# linear differences, part 1. basics

Johan G. F. Belinfante
2007 March 2; corrected 2007 April 4

```
In[1]:= SetDirectory["l:"]; << goedel91.01b; << tools.m

        :Package Title: goedel91.01b     corrected 2007 April 4 at 7:45 p.m.

        It is now:  2007 Apr 4 at 19:44

        Loading Simplification Rules

        TOOLS.M                      Revised 2007 January 7

        weightlimit = 40
```

## summary

Art Quaife fifteen years ago presented an automated development of the arithmetic of natural numbers, using William McCune's automated reasoning program **Otter**. In the course of this work, he introduced and studied a special predicate for statements about linear differences.

```
In[2]:= "Art Quaife, Automated Development of Fundamental Mathematical
           Theories, Appendix 3. Theorems Proved in Peano's Arithmetic,
           Kluwer Academic Publishers, Dordrecht, 1992. See pages 200-202.";
```

The general setting in this notebook is Gödel's class theory, whereas Quaife developed arithmetic based on a set of axioms for arithmetic. The set of linear differences is defined by the following **class**-wrapped membership rule.

```
In[3]:= Begin["Goedel`Private`"];

In[4]:= FirstMatch[class[t_, member[w_, HoldPattern[ld[x_, y_]]]]]

Out[4]= class[t_, member[w_, ld[x_, y_]]] := ReleaseHold[Module[{u = Unique[],
          v = Unique[]}, class[t, exists[u, v, and[member[PAIR[PAIR[w, v],
          u], NATADD]], member[PAIR[x, u], DIV]], member[PAIR[y, v], DIV]]]]]]]
```

## normalization

The set of linear differences is normalized by the following rewrite rule:

```
In[5]:= ld[x, y] // Normality // Reverse

Out[5]= image[image[inverse[NATADD], image[DIV, set[x]]], image[DIV, set[y]]] == ld[x, y]

In[6]:= image[image[inverse[NATADD], image[DIV, set[x_]]], image[DIV, set[y_]]] := ld[x, y]
```

Free variables such as **x** and **y** can be any classes, not necessarily natural numbers. For the theorems, the restrictions to natural numbers can be given either by introducing membership literals, or by using **nat** wrappers. Generally speaking, the use of **nat** wrappers yields simpler expressions, and therefore faster derivations, but on occasion, especially when variables are to be eliminated, these wrappers need to be replaced by corresponding literals.

## special cases

Theorem.

```
In[7]:= SubstTest[image, image[inverse[NATADD], image[DIV, set[x]]], image[DIV, set[y]], y → 0]

Out[7]= ld[x, 0] == image[DIV, set[x]]

In[8]:= ld[x_, 0] := image[DIV, set[x]]
```

Theorem.

```
In[9]:= SubstTest[image, image[inverse[NATADD], image[DIV, set[x]]], image[DIV, set[y]], x → 0]

Out[9]= ld[0, y] == intersection[image[V, intersection[omega, set[y]]], set[0]]

In[10]:= ld[0, y_] := intersection[image[V, intersection[omega, set[y]]], set[0]]
```

Theorem.

```
In[11]:= SubstTest[image, image[inverse[NATADD], image[DIV, set[x]]], image[DIV, set[y]], y → x]

Out[11]= ld[x, x] == image[DIV, set[x]]

In[12]:= ld[x_, x_] := image[DIV, set[x]]
```

## sethood of ld[x,y]

Theorem.

```
In[13]:= SubstTest[subclass, image[u, v], range[u],
           {u → rotate[NATADD], v → cart[image[DIV, set[x]], image[DIV, set[y]]]}] // Reverse

Out[13]= subclass[ld[x, y], omega] == True

In[14]:= subclass[ld[x_, y_], omega] := True
```

Corollary.

```
In[15]:= SubstTest[implies, subclass[t, omega], member[t, V], t → ld[x, y]] // Reverse

Out[15]= member[ld[x, y], V] == True

In[16]:= member[ld[x_, y_], V] := True
```

## empty ld[x,y]

Lemma.

```
In[17]:= ImageComp[S, DIV, x] // Reverse

Out[17]= image[S, image[DIV, x]] == image[V, intersection[omega, x]]

In[18]:= image[S, image[DIV, x_]] := image[V, intersection[omega, x]]
```

Theorem. The set **ld[x, y]** of linear differences is empty when **x** or **y** is not a natural number.

```
In[19]:= SubstTest[equal, 0, image[u, v],
           {u → rotate[NATADD], v → cart[image[DIV, set[x]], image[DIV, set[y]]]}] // Reverse

Out[19]= equal[0, ld[x, y]] == or[not[member[x, omega]], not[member[y, omega]]]

In[20]:= equal[0, ld[x_, y_]] := or[not[member[x, omega]], not[member[y, omega]]]
```

## a membership rule

```
In[21]:= SubstTest[member, y,
          image[image[inverse[NATADD], t], image[DIV, set[x]]], t → image[DIV, set[z]]]

Out[21]= member[pair[x, y], composite[image[inverse[NATADD], image[DIV, set[z]]], DIV]] ==
          member[y, ld[z, x]]

In[22]:= member[pair[x_, y_], composite[image[inverse[NATADD], image[DIV, set[z_]]], DIV]] :=
          member[y, ld[z, x]]
```

## membership rule for composite[inverse[NATADD],DIV]

Lemma.

```
In[23]:= SubstTest[and, member[t, omega], member[pair[x, t], DIV], t → natadd[y, z]] // Reverse

Out[23]= and[member[y, omega], member[z, omega], member[pair[x, natadd[y, z]], DIV]] ==
          member[pair[x, natadd[y, z]], DIV]

In[24]:= and[member[y_, omega], member[z_, omega], member[pair[x_, natadd[y_, z_]], DIV]] :=
          member[pair[x, natadd[y, z]], DIV]
```

Lemma.

```
In[25]:= SubstTest[and, member[y, omega], and[member[y, omega], member[z, omega], p],
           p -> member[pair[x, natadd[y, z]], DIV]] // Reverse

Out[25]= and[member[y, omega], member[pair[x, natadd[y, z]], DIV]] ==
          member[pair[x, natadd[y, z]], DIV]

In[26]:= and[member[y_, omega], member[pair[x_, natadd[y_, z_]], DIV]] :=
          member[pair[x, natadd[y, z]], DIV]
```

Theorem.

```
In[27]:= (member[pair[x, pair[y, z]], composite[inverse[funpart[t]], v]] // AssertTest) /.
          {t → NATADD, v → DIV}

Out[27]= member[pair[x, pair[y, z]], composite[inverse[NATADD], DIV]] ==
          member[pair[x, natadd[y, z]], DIV]

In[28]:= member[pair[x_, pair[y_, z_]], composite[inverse[NATADD], DIV]] :=
          member[pair[x, natadd[y, z]], DIV]
```

## a basic membership theorem for ld[x,y]

In this section a basic theorem about membership in the set of linear differences is derived.

Lemma.

```
In[29]:= SubstTest[and, member[y, omega], member[y, v], member[pair[z, natadd[w, y]], DIV], v → V]

Out[29]= and[member[y, V], member[pair[z, natadd[w, y]], DIV]] ==
          member[pair[z, natadd[w, y]], DIV]

In[30]:= and[member[y_, V], member[pair[z_, natadd[w_, y_]], DIV]] :=
          member[pair[z, natadd[w, y]], DIV]
```

Theorem.

```
In[31]:= SubstTest[implies, and[member[pair[z, w], composite[Id, v]],
             member[pair[w, x], composite[Id, u]]], member[pair[z, x], composite[u, v]],
            {v → DIV, u -> image[inverse[NATADD], image[DIV, set[y]]]}] // Reverse // MapNotNot

Out[31]= or[member[x, ld[y, z]],
           not[member[pair[y, natadd[w, x]], DIV]], not[member[pair[z, w], DIV]]] == True

In[32]:= or[member[x_, ld[y_, z_]], not[member[pair[y_, natadd[w_, x_]], DIV]],
           not[member[pair[z_, w_], DIV]]] := True
```

Corollary. (What one thinks is a linear difference actually is a linear difference, provided it makes sense.)

```
In[33]:= SubstTest[implies,
           and[member[pair[y, t], DIV], member[pair[x, natadd[w, t]], DIV]], member[w, ld[x, y]],
           {t → natmul[v, y], w → natsub[natmul[u, x], natmul[v, y]]}] // Reverse

Out[33]= or[member[natsub[natmul[u, x], natmul[v, y]], ld[x, y]],
           not[member[u, omega]], not[member[v, omega]], not[member[x, omega]],
           not[member[y, omega]], not[subclass[natmul[v, y], natmul[u, x]]]] == True

In[34]:= (% /. {u → u_, v → v_, x → x_, y → y_}) /. Equal → SetDelayed
```

A converse also holds:

```
In[35]:= SubstTest[implies, member[w, z], member[w, V],
           {w -> natsub[natmul[u, x], natmul[v, y]], z → ld[x, y]}] // Reverse

Out[35]= or[and[member[u, omega], member[v, omega], member[x, omega],
            member[y, omega], subclass[natmul[v, y], natmul[u, x]]],
           not[member[natsub[natmul[u, x], natmul[v, y]], ld[x, y]]]] == True

In[36]:= (% /. {u → u_, v → v_, x → x_, y → y_}) /. Equal → SetDelayed
```

Theorem.

```
In[37]:= equiv[member[natsub[natmul[u, x], natmul[v, y]], ld[x, y]],
          and[member[u, omega], member[v, omega], member[x, omega],
           member[y, omega], subclass[natmul[v, y], natmul[u, x]]]]

Out[37]= True

In[38]:= member[natsub[natmul[u_, x_], natmul[v_, y_]], ld[x_, y_]] :=
          and[member[u, omega], member[v, omega], member[x, omega],
           member[y, omega], subclass[natmul[v, y], natmul[u, x]]]
```

---

## a reify rule for ld[x,y]

A **reify** rule for linear differences can be derived as follows.

```
In[39]:= SubstTest[reify, x, image[rotate[NATADD],
            cart[image[v, set[f[x]]], image[v, set[g[x]]]]], v → DIV] // Reverse

Out[39]= reify[x, ld[f[x], g[x]]] == composite[SECOND,
           intersection[composite[inverse[FIRST], DIV, VERTSECT[reify[x, g[x]]]],
            composite[inverse[NATADD], DIV, VERTSECT[reify[x, f[x]]]]]]

In[40]:= reify[x_, ld[y_, z_]] :=
          composite[SECOND, intersection[composite[inverse[FIRST], DIV, VERTSECT[reify[x, z]]],
            composite[inverse[NATADD], DIV, VERTSECT[reify[x, y]]]]]
```

The following elementary application can also be established using **ReifNormality**.

```
In[41]:= SubstTest[reify, x, ld[f[x], x], f[x] → x]

Out[41]= composite[SECOND,
           intersection[composite[inverse[FIRST], DIV], composite[inverse[NATADD], DIV]]] == DIV

In[42]:= composite[SECOND,
           intersection[composite[inverse[FIRST], DIV], composite[inverse[NATADD], DIV]]] := DIV
```