# axch => image[Q, Ω] = V

Johan G. F. Belinfante
2011 July 8

In[1]:= **SetDirectory["l:"]; << goedel.11jul08a**

```
:Package Title: goedel.11jul08a                2011 July 8 at 10:20 a.m.

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

It is now:  2011 Jul 8 at 16:47

Loading Simplification Rules

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

weightlimit = 40

Loading completed.

It is now:  2011 Jul 8 at 16:58
```

## summary

In this notebook it is shown that the axiom of choice is equivalent to the statement that every set is equipollent to some ordinal.

## introduction

The goal in this notebook is to show that **axch ⟺ V = image[Q, Ω]**. The implication in one direction has already been derived 2005 July 20 in the posted notebook **wo-ac.nb**.

In[2]:= **implies[equal[V, image[Q, OMEGA]], axch]**

Out[2]= True

In deriving the converse, one can use Zorn's lemma because the hypothesis is that **axch** holds. The following simple version of Zorn's lemma will suffice:

In[3]:= **implies[and[axch, equal[Uchains[t], t], member[t, V]], not[empty[maximal[S, t]]]]**

Out[3]= True

The intention is to apply Zorn's lemma to the class **t = BIJ ⋂ image[inverse[IMAGE[FIRST]], Ω] ⋂ P[V × x]**, where **x** is any set. This class **t** is the class of all bijections whose domain is an ordinal, and whose range is contained in **x**.

```
In[4]:= member[w, intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, x]]]]

Out[4]= and[FUNCTION[inverse[w]], member[w, map[domain[w], x]], member[domain[w], OMEGA]]
```

This class **t** is closed under chain unions because it is the intersection of three classes each of which has this property. The nontirvial fact that **t** is a set was recently shown in the posted notebook **hart-bij.nb** based on the theorem that the Hartogs number of any set is a set. (This fact about Hartogs numbers was derived 2010 November 8 in the posted notebook **hart-do.nb**.)

Theorem. Application of Zorn's lemma to the set **t**.

```
In[5]:= SubstTest[implies, and[axch, equal[Uchains[t], t], member[t, V]],
          not[subclass[t, image[inverse[PS], t]]], t -> intersection[BIJ,
            image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]] // Reverse

Out[5]= or[not[axch], not[subclass[
            intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]],
            image[inverse[PS], intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA],
              P[cart[V, setpart[x]]]]]]]] == True

In[6]:= (% /. x → x_) /. Equal → SetDelayed
```

Restatement.

```
In[7]:= implies[axch, not[empty[maximal[S,
            intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]]]

Out[7]= True
```

To complete the proof it must be shown that if a set **x** is not equipollent to an ordinal, then there is no maximal bijection from any ordinal into **x**.

## maximal bijections

In this section it will be shown that if a set **x** is not equipollent to any ordinal, then there is no maximal bijection from an ordinal into **x**. The idea is to show that if **w** is any bijection from an ordinal into **x**, then **range[w]** cannot be equal to **x**, and so if **y** is any member of **x – range[w]**, one can adjoin the point **pair[domain[w], y]** to **w** to obtain a bijection from the successor of the ordinal **domain[w]** into **x**.

Lemma. Adding a new point to an inverse function.

```
In[8]:= SubstTest[implies, and[FUNCTION[t], not[member[x, domain[t]]]],
          FUNCTION[union[t, set[PAIR[x, y]]]], t → inverse[w]] // Reverse

Out[8]= or[FUNCTION[union[cart[set[x], set[y]], inverse[w]]],
          member[x, range[w]], not[FUNCTION[inverse[w]]]] == True

In[9]:= or[FUNCTION[union[cart[set[x_], set[y_]], inverse[w_]]],
          member[x_, range[w_]], not[FUNCTION[inverse[w_]]]] := True
```

Lemma. If one can add the point **PAIR[domain[w], setpart[w]]** to **w ∈ t = BIJ ⋂ image[inverse[IMAGE[FIRST]], Ω] ⋂ P[V × x]** to obtain a larger element of **t**, then **w** is not a maximal element of **t**.

```
In[10]:= SubstTest[or, member[u, w], member[w, image[inverse[PS], t]], not[member[u, z]],
           not[member[union[w, set[u]], t]], {u -> PAIR[domain[w], setpart[y]],
            t -> intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]],
            z → cart[V, V]}] // Reverse

Out[10]= or[member[w, image[inverse[PS],
              intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           member[pair[domain[w], setpart[y]], w],
           not[FUNCTION[union[w, cart[set[domain[w]], set[setpart[y]]]]]],
           not[FUNCTION[union[cart[set[setpart[y]], set[domain[w]]], inverse[w]]]],
           not[member[w, V]], not[member[domain[w], OMEGA]],
           not[member[setpart[y], setpart[x]]], not[subclass[w, cart[V, setpart[x]]]]] == True

In[11]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

The following a general result about adding a new point to a function.

Theorem. If **t: x ⟶ y** and **u** does not belong to **x**, then **t ⋃ ({u} × {v})** is a function.

```
In[12]:= Map[not, SubstTest[and, implies[p1, p3], implies[and[p1, p2], p4],
            implies[and[p3, p4], p5], not[implies[and[p1, p2], p5]], {p1 → member[t, map[x, y]],
             p2 → not[member[u, x]], p3 → FUNCTION[t], p4 → not[member[u, domain[t]]],
             p5 → FUNCTION[union[t, cart[set[u], set[v]]]]}]] // Reverse

Out[12]= or[FUNCTION[union[t, cart[set[u], set[v]]]],
           member[u, x], not[member[t, map[x, y]]]] == True

In[13]:= or[FUNCTION[union[t_, cart[set[u_], set[v_]]]],
           member[u_, x_], not[member[t_, map[x_, y_]]]] := True
```

Special theorem. If **x ∈ Ω** and **t: x ⟶ z**, then **t ⋃ ({x} × {y})** is a function.

```
In[14]:= Map[not, SubstTest[and, implies[p2, p3], implies[and[p1, p3], p4],
            not[implies[and[p1, p2], p4]], {p1 -> member[t, map[x, z]], p2 -> member[x, OMEGA],
             p3 → not[member[x, x]], p4 -> FUNCTION[union[t, cart[set[x], set[y]]]]}]] // Reverse

Out[14]= or[FUNCTION[union[t, cart[set[x], set[y]]]],
           not[member[t, map[x, z]]], not[member[x, OMEGA]]] == True

In[15]:= or[FUNCTION[union[t_, cart[set[x_], set[y_]]]],
           not[member[t_, map[x_, z_]]], not[member[x_, OMEGA]]] := True
```

It follows in particular that if **w ∈ BIJ ⋂ image[inverse[IMAGE[FIRST]], Ω] ⋂ P[V × x]** then **w ⋃ ({domain[w]} × {setpart[y]})** is a function.

```
In[16]:= or[FUNCTION[union[w, cart[set[domain[w]], set[setpart[y]]]]],
          not[FUNCTION[inverse[w]]], not[member[w, map[domain[w], x]]],
          not[member[domain[w], OMEGA]]]

Out[16]= True
```

For the present purposes one also needs the following special fact which uses the fact that an ordinal cannot be a member of itself.

Special Theorem. If **x** $\in$ **Ω** and **w: x ⟶ z**, then **pair[x, y]** does not belong to **w**.

```
In[17]:= Map[not, SubstTest[and, implies[p2, p6], implies[and[p2, p6], p14],
            not[implies[p2, p14]], {p2 → and[member[x, OMEGA], member[w, map[x, z]]],
             p6 → not[member[x, x]], p14 → not[member[pair[x, y], w]]}]] // Reverse

Out[17]= or[not[member[w, map[x, z]]], not[member[x, OMEGA]], not[member[pair[x, y], w]]] == True

In[18]:= or[not[member[w_, map[x_, z_]]],
           not[member[x_, OMEGA]], not[member[pair[x_, y_], w_]]] := True
```

Lemma. A technical simplification.

```
In[19]:= Map[not, SubstTest[and, implies[p2, p13], implies[p2, p14],
            implies[and[p8, p13, p14], p15], not[implies[and[p2, p8], p15]], {p2 -> member[w,
               intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]],
             p8 → and[FUNCTION[union[w, set[PAIR[domain[w], setpart[y]]]]],
               member[union[w, set[PAIR[domain[w], setpart[y]]]], V],
               FUNCTION[inverse[union[w, set[PAIR[domain[w], setpart[y]]]]]],
               member[domain[union[w, set[PAIR[domain[w], setpart[y]]]]], OMEGA],
               subclass[range[union[w, set[PAIR[domain[w], setpart[y]]]]], setpart[x]]],
             p13 -> subclass[w, cart[V, setpart[x]]],
             p14 → not[member[pair[domain[w], setpart[y]], w]],
             p15 -> member[w, image[inverse[PS], intersection[BIJ,
                 image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]}]] // Reverse

Out[19]= or[member[w, image[inverse[PS], intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA],
             P[cart[V, setpart[x]]]]]], not[FUNCTION[inverse[w]]],
           not[FUNCTION[union[w, cart[set[domain[w]], set[setpart[y]]]]]],
           not[FUNCTION[union[cart[set[setpart[y]], set[domain[w]]], inverse[w]]]],
           not[member[w, map[domain[w], setpart[x]]]],
           not[member[domain[w], OMEGA]], not[member[setpart[y], setpart[x]]]] == True

In[20]:= (% /. {w → w_, x → x_, y → y_}) /. Equal → SetDelayed
```

The derivation of the following theorem goes faster if one omits the proof step commented out with **(* ⋯ *)**.

Theorem. If **w** $\in$ **t = BIJ ⋂ image[inverse[IMAGE[FIRST]], Ω] ⋂ P[V × x]** and **y** is an element of **x – range[w]**, then **w** is not a maximal member of **t**.

```
In[21]:= Map[not, SubstTest[and, implies[p2, p8] , implies[p2, p9] ,
            implies[and[p2, p5b], p10], implies[and[p2, p5a], p11], implies[and[p2, p5a], p12],
            (* implies[and[p2,p8,p9,p10,p11,p12,p15], *)
            not[implies[and[p2, p5a, p5b], p15]], {p2 -> member[w,
               intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]],
             p5a → member[setpart[y], setpart[x]], p5b -> not[member[setpart[y], range[w]]],
             p8 → FUNCTION[union[w, set[PAIR[domain[w], setpart[y]]]]],
             p9 → member[union[w, set[PAIR[domain[w], setpart[y]]]], V],
             p10 → FUNCTION[inverse[union[w, set[PAIR[domain[w], setpart[y]]]]]],
             p11 → member[domain[union[w, set[PAIR[domain[w], setpart[y]]]]], OMEGA],
             p12 → subclass[range[union[w, set[PAIR[domain[w], setpart[y]]]]], setpart[x]],
             p13 -> subclass[w, cart[V, setpart[x]]],
             p14 → not[member[pair[domain[w], setpart[y]], w]],
             p15 -> member[w, image[inverse[PS], intersection[BIJ,
                 image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]}]] // Reverse

Out[21]= or[member[w, image[inverse[PS],
             intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           member[setpart[y], range[w]], not[FUNCTION[inverse[w]]],
           not[member[w, map[domain[w], setpart[x]]]],
           not[member[domain[w], OMEGA]], not[member[setpart[y], setpart[x]]]] == True

In[22]:= or[member[w_, image[inverse[PS],
             intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x_]]]]]],
           member[setpart[y_], range[w_]], not[FUNCTION[inverse[w_]]],
           not[member[domain[w_], OMEGA]], not[member[w_, map[domain[w_], setpart[x_]]]],
           not[member[setpart[y_], setpart[x_]]]] := True
```

Restatement.

```
In[23]:= implies[member[setpart[y], s], not[member[w, t]]] /.
          {s → dif[setpart[x], range[w]], t → maximal[S,
             intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]}

Out[23]= True
```

## eliminating the variable y

In this section, the variable **y** is eliminated. This turned out to be technically difficult, and required temporarily clearing the **cond** and **simplify** flags.

Lemma. (Eliminating the **setpart** wrapper on **y**.)

```
In[24]:= SubstTest[implies, equal[y, setpart[t]], implies[member[y, dif[setpart[x], range[w]]],
            not[member[w, maximal[S, intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA],
                P[cart[V, setpart[x]]]]]]]]], t → y] // Reverse

Out[24]= or[member[w, image[inverse[PS],
              intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           member[y, range[w]], not[FUNCTION[inverse[w]]],
           not[member[w, map[domain[w], setpart[x]]]],
           not[member[y, setpart[x]]], not[member[domain[w], OMEGA]]] == True

In[25]:= (% /. {x → x_, y → y_, w → w_}) /. Equal → SetDelayed
```

Lemma. An application of double negation.

```
In[26]:= or[member[w, image[inverse[PS],
              intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           member[setpart[y], range[w]], not[FUNCTION[w]], not[FUNCTION[inverse[w]]],
           not[member[w, V]], not[member[domain[w], OMEGA]],
           not[member[setpart[y], setpart[x]]],
           not[subclass[w, cart[V, setpart[x]]]]] // NotNotTest

Out[26]= or[member[w, image[inverse[PS],
              intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           member[setpart[y], range[w]], not[FUNCTION[w]], not[FUNCTION[inverse[w]]],
           not[member[w, V]], not[member[domain[w], OMEGA]],
           not[member[setpart[y], setpart[x]]], not[subclass[w, cart[V, setpart[x]]]]] == True

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

The hard step is to use **class** to eliminate **y**. Here two flags were termporarily cleared.

```
In[28]:= cond = False; simplify= False;
```

Lemma. (Eliminating the variable **y**. Here again double negation was used to simplify the final result.)

```
In[29]:= Map[equal[V, #] &, SubstTest[class, y, member[setpart[y], t],
            t → union[complement[image[V, intersection[BIJ, set[w]]]], complement[
               image[V, intersection[OMEGA, set[domain[w]]]]], complement[setpart[x]],
              image[V, intersection[OMEGA, image[IMAGE[FIRST], intersection[BIJ,
                  complement[set[w]], image[S, set[w]], P[cart[V, setpart[x]]]]]]]], image[V,
               intersection[w, complement[cart[V, setpart[x]]]]], range[w]]]] // MapNotNot

Out[29]= or[member[w, image[inverse[PS],
              intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           not[FUNCTION[inverse[w]]], not[member[w, map[domain[w], setpart[x]]]],
           not[member[domain[w], OMEGA]], subclass[setpart[x], range[w]]] == True

In[30]:= (% /. {x → x_, w → w_}) /. Equal → SetDelayed
```

Restatement.

```
In[31]:= implies[not[subclass[setpart[x], range[w]]], not[member[w, maximal[S,
             intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]]]

Out[31]= True
```

The cleared flags can now be reset.

```
In[32]:= cond = True; simplify = True;
```

## final steps

Lemma. If a set **x** is not equipollent to an ordinal, then any bijection **w** from an ordinal into **x** is not maximal.

```
In[33]:= Map[not, SubstTest[and, implies[and[p1, p2], p3], implies[p2, p4],
           implies[and[p3, p4], p5], implies[p5, p6], not[implies[and[p1, p2], p6]],
           {p1 -> not[member[setpart[x], image[Q, OMEGA]]], p2 -> member[w,
              intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]],
            p3 -> not[equal[setpart[x], range[w]]], p4 → subclass[range[w], setpart[x]],
            p5 -> not[subclass[setpart[x], range[w]]],
            p6 → not[member[w, maximal[S, intersection[BIJ,
                 image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]]}]] // Reverse

Out[33]= or[member[w, image[inverse[PS],
             intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]],
           member[setpart[x], image[Q, OMEGA]], not[FUNCTION[inverse[w]]],
           not[member[w, map[domain[w], setpart[x]]]], not[member[domain[w], OMEGA]]] == True

In[34]:= (% /. {x → x_, w → w_}) /. Equal → SetDelayed
```

Theorem. Eliminate the variable **w**. (This takes quite a while.)

```
In[35]:= Map[equal[V, #] &, SubstTest[class, w,
          implies[not[member[setpart[x], image[Q, OMEGA]]], not[member[w, t]]], t -> maximal[S,
            intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]]]

Out[35]= or[member[setpart[x], image[Q, OMEGA]],
          subclass[intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA],
            P[cart[V, setpart[x]]]], image[inverse[PS], intersection[BIJ,
             image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]] == True

In[36]:= or[member[setpart[x_], image[Q, OMEGA]],
          subclass[intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA],
            P[cart[V, setpart[x_]]]], image[inverse[PS], intersection[BIJ,
             image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x_]]]]]]] := True
```

Restatement. If a set **x** is not equipollent to an ordinal, then there is no maximal bijection from an ordinal into **x**.

```
In[37]:= implies[not[member[setpart[x], image[Q, OMEGA]]], empty[maximal[S,
            intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]]

Out[37]= True
```

Theorem. The axiom of choice implies that every set is equipollent to an ordinal.

```
In[38]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
           {p1 → axch, p2 -> not[subclass[intersection[BIJ, image[inverse[IMAGE[FIRST]], OMEGA],
                P[cart[V, setpart[x]]]], image[inverse[PS], intersection[BIJ,
                 image[inverse[IMAGE[FIRST]], OMEGA], P[cart[V, setpart[x]]]]]]]],
            p3 → member[setpart[x], image[Q, OMEGA]]}]] // Reverse

Out[38]= or[member[setpart[x], image[Q, OMEGA]], not[axch]] == True

In[39]:= or[member[setpart[x_], image[Q, OMEGA]], not[axch]] := True
```

Lemma. Eliminating the variable **x**.

```
In[40]:= Map[equal[V, #] &,
          SubstTest[class, x, or[member[setpart[x], t], not[axch]], t -> image[Q, OMEGA]]]

Out[40]= or[equal[V, image[Q, OMEGA]], not[axch]] == True

In[41]:= % /. Equal → SetDelayed
```

The following is a corollary.

Corollary. If the axiom of choice holds, then the relation **SMALLER** is well-founded.

```
In[42]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
           {p1 → axch, p2 → equal[V, image[Q, OMEGA]], p3 → WELLFOUNDED[SMALLER]}]] // Reverse

Out[42]= or[not[axch], WELLFOUNDED[SMALLER]] == True

In[43]:= or[not[axch], WELLFOUNDED[SMALLER]] := True
```

Theorem. The axiom of choice is equivalent to the equation **V = image[Q, Ω]**.

```
In[44]:= equiv[equal[V, image[Q, OMEGA]], axch]

Out[44]= True

In[45]:= equal[V, image[Q, OMEGA]] := axch
```

Another immediate corollary is the well-ordering theorem: **axch** implies that every set is the fixed-point class for some well-ordering.

Well-ordering theorem. The axiom of choice implies that every set can be well-ordered.

```
In[46]:= Map[implies[axch, #] &,
          SubstTest[and, equal[u, V], subclass[u, v], {u → image[Q, OMEGA], v → image[FIX, WO]}]]

Out[46]= or[equal[V, image[IMAGE[inverse[DUP]], WO]], not[axch]] == True

In[47]:= or[equal[V, image[IMAGE[inverse[DUP]], WO]], not[axch]] := True
```