next limit ordinal

Johan G. F. Belinfante
2007 May 30

```
In[1]:= SetDirectory["l:"]; << goedel93.29a; << tools.m

        :Package Title: goedel93.29a      2007 May 29 at 11:20 p.m.

        It is now:  2007 May 30 at 20:44

        Loading Simplification Rules

        TOOLS.M                                     Revised 2007 May 6

        weightlimit = 40
```

summary

The **successor orbit** of a set **x** is the set whose members are **x, succ[x]**, **succ[succ[x]]**, ... ad infinitum. The successor orbit can equivalently be described as the successor-invariant hull of the singleton **set[x]**.

```
In[2]:= range[iterate[SUCC, set[x]]]

Out[2]= hull[invar[SUCC], set[x]]
```

The least limit ordinal greater than a given ordinal **ord[x]** is **hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]**. In this notebook it is shown that this next limit ordinal is the union of **ord[x]** and its successor-orbit. The next limit ordinal can also be written as the sum class of the successor orbit. The successor orbit of **ord[x]** is the range of the function **ordlist[complement[ord[x]]]** which iteratively lists in strictly increasing order the ordinals not less than **ord[x]**, starting with **ord[x]** itself.

```
In[3]:= ordlist[complement[ord[x]]]

Out[3]= iterate[SUCC, set[ord[x]]]
```

One of the main ideas is to derive formulas for the next limit ordinal by applying the general theory of **ordlist** functions.

application of ordlist theory

Theorem. Since the complement of an ordinal is an infinite class, the range of the corresponding **ordlist** function is a set of ordinals with no greatest member.

```
In[4]:= SubstTest[subclass, range[ordlist[t]],
          U[range[ordlist[t]]], t → complement[ord[x]]] // Reverse

Out[4]= subclass[hull[invar[SUCC], set[ord[x]]], U[hull[invar[SUCC], set[ord[x]]]]] == True

In[5]:= (% /. x → x_) /. Equal → SetDelayed
```

Corollary. The ordinal **ord[x]** belongs to this sum class.

```
In[6]:= SubstTest[implies, and[member[u, v], subclass[v, w]], member[u, w],
          {u → ord[x], v -> hull[invar[SUCC], set[ord[x]]],
           w -> U[hull[invar[SUCC], set[ord[x]]]]}] // Reverse

Out[6]= member[ord[x], U[hull[invar[SUCC], set[ord[x]]]]] == True

In[7]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem. The sum class is an ordinal.

```
In[8]:= SubstTest[implies, member[t, P[OMEGA]],
          member[U[t], OMEGA], t -> hull[invar[SUCC], set[ord[x]]]] // Reverse

Out[8]= member[U[hull[invar[SUCC], set[ord[x]]]], OMEGA] == True

In[9]:= (% /. x → x_) /. Equal → SetDelayed
```

It is in fact a limit ordinal, a fact that can be made into a temporary rewrite rule.

```
In[10]:= SubstTest[implies, and[subclass[t, OMEGA], subclass[t, U[t]]],
          equal[U[U[t]], U[t]], t → hull[invar[SUCC], set[ord[x]]]]

Out[10]= True == equal[U[hull[invar[SUCC], set[ord[x]]]], U[U[hull[invar[SUCC], set[ord[x]]]]]]

In[11]:= U[U[hull[invar[SUCC], set[ord[x_]]]]] := U[hull[invar[SUCC], set[ord[x]]]]
```

Theorem. The least limit ordinal greater than **ord[x]** is no bigger than **U[hull[invar[SUCC], set[ord[x]]]]**.

```
In[12]:= SubstTest[implies, member[u, v], subclass[A[v], u],
           {u -> U[hull[invar[SUCC], set[ord[x]]]],
            v -> intersection[OMEGA, fix[BIGCUP], image[S, set[set[ord[x]]]]]}] // Reverse

Out[12]= subclass[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           U[hull[invar[SUCC], set[ord[x]]]]] == True

In[13]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma. (Any ordinal is the set of all lesser ordinals.)

```
In[14]:= SubstTest[implies, member[u, OMEGA], subclass[u, OMEGA],
           u -> U[hull[invar[SUCC], set[ord[x]]]]] // Reverse

Out[14]= subclass[U[hull[invar[SUCC], set[ord[x]]]], OMEGA] == True

In[15]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma. (Temporary rewrite rule.)

```
In[16]:= equal[intersection[OMEGA, U[hull[invar[SUCC], set[ord[x]]]]],
          U[hull[invar[SUCC], set[ord[x]]]]]

Out[16]= True

In[17]:= intersection[OMEGA, U[hull[invar[SUCC], set[ord[x_]]]]] :=
          U[hull[invar[SUCC], set[ord[x]]]]
```

Theorem. From the general theory of **ordlist** functions one obtains this upper bound on the sum class:

```
In[18]:= SubstTest[subclass, intersection[t, OMEGA, U[range[ordlist[t]]]],
           range[ordlist[t]], t → complement[ord[x]]] // Reverse

Out[18]= subclass[U[hull[invar[SUCC], set[ord[x]]]],
           union[hull[invar[SUCC], set[ord[x]]], ord[x]]] == True

In[19]:= (% /. x → x_) /. Equal → SetDelayed
```

main theorem

Lemma. The next limit ordinal holds **ord[x]**. So the reverse is not true:

```
In[20]:= Map[not, SubstTest[implies, member[ord[x], ord[y]], not[member[ord[y], ord[x]]],
            y -> hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]]] // Reverse

Out[20]= member[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]], ord[x]] == False

In[21]:= member[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]], ord[x_]] := False
```

The range of **iterate[x,y]** is contained in **union[range[x],y]**. This implies that, apart possibly from **ord[x]**, the successor orbit of **ord[x]** consists entirely of successor ordinals.

```
In[22]:= SubstTest[subclass, range[iterate[u, v]], union[range[u], v],
           {u → composite[id[OMEGA], SUCC], v → set[ord[x]]}] // Reverse

Out[22]= subclass[hull[invar[SUCC], set[ord[x]]],
           union[intersection[OMEGA, complement[fix[BIGCUP]]], set[ord[x]]]] == True

In[23]:= subclass[hull[invar[SUCC], set[ord[x_]]],
           union[intersection[OMEGA, complement[fix[BIGCUP]]], set[ord[x_]]]] := True
```

Lemma. The next limit ordinal is not the ordinal **ord[x]** itself.

```
In[24]:= Map[not, SubstTest[implies, and[equal[u, v], member[u, v]], member[u, u],
            {u → ord[x], v -> hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]}] // Reverse]

Out[24]= equal[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]], ord[x]] == False

In[25]:= equal[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]], ord[x_]] := False
```

Corollary. The next limit ordinal is not in the successor orbit of **ord[x]**.

```
In[26]:= Map[not, SubstTest[implies, and[member[u, v], subclass[v, w]], member[u, w],
           {u -> hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
            v -> hull[invar[SUCC], set[ord[x]]],
            w -> union[intersection[OMEGA, complement[fix[BIGCUP]]], set[ord[x]]]}]] // Reverse

Out[26]= member[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           hull[invar[SUCC], set[ord[x]]]] == False

In[27]:= member[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]],
           hull[invar[SUCC], set[ord[x_]]]] := False
```

Lemma. The next limit ordinal is not less than the sum class of the successor orbit of **ord[x]**.

```
In[28]:= Map[not, SubstTest[implies, and[member[u, v], subclass[v, w]], member[u, w],
           {u -> hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
            v -> U[hull[invar[SUCC], set[ord[x]]]],
            w -> union[hull[invar[SUCC], set[ord[x]]], ord[x]]}]] // Reverse

Out[28]= member[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           U[hull[invar[SUCC], set[ord[x]]]]] == False

In[29]:= (% /. x → x_) /. Equal → SetDelayed
```

Main Theorem. By trichotomy, the next limit ordinal is the sum class of the successor orbit of **ord[x]**.

```
In[30]:= SubstTest[implies, and[member[u, OMEGA], member[v, OMEGA], subclass[u, v]],
          or[member[u, v], equal[u, v]],
          {u -> hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           v -> U[hull[invar[SUCC], set[ord[x]]]]}]

Out[30]= True == equal[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           U[hull[invar[SUCC], set[ord[x]]]]]

In[31]:= U[hull[invar[SUCC], set[ord[x_]]]] :=
          hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]
```

Comment. Because of this new rewrite rule for the sum class of the successor orbit of **ord[x]**, a few of the theorems about this class derived above are no longer be recognized, and will have to be rederived.

corollaries

Theorem. Repeating an argument used previously now yields the statement that the successor orbit of **ord[x]** is a subclass of the next limit ordinal.

```
In[32]:= SubstTest[subclass, range[ordlist[t]],
           U[range[ordlist[t]]], t → complement[ord[x]]] // Reverse

Out[32]= subclass[hull[invar[SUCC], set[ord[x]]],
           hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]] == True

In[33]:= subclass[hull[invar[SUCC], set[ord[x_]]],
           hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]]] := True
```

Lemma. The ordinal **ord[x]** is a subclass of the next limit ordinal.

```
In[34]:= SubstTest[implies, subclass[u, v], subclass[U[u], U[v]],
           {u → set[ord[x]], v -> hull[invar[SUCC], set[ord[x]]]}] // Reverse

Out[34]= subclass[ord[x], hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]] == True

In[35]:= subclass[ord[x_], hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]]] := True
```

Theorem. (Any ordinal is the set of lesser ordinals.)

```
In[36]:= SubstTest[implies, member[u, OMEGA], subclass[u, OMEGA],
           u -> U[hull[invar[SUCC], set[ord[x]]]]] // Reverse

Out[36]= subclass[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]], OMEGA] == True

In[37]:= subclass[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]], OMEGA] := True
```

Lemma. (Simplification rule.)

```
In[38]:= equal[intersection[OMEGA, hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]],
          hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]]

Out[38]= True

In[39]:= intersection[OMEGA, hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x_]]]] :=
          hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]
```

Lemma.

```
In[40]:= SubstTest[subclass, intersection[t, OMEGA, U[range[ordlist[t]]]],
           range[ordlist[t]], t → complement[ord[x]]] // Reverse

Out[40]= subclass[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           union[hull[invar[SUCC], set[ord[x]]], ord[x]]] == True

In[41]:= (% /. x → x_) /. Equal → SetDelayed
```

Theorem. (The preceding inclusion can be replaced with an equation, which says that the next limit ordinal for a given ordinal is the union of that ordinal and its successor orbit.)

```
In[42]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           v -> union[hull[invar[SUCC], set[ord[x]]], ord[x]]}]

Out[42]= equal[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           union[hull[invar[SUCC], set[ord[x]]], ord[x]]] == True

In[43]:= union[hull[invar[SUCC], set[ord[x_]]], ord[x_]] :=
          hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]
```

Theorem. The successor orbit of **ord[x]** and **ord[x]** are disjoint.

```
In[44]:= SubstTest[subclass, range[ordlist[t]], t, t -> complement[ord[x]]] // Reverse

Out[44]= equal[0, intersection[hull[invar[SUCC], set[ord[x]]], ord[x]]] == True

In[45]:= intersection[hull[invar[SUCC], set[ord[x_]]], ord[x_]] := 0
```

---

the next limit ordinal for succ[ord[x]]

Lemma.

```
In[46]:= SubstTest[U, hull[invar[SUCC], set[ord[t]]], t -> succ[ord[x]]] // Reverse

Out[46]= U[hull[invar[SUCC], set[succ[ord[x]]]]] ==
          hull[intersection[OMEGA, fix[BIGCUP]], set[succ[ord[x]]]]

In[47]:= U[hull[invar[SUCC], set[succ[ord[x_]]]]] :=
          hull[intersection[OMEGA, fix[BIGCUP]], set[succ[ord[x]]]]
```

Theorem. The next limit ordinals for **ord[x]** and **succ[ord[x]]** are the same.

```
In[48]:= Map[equal[U[hull[invar[SUCC], set[ord[x]]]], U[#]] &,
          SubstTest[range, iterate[SUCC, set[ord[t]]], t → succ[ord[x]]]]

Out[48]= equal[hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]],
           hull[intersection[OMEGA, fix[BIGCUP]], set[succ[ord[x]]]]] == True

In[49]:= hull[intersection[OMEGA, fix[BIGCUP]], set[succ[ord[x_]]]] :=
          hull[intersection[OMEGA, fix[BIGCUP]], set[ord[x]]]
```

---

an example

The limit ordinal **omega** is the next limit ordinal for **0**.

```
In[51]:= SubstTest[union, hull[invar[SUCC], set[ord[x]]], ord[x], x → 0]

Out[51]= hull[intersection[OMEGA, fix[BIGCUP]], set[0]] == omega
```

Lemma.

```
In[53]:= equal[union[omega, nat[x]], omega]

Out[53]= True

In[55]:= union[omega, nat[x_]] := omega
```

Theorem. For any natural number, the next limit ordinal is **omega**.

```
In[56]:= SubstTest[union, hull[invar[SUCC], set[ord[t]]], ord[t], t → nat[x]]

Out[56]= hull[intersection[OMEGA, fix[BIGCUP]], set[nat[x]]] == omega

In[57]:= hull[intersection[OMEGA, fix[BIGCUP]], set[nat[x_]]] := omega
```

A variable-free version of this statement can be derived using **reify**.

```
In[61]:= Map[composite[VERTSECT[#], id[omega]] &,
          SubstTest[reify, x, hull[intersection[OMEGA, fix[BIGCUP]], f[x]], f[x] → set[nat[x]]]]

Out[61]= composite[HULL[intersection[OMEGA, fix[BIGCUP]]], SINGLETON, id[omega]] ==
          cart[omega, set[omega]]

In[62]:= composite[HULL[intersection[OMEGA, fix[BIGCUP]]], SINGLETON, id[omega]] :=
          cart[omega, set[omega]]
```