# cardinality rules for singletons

Johan G. F. Belinfante
2012 June 26

```
In[1]:= SetDirectory["l:"]; << goedel.12jun25a

        :Package Title: goedel.12jun25a                    2012 June 25 at 1:45 a.m.

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

        It is now:  2012 Jun 26 at 20:5

        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 Jun 26 at 20:22
```

## summary

If **x ⊂ y**, then **card[x] ⊂ card[y]**, whether or not **x** and **y** are equipollent to ordinals. The axiom of choice need not be assumed.

## derivation

Theorem.

```
In[23]:= Map[not, SubstTest[and, implies[and[p1, p2], p3], implies[and[p1, p3], p4],
           not[implies[and[p1, p2], p4]], {p1 → subclass[y, x], p2 → member[x, image[Q, OMEGA]],
            p3 → member[y, image[Q, OMEGA]], p4 → subclass[card[y], card[x]]}]] // Reverse

Out[23]= or[not[member[card[x], card[y]]], not[subclass[y, x]]] == True

In[24]:= or[not[member[card[x_], card[y_]]], not[subclass[y_, x_]]] := True
```

Corollary.

```
In[26]:= SubstTest[implies, subclass[x, t], subclass[card[x], card[t]], t → set[y]] // Reverse //
          MapNotNot

Out[26]= or[equal[0, x], not[member[set[0], card[x]]], not[subclass[x, set[y]]]] == True

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

Theorem.

```
In[28]:= SubstTest[and, implies[p, q], or[p, q],
          {p -> equal[0, x], q -> or[not[member[set[0], card[x]]], not[subclass[x, set[y]]]]}]

Out[28]= or[not[member[set[0], card[x]]], not[subclass[x, set[y]]]] == True

In[29]:= or[not[member[set[0], card[x_]]], not[subclass[x_, set[y_]]]] := True
```

Corollary.

```
In[32]:= Map[not, SubstTest[implies, subclass[t, set[x]], not[member[set[0], card[t]]], t → Z]] //
          Reverse

Out[32]= subclass[Z, set[x]] == False

In[33]:= subclass[Z, set[x_]] := False
```

Theorem. A variable-free formulation.

```
In[35]:= Map[equal[V, domain[#]] &, SubstTest[reify, x,
           case[implies[member[x, s], subclass[card[first[x]], card[second[x]]]], s → S]]

Out[35]= subclass[composite[id[image[Q, OMEGA]], S], composite[inverse[CARD], S, CARD]] == True

In[36]:= subclass[composite[id[image[Q, OMEGA]], S], composite[inverse[CARD], S, CARD]] := True
```