initial segments of wellorders

Johan G. F. Belinfante
2008 March 20

```
In[1]:= SetDirectory["l:"]; << goedel.08mar20a; << tools.m

        :Package Title: goedel.08mar20a                    2008 March 20 at 5:20 a.m.

        It is now:  2008 Mar 20 at 8:33

        Loading Simplification Rules

        TOOLS.M                                      Revised 2008 February 12

        weightlimit = 40
```

summary

Initial segments of a well-order relation **w** are proper subsets of **fix[w]** that are invariant under **inverse[w]**. All such sets are open half-intervals. This characterization of initial segments of wellorderings is derived in this notebook, inspired by the short proof of lemma 1.2 on page 104 in the following reference.

```
In[2]:= "Karel Hrbacek and Thomas Jech, Introduction to
         Set Theory, Marcel Dekker, Inc., New York, third ed., 1999.";
```

Vertical sections for well orders and their inverses are closed half-intervals, usually denoted **[a,)** and **(,a]**. Open half-intervals of total orders may be described either as the result of removing the end-points of a closed half-interval, or as relative complements of closed half-intervals. The latter is adopted here. The main result is proved only for the case that the well-order is a set. For convenience, the compound wrapper **wo[setpart[w]]** is used in place of sethood and **WELLORDER** literals.

derivation

Theorem. The condition that a set be both invariant and subvariant can be written as an equation.

```
In[3]:= SubstTest[and, subclass[x, y], subclass[y, x], y → image[wo[w], x]] // Reverse

Out[3]= and[subclass[x, fix[wo[w]]], subclass[image[wo[w], x], x]] == equal[x, image[wo[w], x]]

In[4]:= and[subclass[x_, fix[wo[w_]]], subclass[image[wo[w_], x_], x_]] :=
         equal[x, image[wo[w], x]]
```

Lemma. If **s** is an initial segment, then **fix[wo[w]]** is not a subclass of **s**.

```
In[5]:= Map[not, SubstTest[and, implies[p1, p3],
          implies[and[p2, p3], p4], not[implies[and[p1, p2], p4]],
          {p1 -> equal[s, image[inverse[wo[w]], s]], p2 → not[equal[s, fix[wo[w]]]],
           p3 → subclass[s, fix[wo[w]]], p4 → not[subclass[fix[wo[w]], s]]}]] // Reverse

Out[5]= or[equal[s, fix[wo[w]]],
          not[equal[s, image[inverse[wo[w]], s]]], not[subclass[fix[wo[w]], s]]] == True

In[6]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Theorem. The existence of least elements applied to the complement of an initial segment **s** yields an inclusion in one direction.

```
In[7]:= (SubstTest[or, not[member[x, V]], not[subclass[x, fix[wo[t]]]],
          subclass[x, image[wo[t], set[APPLY[LEAST[wo[t]], x]]]],
          x → dif[fix[wo[t]], s]] // Reverse) /. t → setpart[w]

Out[7]= subclass[fix[wo[setpart[w]]],
          union[s, image[wo[setpart[w]], set[APPLY[LEAST[wo[setpart[w]]],
              intersection[complement[s], fix[wo[setpart[w]]]]]]]]] == True

In[8]:= (% /. {w → w_, s → s_, y → y_}) /. Equal → SetDelayed
```

Lemma.

```
In[10]:= SubstTest[implies, and[member[t, s], subclass[s, z]],
           member[t, z], z → image[inverse[wo[w]], s]] // Reverse

Out[10]= or[member[t, image[inverse[wo[w]], s]],
           not[member[t, s]], not[subclass[s, fix[wo[w]]]]] == True

In[11]:= or[member[t_, image[inverse[wo[w_]], s_]],
           not[member[t_, s_]], not[subclass[s_, fix[wo[w_]]]]] := True
```

Lemma. Existence of least member.

```
In[16]:= (Map[or[#, member[APPLY[LEAST[wo[t]], intersection[complement[s], fix[wo[t]]]]],
               fix[wo[t]]]] &, (implies[member[x, domain[LEAST[wo[t]]]],
                member[APPLY[LEAST[wo[t]], x], least[wo[t], x]]] // NotNotTest) /.
             x → dif[fix[wo[t]], s]] // MapNotNot) /. t → setpart[w]

Out[16]= or[member[
            APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]],
            fix[wo[setpart[w]]]], subclass[fix[wo[setpart[w]]], s]] == True

In[17]:= (% /. {w → w_, s → s_}) /. Equal → SetDelayed
```

Lemma.

```
In[20]:= ((SubstTest[implies, and[member[x, V], not[equal[0, x]], subclass[x, fix[wo[t]]]],
            member[APPLY[LEAST[wo[t]], x], x], x → dif[fix[wo[t]], s]] //
          Reverse) /. t → setpart[w]) // MapNotNot

Out[20]= or[not[member[
           APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]],
           s]], subclass[fix[wo[setpart[w]]], s]] == True

In[21]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Lemma.

```
In[24]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3],
           not[implies[p1, p3]], {p1 → and[equal[s, image[inverse[wo[setpart[w]]], s]],
              not[equal[s, fix[wo[setpart[w]]]]]], p2 → not[subclass[fix[wo[setpart[w]]], s]],
            p3 → not[member[APPLY[LEAST[wo[setpart[w]]],
                intersection[complement[s], fix[wo[setpart[w]]]]], s]]}]] // Reverse

Out[24]= or[equal[s, fix[wo[setpart[w]]]], not[equal[s, image[inverse[wo[setpart[w]]], s]]],
           not[member[APPLY[LEAST[wo[setpart[w]]],
              intersection[complement[s], fix[wo[setpart[w]]]]], s]]] == True

In[25]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Lemma.

```
In[40]:= Map[not, SubstTest[and, implies[and[p1, p4], p5],
           not[and[p2, p3, p4, p5]], not[implies[and[p2, p3, p4], not[p1]]], {p1 → member[
              APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]],
              image[inverse[wo[setpart[w]]], s]], p2 → not[equal[s, fix[wo[setpart[w]]]]],
            p3 → equal[s, image[inverse[wo[setpart[w]]], s]],
            p4 → subclass[image[inverse[wo[setpart[w]]], s], s],
            p5 → member[APPLY[LEAST[wo[setpart[w]]],
               intersection[complement[s], fix[wo[setpart[w]]]]], s]}]] // Reverse

Out[40]= or[equal[s, fix[wo[setpart[w]]]],
           not[equal[s, image[inverse[wo[setpart[w]]], s]]], not[member[
             APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]],
             image[inverse[wo[setpart[w]]], s]]]] == True

In[41]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Lemma.

```
In[45]:= or[and[not[member[APPLY[LEAST[wo[setpart[w]]], intersection[complement[s],
              fix[wo[setpart[w]]]]], image[inverse[wo[setpart[w]]], s]]],
          subclass[s, fix[wo[setpart[w]]]]], equal[s, fix[wo[setpart[w]]]]],
         not[equal[s, image[inverse[wo[setpart[w]]], s]]]] // NotNotTest

Out[45]= or[and[not[member[
             APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]],
             image[inverse[wo[setpart[w]]], s]]],
          subclass[s, fix[wo[setpart[w]]]]], equal[s, fix[wo[setpart[w]]]]],
         not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[46]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Theorem. (combining the two directions)

```
In[48]:= SubstTest[and, implies[p, subclass[s, t]], implies[p, subclass[t, s]],
          {p → and[equal[s, image[inverse[wo[setpart[w]]], s]],
             not[equal[s, fix[wo[setpart[w]]]]]],
           t → intersection[complement[image[wo[setpart[w]], set[APPLY[LEAST[wo[setpart[w]]],
                intersection[complement[s], fix[wo[setpart[w]]]]]]]], fix[wo[setpart[w]]]]}]

Out[48]= or[equal[s, fix[wo[setpart[w]]]], equal[s,
          intersection[complement[image[wo[setpart[w]], set[APPLY[LEAST[wo[setpart[w]]],
                intersection[complement[s], fix[wo[setpart[w]]]]]]]], fix[wo[setpart[w]]]]],
         not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[49]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Lemma.

```
In[59]:= Map[not,
         SubstTest[and, implies[p2, p3], implies[and[p1, p3], p4], implies[and[p2, p4], p5],
          not[implies[and[p1, p2], p5]], {p1 → not[equal[s, fix[wo[setpart[w]]]]],
           p2 → equal[s, image[inverse[wo[setpart[w]]], s]],
           p3 → subclass[s, fix[wo[setpart[w]]]], p4 → not[subclass[fix[wo[setpart[w]]], s]],
           p5 → member[APPLY[LEAST[wo[setpart[w]]], intersection[complement[s],
                fix[wo[setpart[w]]]]], fix[wo[setpart[w]]]]}]] // Reverse

Out[59]= or[equal[s, fix[wo[setpart[w]]]], member[
          APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]],
          fix[wo[setpart[w]]]], not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[60]:= (% /. {w → w_, s → s_}) /. Equal → SetDelayed
```

Theorem. The relative complement of an initial segment **s** is a closed half-interval.

```
In[63]:= (Map[not, SubstTest[and, implies[p1, p3], implies[and[p0, p1], or[p2, p4]],
      implies[p4, p5], not[implies[and[p0, p1], or[p2, p5]]],
      {p0 → equal[x, APPLY[LEAST[wo[setpart[w]]], intersection[complement[s],
           fix[wo[setpart[w]]]]]], p1 → equal[s, image[inverse[wo[setpart[w]]], s]],
       p2 → equal[s, fix[wo[setpart[w]]]], p3 → subclass[s, fix[wo[setpart[w]]]],
       p4 → equal[s, intersection[complement[image[wo[setpart[w]], set[x]]],
          fix[wo[setpart[w]]]]], p5 → equal[image[wo[setpart[w]], set[x]],
         intersection[complement[s], fix[wo[setpart[w]]]]]}]] // Reverse) /.
    x -> APPLY[LEAST[wo[setpart[w]]], intersection[complement[s],
       fix[wo[setpart[w]]]]]

Out[63]= or[equal[s, fix[wo[setpart[w]]]], equal[image[wo[setpart[w]], set[
       APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]]]],
     intersection[complement[s], fix[wo[setpart[w]]]]],
    not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[64]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Lemma.

```
In[68]:= Map[not, SubstTest[and, implies[and[p1, not[p2]], p3], implies[p2, p3],
     not[implies[p1, p3]], {p1 → equal[s, image[inverse[wo[setpart[w]]], s]],
      p2 → equal[s, fix[wo[setpart[w]]]], p3 → equal[image[wo[setpart[w]], set[APPLY[
           LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]]]],
        intersection[complement[s], fix[wo[setpart[w]]]]]}]] // Reverse

Out[68]= or[equal[image[wo[setpart[w]], set[
       APPLY[LEAST[wo[setpart[w]]], intersection[complement[s], fix[wo[setpart[w]]]]]]],
     intersection[complement[s], fix[wo[setpart[w]]]]],
    not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[69]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Lemma.

```
In[80]:= SubstTest[implies, and[member[wo[t], V], equal[
      image[wo[t], set[APPLY[LEAST[wo[t]], intersection[complement[s], fix[wo[t]]]]]],
      intersection[complement[s], fix[wo[t]]]]],
    member[dif[fix[wo[t]], s], range[VERTSECT[wo[t]]]], t → setpart[w]] // Reverse

Out[80]= or[member[intersection[complement[s], fix[wo[setpart[w]]]],
     range[VERTSECT[wo[setpart[w]]]]],
    not[equal[image[wo[setpart[w]], set[APPLY[LEAST[wo[setpart[w]]],
        intersection[complement[s], fix[wo[setpart[w]]]]]]],
      intersection[complement[s], fix[wo[setpart[w]]]]]]] == True

In[81]:= (% /. {s → s_, w → w_}) /. Equal → SetDelayed
```

Theorem. The complement of a subset invariant under the inverse of a (small) well-order is a vertical section.

```
In[83]:= Map[not, SubstTest[and, implies[p1, p2], implies[p2, p3],
            not[implies[p1, p3]], {p1 → equal[s, image[inverse[wo[setpart[w]]], s]],
             p2 → equal[image[wo[setpart[w]], set[APPLY[LEAST[wo[setpart[w]]],
                  intersection[complement[s], fix[wo[setpart[w]]]]]]],
               intersection[complement[s], fix[wo[setpart[w]]]]],
             p3 → member[intersection[complement[s], fix[wo[setpart[w]]]],
               range[VERTSECT[wo[setpart[w]]]]]}]] // Reverse

Out[83]= or[member[intersection[complement[s], fix[wo[setpart[w]]]],
            range[VERTSECT[wo[setpart[w]]]]],
          not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[85]:= or[member[intersection[complement[s_], fix[wo[setpart[w_]]]],
            range[VERTSECT[wo[setpart[w_]]]]],
          not[equal[image[inverse[wo[setpart[w_]]], s_], s_]]] := True
```

---

eliminating the variable s

Lemma.

```
In[89]:= or[and[member[intersection[complement[s], fix[wo[setpart[w]]]],
             range[VERTSECT[wo[setpart[w]]]]], subclass[s, fix[wo[setpart[w]]]]],
          not[equal[s, image[inverse[wo[setpart[w]]], s]]]] // NotNotTest

Out[89]= or[and[member[intersection[complement[s], fix[wo[setpart[w]]]],
             range[VERTSECT[wo[setpart[w]]]]], subclass[s, fix[wo[setpart[w]]]]],
          not[equal[s, image[inverse[wo[setpart[w]]], s]]]] == True

In[90]:= (% /. {w → w_, s → s_}) /. Equal → SetDelayed
```

Theorem. Elimination of the variable **s**.

```
In[93]:= Map[equal[V, #] &,
          SubstTest[class, s, implies[and[member[t, V], member[s, u]], member[s, v]],
           {t → wo[setpart[w]], u → fix[IMAGE[inverse[wo[setpart[w]]]]],
            v → image[RC[fix[wo[setpart[w]]]], range[VERTSECT[wo[setpart[w]]]]]}]]

Out[93]= subclass[fix[IMAGE[inverse[wo[setpart[w]]]]],
          image[RC[fix[wo[setpart[w]]]], range[VERTSECT[wo[setpart[w]]]]]] == True

In[94]:= (% /. w → w_) /. Equal → SetDelayed
```

An inclusion in the reverse direction holds for total orders. This result is here specialized to the case of well orders:

```
In[96]:= SubstTest[subclass, image[RC[fix[to[setpart[x]]]], range[VERTSECT[to[setpart[x]]]]],
          fix[IMAGE[inverse[to[setpart[x]]]]], x → wo[setpart[w]]] // Reverse

Out[96]= subclass[image[RC[fix[wo[setpart[w]]]], range[VERTSECT[wo[setpart[w]]]]],
          fix[IMAGE[inverse[wo[setpart[w]]]]]] == True

In[97]:= (% /. w → w_) /. Equal → SetDelayed
```

Main theorem.

```
In[98]:= SubstTest[and, subclass[u, v], subclass[v, u],
          {u -> image[RC[fix[wo[setpart[w]]]], range[VERTSECT[wo[setpart[w]]]]],
           v -> fix[IMAGE[inverse[wo[setpart[w]]]]]}]

Out[98]= equal[fix[IMAGE[inverse[wo[setpart[w]]]]],
           image[RC[fix[wo[setpart[w]]]], range[VERTSECT[wo[setpart[w]]]]]] == True

In[99]:= fix[IMAGE[inverse[wo[setpart[w_]]]]] :=
          image[RC[fix[wo[setpart[w]]]], range[VERTSECT[wo[setpart[w]]]]]
```

Corollary. Wrapper-free restatement.

```
In[100]:=
       Map[implies[member[x, y], #] &, SubstTest[implies, equal[x, wo[setpart[t]]], equal[
           fix[IMAGE[inverse[x]]], image[RC[fix[x]], range[VERTSECT[x]]]], t → x]] // Reverse

Out[100]=
       or[equal[fix[IMAGE[inverse[x]]], image[RC[fix[x]], range[VERTSECT[x]]]],
         not[member[x, y]], not[WELLORDER[x]]] == True

In[101]:=
       or[equal[fix[IMAGE[inverse[x_]]], image[RC[fix[x_]], range[VERTSECT[x_]]]],
         not[member[x_, y_]], not[WELLORDER[x_]]] := True
```