# theorem FIN-IND2 on FINITE induction


## Johan G. F. Belinfante
## 2014 April 18


```
In[1]:= SetDirectory["l:"]; << goedel.14apr14a

        :Package Title: goedel.14apr14a                    2014 April 14 at 6:25 p.m.

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

        It is now:  2014 Apr 18 at 13:48

        Loading Simplification Rules

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

        weightlimit = 40

        Loading completed.

        It is now:  2014 Apr 18 at 14:5
```

---

## summary

At the conference FTP2000 in St. Andrews, Scotland in July 2000, the author reported on three clause formulations of **FINITE** induction that had been proved using Bill McCune's automated reasoning program **Otter**.

```
In[2]:=  "Johan G. F. Belinfante, Computer Proofs about Finite and
         Regular Sets: The Unifying Concept of Subvariance,in special issue
         on First Order Theorem Proving, edited by P. Baumgartner and H.
         Zhang, Journal of Symbolic Computation, vol. 36, pp. 271-285 (2003).";
```

The following two of these three clause formulations of **FINITE** induction are already available in the **GOEDEL** program. (These are called **FIN-IND1** and **FIN-K-1** respectively.)

```
In[3]:= implies[and[member[0, x], subclass[image[CUP, cart[x, range[SINGLETON]]], x]],
         subclass[FINITE, x]]

Out[3]= True

In[4]:= implies[and[member[0, x], invariant[K, x]], subclass[FINITE, x]]

Out[4]= True
```

In this notebook, a short derivation of the third version, called **FIN-IND2**, is presented.

## remarks

In the **GOEDEL** program, the **FIN-IND1** clause is automatically rewritten to the **FIN-K-1** clause via the following term rewrite rule.

```
In[5]:= image[CUP, cart[x, range[SINGLETON]]]

Out[5]= union[image[K, x], intersection[x, complement[set[0]]]]
```

This rewrite rule transforms the second literal in the **FIN-IND1** clause as follows.

```
In[6]:= subclass[image[CUP, cart[x, range[SINGLETON]]], x]

Out[6]= subclass[image[K, x], x]
```

## derivation

Lemma.

```
In[7]:= Map[implies[subclass[image[CUP, cart[x, image[SINGLETON, y]]], x], invariant[#, x]] &,
          Assoc[CUP, id[cart[P[y], P[y]]],
           composite[id[cart[V, range[SINGLETON]]], inverse[FIRST]]]] // Reverse

Out[7]= or[not[subclass[image[CUP, cart[x, image[SINGLETON, y]]], x]],
          subclass[intersection[image[K, x], P[y]], x]] == True

In[8]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[9]:= SubstTest[implies, and[member[0, t], invariant[K, t]],
          subclass[FINITE, t], t → union[x, complement[P[y]]]] // Reverse

Out[9]= or[not[member[0, x]], not[subclass[intersection[image[K, x], P[y]], x]],
          subclass[intersection[FINITE, P[y]], x]] == True

In[10]:= (% /. {x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem **FIN-IND2**.

```
In[11]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3], not[implies[p1, p3]],
            {p1 -> subclass[image[CUP, cart[x, image[SINGLETON, y]]], x],
             p2 -> subclass[intersection[image[K, x], P[y]], x],
             p3 -> or[not[member[0, x]], subclass[intersection[FINITE, P[y]], x]]}]] // Reverse

Out[11]= or[not[member[0, x]], not[subclass[image[CUP, cart[x, image[SINGLETON, y]]], x]],
           subclass[intersection[FINITE, P[y]], x]] == True
```

```
In[12]:= or[not[member[0, x_]], not[subclass[image[CUP, cart[x_, image[SINGLETON, y_]]], x_]],
           subclass[intersection[FINITE, P[y_]], x_]] := True
```

Comment. The following corollary of **FIN-IND2** is already available in the **GOEDEL** program.

```
In[13]:= implies[and[member[0, x], subclass[image[SINGLETON, y], x],
           subclass[image[CUP, cart[x, x]], x]], subclass[intersection[FINITE, P[y]], x]]

Out[13]= True
```

## a variant of FIN-K-1

In this section, another version of **FINITE** induction is derived that resembles the clause **FIN-K-1**, but replaces the hypothesis **0 ∈ x** with **range[SINGLETON] ⊂ x**.

Lemma.

```
In[14]:= SubstTest[implies, and[member[0, t], invariant[K, t]],
           subclass[FINITE, t], t → union[x, set[0]]] // Reverse

Out[14]= or[not[subclass[image[K, x], union[x, set[0]]]],
           not[subclass[range[SINGLETON], union[x, set[0]]]],
           subclass[FINITE, union[x, set[0]]]] == True

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

Theorem.

```
In[16]:= Map[not, SubstTest[and, implies[p1, p2], implies[p1, p3], implies[and[p2, p3], p4],
              not[implies[p1, p4]], {p1 → and[invariant[K, x], subclass[range[SINGLETON], x]],
               p2 -> subclass[image[K, x], union[x, set[0]]],
               p3 -> subclass[range[SINGLETON], union[x, set[0]]],
               p4 -> subclass[FINITE, union[x, set[0]]]}]] // Reverse

Out[16]= or[not[subclass[image[K, x], x]],
           not[subclass[range[SINGLETON], x]], subclass[FINITE, union[x, set[0]]]] == True

In[17]:= or[not[subclass[image[K, x_], x_]],
           not[subclass[range[SINGLETON], x_]], subclass[FINITE, union[x_, set[0]]]] := True
```