```
image[RATADD, cart[intersection[RATS, x_], y_]] := image[RATADD, cart[x, y]]
```

Theorem,

In[103]:=
```
ImageComp[RATADD, id[cart[V, RATS]], cart[x, y]] // Reverse
```

Out[103]=
```
image[RATADD, cart[x, intersection[RATS, y]]] == image[RATADD, cart[x, y]]
```

In[104]:=
```
image[RATADD, cart[x_, intersection[RATS, y_]]] := image[RATADD, cart[x, y]]
```

In[105]:=
```
Map[empty[composite[Id, complement[#]]] &,
 SubstTest[class, pair[x, y], implies[member[pair[x, y], u], member[pair[x, y], v]],
  {u -> cart[POSRATS, POSRATS], v -> image[inverse[RATADD], POSRATS]}]]
```

Out[105]=
```
subclass[
  image[U[image[RATADD, cart[P[complement[cart[range[PLUS], complement[range[PLUS]]]]],
      P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]]],
   range[PLUS]], range[PLUS]] == True
```

In[106]:=
```
% /. Equal → SetDelayed
```

Lemma.

In[107]:=
```
SubstTest[implies, subclass[u, v], subclass[image[RATADD, u], image[RATADD, v]],
  {u → cart[set[cart[Z, set[id[omega]]]], POSRATS], v → cart[POSRATS, POSRATS]}] // Revers
```

Out[107]=
```
subclass[intersection[RATS, P[complement[cart[range[PLUS], complement[range[PLUS]]]]]],
  image[RATADD, cart[P[complement[cart[range[PLUS], complement[range[PLUS]]]]],
    P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]]] == True
```

In[108]:=
```
% /. Equal → SetDelayed
```

Theorem. The sum of non-negative rationals is non-negative.

In[109]:=

```
SubstTest[and, subclass[u, v], subclass[v, u],
  {u → image[RATADD, cartsq[POSRATS]], v → POSRATS}]
```

Out[109]=

```
equal[image[RATADD, cart[P[complement[cart[range[PLUS], complement[range[PLUS]]]]],
    P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]],
  intersection[RATS, P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]] == True
```

In[111]:=

```
image[RATADD, cart[P[complement[cart[range[PLUS], complement[range[PLUS]]]]],
   P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]] :=
 intersection[RATS, P[complement[cart[range[PLUS], complement[range[PLUS]]]]]]
```