alephs are limit ordinals

Johan G. F. Belinfante
2010 November 20

It is shown in this notebook that the cardinality of any infinite ordinal and its successor are equal. From this it follows that every infinite cardinal is a limit ordinal. An elegant variable-free formulation for this is derived in terms of the $\aleph$ function.

cardinality of successor ordinals

In this section it will be shown that any infinite ordinal has the same cardinality as its successor. The idea is to set up an explicit one-to-one correspondence between $\text{ord}[x]$ and $\text{succ}[\text{ord}[x]]$. To save writing, a temporary abbreviation for it will be introduced.

$$f[x_\_]: = \text{union}[\text{cart}[\text{set}[x], \text{set}[0]], \\
\text{composite}[\text{id}[\text{omega}], \text{SUCC}, \text{id}[\text{intersection}[x, \text{complement}[\text{omega}]]]]]$$

Lemma. A simplification rule.

$$\text{Assoc}[\text{SUCC}, \text{id}[\text{omega}], \text{id}[\text{dif}[x, \text{omega}]]]$$

Theorem. If $\text{ord}[x] \geq \omega$ then $f[x]$ is a function.

$$\text{SubstTest}[\text{subclass}, \text{composite}[t, \text{inverse}[t]], \text{Id}, t \rightarrow f[\text{ord}[x]]]$$
Theorem. The inverse of $f[x]$ is always a function.

Lemma. Converse theorem.
Comment. The special case that \( \text{ord}[x] = \omega \) was done using the GOEDEL program in 2005 January 7. The argument used then was essentially the same as the present one. The proof that \( f(\omega) \) is a bijection was done at that time using AssertTest.

variable-free formulation

A variable-free formulation of the result in the preceding section is derived here. The first step is to remove, the \( \text{ord} \) wrapper.

\[
\text{In}[17]:= \quad \text{SubstTest}[\text{implies}, \text{and}[\text{equal}[x, \text{ord}[t]], \text{not}[\text{member}[x, \text{omega}]]], \text{member}[\text{pair}[x, \text{succ}[x]], Q], t \rightarrow x] // \text{Reverse}
\]

\[
\text{Out}[17]= \quad \text{or}[\text{member}[x, \text{omega}], \text{member}[\text{pair}[x, \text{succ}[x]], Q], \text{not}[\text{member}[x, \text{OMEGA}]]] = \text{True}
\]

\[
\text{In}[18]:= \quad \text{or}[\text{member}[_{-}, \text{omega}], \text{member}[\text{pair}[_{-}, \text{succ}[_{-}}], Q], \text{not}[\text{member}[_{-}, \text{OMEGA}]]] := \text{True}
\]

Lemma. A membership rule.

\[
\text{In}[19]:= \quad (\text{member}[x, \text{fix}[	ext{composite}[\text{inverse}[y], \text{SUCC}]]]) // \text{AssertTest} /. \, y \rightarrow Q
\]

\[
\text{Out}[19]= \quad \text{member}[x, \text{fix}[	ext{composite}[Q, \text{SUCC}]]] = \text{member}[\text{pair}[x, \text{succ}[x]], Q]
\]

\[
\text{In}[20]:= \quad \text{member}[_{-}, \text{fix}[	ext{composite}[Q, \text{SUCC}]]] := \text{member}[\text{pair}[x, \text{succ}[x]], Q]
\]

Lemma. (Eliminating the variable \( x \).)

\[
\text{In}[21]:= \quad \text{Map}[\text{equal}[V, \#] & , \text{SubstTest}[\text{class}, x, \text{implies}[	ext{member}[x, u], \text{member}[x, v]], \{u \rightarrow \text{dif}[\text{OMEGA}, \text{omega}], v \rightarrow \text{fix}[	ext{composite}[Q, \text{SUCC}]]\}]]
\]

\[
\text{Out}[21]= \quad \text{subclass}[\text{OMEGA}, \text{union}[\text{omega}, \text{fix}[	ext{composite}[Q, \text{SUCC}]]]] = \text{True}
\]

\[
\text{In}[22]:= \quad % /. \, \text{Equal} \rightarrow \text{SetDelayed}
\]

Theorem. A better variable-free formulation.

\[
\text{In}[23]:= \quad \text{SubstTest}[\text{empty}, \text{domain}[\text{dif}[u, v]], \{u \rightarrow \text{composite}[\text{SUCC}, \text{id}[\text{intersection}[\text{OMEGA}, \text{complement}[\text{omega}]]]], v \rightarrow Q\}]
\]

\[
\text{Out}[23]= \quad \text{subclass}[\text{composite}[\text{SUCC}, \text{id}[\text{intersection}[\text{OMEGA}, \text{complement}[\text{omega}]]]], Q] = \text{True}
\]

\[
\text{In}[24]:= \quad \text{subclass}[\text{composite}[\text{SUCC}, \text{id}[\text{intersection}[\text{OMEGA}, \text{complement}[\text{omega}]]]], Q] := \text{True}
\]

Corollary. An alternate formulation.

\[
\text{In}[25]:= \quad \text{Map}[	ext{subclass}[\#, Q]] & , \text{Assoc}[	ext{SUCC}, \text{id}[\text{OMEGA}], \text{id}[\text{complement}[\text{omega}]]]] // \text{Reverse}
\]

\[
\text{Out}[25]= \quad \text{subclass}[\text{composite}[\text{id}[\text{OMEGA}], \text{SUCC}, \text{id}[\text{complement}[\text{omega}]]], Q] = \text{True}
\]

\[
\text{In}[26]:= \quad \text{subclass}[\text{composite}[\text{id}[\text{OMEGA}], \text{SUCC}, \text{id}[\text{complement}[\text{omega}]]], Q] := \text{True}
\]
infinite cardinals are limit ordinals

Lemma.

\[ \text{Lemma.} \]

\[ \text{Theorem.} \]

\[ \text{Simplification rule.} \]

\[ \text{Lemma.} \]

\[ \text{This the (implicit) step } p_2 \Rightarrow \neg p_5 \text{ used in the proof of the main theorem below.} \]

\[ \text{Theorem.} \]

\[ \text{This is the step } p_2 \Rightarrow p_3 \text{ in the proof of main theorem.} \]

\[ \text{Lemma.} \]

\[ \text{Theorem.} \]

\[ \text{This is the step } p_1 \land p_2 \Rightarrow p_4. \]
Main Theorem. Any infinite cardinal is a limit ordinal.

In[38]:= (% /. \(x\to x_\)) /. Equal\to SetDelayed

Corollary. (Remove wrappers.)

In[41]:= SubstTest[implies, equal[x, card[ord[t]]]],
or[equal[U[x], x], member[x, omega], \(t\to x\)] // Reverse

Theorem. Eliminate the variable.

In[43]:= Map[equal[V, #] & SubstTest{class, x, implies[member[x, u], member[x, v]],
\{u \to fix[CARD], v \to union[omega, fix[BIGCUP]]\}]}

In[44]:= subclass[fix[CARD], union[omega, fix[BIGCUP]]] := True

\(\aleph\) formula

Theorem. (This statement contains a redundant literal.)

In[45]:= SubstTest[implies, and[member[u, v], subclass[v, w]], member[u, w],
\{u \to APPLY[ALEPH, x], v \to dif[fix[CARD], omega], w \to fix[BIGCUP]\}] // Reverse

Out[45]= or[equal[APPLY[ALEPH, x], U[APPLY[ALEPH, x]]], not[member[x, OMEGA]]] = True

In[46]:= (% /. \(x\to x_\)) /. Equal\to SetDelayed

Lemma. (Needed to eliminate the redundant literal.)

In[47]:= SubstTest[implies, equal[t, V], equal[U[t], t], \(t\to APPLY[ALEPH, x]\)] // Reverse

Out[47]= or[equal[APPLY[ALEPH, x], U[APPLY[ALEPH, x]]], member[x, OMEGA]] = True

In[48]:= (% /. \(x\to x_\)) /. Equal\to SetDelayed
The redundant literal can now be eliminated to obtain a wrapper-free simplification rule.

**Theorem.** Every aleph is a limit ordinal.

\begin{verbatim}
In[49]:= SubstTest[and, implies[p, q], or[p, q],
        {p \rightarrow member[x, OMEGA], q \rightarrow equal[APPLY[ALEPH, x], U[APPLY[ALEPH, x]]]}]
Out[49]= equal[APPLY[ALEPH, x], U[APPLY[ALEPH, x]]] = True

In[50]:= U[APPLY[ALEPH, x_]] := APPLY[ALEPH, x]
In[51]:= Map[composite[VERTSECT[#], id[OMEGA]] &,
        SubstTest[reify, x, U[APPLY[funpart[t], x]], t \rightarrow ALEPH]]
Out[51]= composite[BIGCUP, ALEPH] = ALEPH

In[52]:= composite[BIGCUP, ALEPH] := ALEPH
\end{verbatim}