monotonicity of ordlist[x]

Johan G. F. Belinfante
2006 April 30

```
In[1]:= SetDirectory["l:"]; << goedel80.29a; << tools.m

        :Package Title: goedel80.29a          2006 April 29 at 3:00 p.m.

        It is now:  2006 Apr 30 at 11:21

        Loading Simplification Rules

        TOOLS.M                       Revised 2006 March 7

        weightlimit = 40
```

summary

The function **ordlist[x]** is strictly monotone.

APPLY for ordlist

Vertical sections of functions are singletons.

```
In[2]:= SubstTest[image, funpart[z], set[y], z → ordlist[x]]

Out[2]= image[ordlist[x], set[y]] == set[APPLY[ordlist[x], y]]

In[3]:= image[ordlist[x_], set[y_]] := set[APPLY[ordlist[x], y]]
```

monotone property

Lemma.

```
In[4]:= SubstTest[implies, and[subclass[t, u], subclass[v, w]],
          subclass[composite[t, v], composite[u, w]],
          {t → HULL[x], u → S, v → SUCC, w → S}]

Out[4]= subclass[composite[HULL[x], SUCC], S] == True

In[5]:= subclass[composite[HULL[x_], SUCC], S] := True
```

Lemma.

```
In[6]:= SubstTest[implies, and[subclass[t, u], subclass[v, w]],
         subclass[composite[t, v], composite[u, w]],
         {t → HULL[x], u → S, v → SUCC, w → E}]

Out[6]= subclass[composite[HULL[x], SUCC], E] == True

In[7]:= subclass[composite[HULL[x_], SUCC], E] := True
```

Temporary lemma.

```
In[9]:= equal[intersection[S, composite[HULL[intersection[OMEGA, x]], SUCC]],
         composite[HULL[intersection[OMEGA, x]], SUCC]]

Out[9]= True

In[10]:= intersection[S, composite[HULL[intersection[OMEGA, x_]], SUCC]] :=
          composite[HULL[intersection[OMEGA, x]], SUCC]
```

Theorem. The function **ordlist[x]** is monotone.

```
In[11]:= SubstTest[monotone, iterate[funpart[intersection[S, u]], set[v]], S, S,
          {u → composite[HULL[intersection[OMEGA, x]], SUCC], v → A[intersection[OMEGA, x]]}]

Out[11]= subclass[composite[ordlist[x], S, inverse[ordlist[x]]], S] == True

In[12]:= subclass[composite[ordlist[x_], S, inverse[ordlist[x_]]], S] := True
```

---

strict monotonicity

Lemma.

```
In[13]:= Assoc[ordlist[x], SUCC, id[OMEGA]]

Out[13]= composite[ordlist[x], id[OMEGA], SUCC] ==
          composite[HULL[intersection[OMEGA, x]], SUCC, ordlist[x], id[OMEGA]]

In[14]:= composite[ordlist[x_], id[OMEGA], SUCC] :=
          composite[HULL[intersection[OMEGA, x]], SUCC, ordlist[x], id[OMEGA]]
```

Lemma.

```
In[15]:= SubstTest[implies, subclass[u, v], subclass[composite[u, w], composite[v, w]],
          {u -> composite[HULL[intersection[OMEGA, x]], SUCC], v → E, w → ordlist[x]}]

Out[15]= subclass[composite[HULL[intersection[OMEGA, x]], SUCC, ordlist[x]],
           composite[inverse[IMAGE[inverse[ordlist[x]]], E]] == True

In[16]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem.

```
In[17]:= SubstTest[implies, subclass[composite[u, v], composite[w, u]],
          subclass[composite[u, trv[v]], composite[trv[w], u]],
          {u → ordlist[x], v → composite[id[omega], SUCC], w → composite[id[OMEGA], E]}]

Out[17]= subclass[composite[ordlist[x], E],
           composite[inverse[IMAGE[inverse[ordlist[x]]]], E]] == True

In[18]:= subclass[composite[ordlist[x_], E],
           composite[inverse[IMAGE[inverse[ordlist[x_]]]], E]] := True
```

Lemma.

```
In[19]:= composite[inverse[IMAGE[inverse[ordlist[x]]]],
           inverse[IMAGE[ordlist[x]]], E] // DoubleInverse

Out[19]= composite[inverse[IMAGE[inverse[ordlist[x]]]], inverse[IMAGE[ordlist[x]]], E] ==
          composite[inverse[IMAGE[id[range[ordlist[x]]]]], E]

In[20]:= composite[inverse[IMAGE[inverse[ordlist[x_]]]], inverse[IMAGE[ordlist[x_]]], E] :=
          composite[inverse[IMAGE[id[range[ordlist[x]]]]], E]
```

Lemma.

```
In[21]:= SubstTest[implies, subclass[u, v], subclass[composite[u, w], composite[v, w]],
          {u -> composite[ordlist[x], E],
           v -> composite[inverse[IMAGE[inverse[ordlist[x]]]], E], w → inverse[ordlist[x]]}]

Out[21]= subclass[composite[ordlist[x], inverse[IMAGE[ordlist[x]]], E],
           composite[inverse[IMAGE[id[range[ordlist[x]]]]], E]] == True

In[22]:= (% /. x → x_) /. Equal → SetDelayed
```

Corollary.

```
In[23]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]], subclass[u, w],
          {u -> composite[ordlist[x], inverse[IMAGE[ordlist[x]]], E],
           v -> composite[inverse[IMAGE[id[range[ordlist[x]]]]], E], w → E}]

Out[23]= subclass[composite[ordlist[x], inverse[IMAGE[ordlist[x]]]], S] == True

In[24]:= subclass[composite[ordlist[x_], inverse[IMAGE[ordlist[x_]]]], S] := True
```

Restatement:

```
In[25]:= monotone[ordlist[x], E, E]

Out[25]= True
```

corollaries

Temporary lemma.

```
In[26]:= (subclass[y, composite[S, IMAGE[y]]] // AssertTest) /. y → funpart[x]

Out[26]= subclass[funpart[x], composite[S, IMAGE[funpart[x]]]] ==
          subclass[composite[funpart[x], inverse[IMAGE[funpart[x]]]], S]

In[27]:= subclass[funpart[x_], composite[S, IMAGE[funpart[x_]]]] :=
          subclass[composite[funpart[x], inverse[IMAGE[funpart[x]]]], S]
```

Corollary.

```
In[28]:= SubstTest[subclass, funpart[w], composite[S, IMAGE[funpart[w]]], w -> ordlist[x]]

Out[28]= subclass[ordlist[x], composite[S, IMAGE[ordlist[x]]]] == True

In[29]:= subclass[ordlist[x_], composite[S, IMAGE[ordlist[x_]]]] := True
```

Corollary.

```
In[30]:= SubstTest[implies, subclass[u, v], subclass[image[u, w], image[v, w]],
          {u → ordlist[x], v → composite[S, IMAGE[ordlist[x]]], w → set[y]}]

Out[30]= subclass[image[ordlist[x], y], APPLY[ordlist[x], y]] == True

In[31]:= subclass[image[ordlist[x_], y_], APPLY[ordlist[x_], y_]] := True
```