the class TO of total orderings

Johan G. F. Belinfante
2004 February 10

In[1]:= << goedel54.10b; << tools.m

:Package Title: goedel54.10b 2004 February 10 at 10:10 p.m.
It is now: 2004 Feb 11 at 13:12
Loading Simplification Rules
TOOLS.M Revised 2004 January 3
weightlimit = 40

summary

Because the definition of the predicate TOTALORDER has been wrapped, the membership rule for the class TO of total orderings now looks like this:

In[2]:= member[x, TO]
Out[2]= and[member[x, V], TOTALORDER[x]]

Basic facts about the class TO of all total orderings are derived in this notebook, including two new formulas for this class, either one of which could serve as a suitable starting point for automated reasoning about this class using a program such as Otter.

relation of the class TO to other named classes

The class TO is contained in the class PO, which in turn is the intersection of other named classes. In particular, total orderings are reflexive:

In[3]:= subclass[TO, RFX] // AssertTest

In[4]:= subclass[TO, RFX] := True

The class of total orderings is contained in the class of partial orderings:

In[5]:= Map[subclass[#, PO] & , TO // Normality]

In[6]:= subclass[TO, PO] := True
Corollary: \( TO \) is contained in the class of antisymmetric relations.

\[
\begin{align*}
\text{In}[7]:= & \quad \text{SubstTest[\text{implies, and\{\text{subclass}[u, v], \text{subclass}[v, w], \text{subclass}[u, w],}
\{u \rightarrow \text{TO}, v \rightarrow \text{PO}, w \rightarrow \text{ANTISYM}\}]} \\
\text{Out}[7]= & \quad \text{subclass[\text{TO, ANTSYM}] := True}
\end{align*}
\]

Another corollary: the class \( TO \) is contained in the class of transitive relations.

\[
\begin{align*}
\text{In}[9]:= & \quad \text{SubstTest[\text{implies, and\{\text{subclass}[u, v], \text{subclass}[v, w], \text{subclass}[u, w],}
\{u \rightarrow \text{TO}, v \rightarrow \text{PO}, w \rightarrow \text{TRV}\}]} \\
\text{Out}[9]= & \quad \text{subclass[\text{TO, TRV}] := True}
\end{align*}
\]

**formula for U[TO]**

The following temporary abbreviation is used for the reflexive closure of the singleton of a pair:

\[
\begin{align*}
\text{In}[11]:&= \quad \text{el[x_, y_] := union[cart[\text{singleton[x]}, \text{singleton[y]}], id[\text{pairset[x, y]}]}] \\
\text{This is a total ordering:} \\
\text{In}[12]:= & \quad \text{member[el[x, y], TO]} \\
\text{Out}[12]= & \quad \text{True}
\end{align*}
\]

Since any pair belongs to a total ordering, one derives:

\[
\begin{align*}
\text{In}[13]:&= \quad \text{Map[\text{implies\{\text{member}[y, V], \# \} \&,}
\text{SubstTest[\text{implies, and\{\text{member}[u, v], \text{member}[v, w], \text{member}[u, U[w]],}
\{u \rightarrow \text{pair[x, y], v \rightarrow \text{el[x, y], w \rightarrow TO}]\}]} \\
\text{Out}[13]= & \quad \text{or[\text{member[\text{pair[x, y], U[TO]}], not[\text{member[x, V]}], not[\text{member[y, V]}]] := True}
\end{align*}
\]

Removing the variables yields a lower bound for \( U[TO] \).

\[
\begin{align*}
\text{In}[15]:&= \quad \text{Map[\text{equal}[0, \text{composite}[\text{Id, complement}[\#]] \&,}
\text{SubstTest[\text{class, \text{pair}[x, y], or[\text{member[\text{pair[x, y], z]}],}
\not[\text{member[x, V]}], not[\text{member[y, V]}]], z \rightarrow U[TO]]]} /\text{Reverse} \\
\text{Out}[15]= & \quad \text{subclass[\text{cart[V, V], U[TO}] := True}
\end{align*}
\]

That this is also an upper bound is easily established:

\[
\begin{align*}
\text{In}[17]:&= \quad \text{SubstTest[\text{implies, and\{\text{subclass}[u, v], \text{subclass}[v, w],}
\text{subclass[u, w], \{u \rightarrow \text{TO}, v \rightarrow \text{RFX, w \rightarrow P[\text{cart[V, V]}]}\}]} \\
\text{Out}[17]= & \quad \text{subclass[U[TO], cart[V, V]] := True}
\end{align*}
\]


These two inclusions can be combined into an equation:

\[
\text{In[19]:= } \text{SubstTest[and, subclass[u, v], subclass[v, u], \{u \rightarrow \text{cart}[V, V], v \rightarrow U[TO]\}]} \\
\text{Out[19]= True = equal[cart[V, V], U[TO]]}
\]

\[
\text{In[20]:= U[TO] := cart[V, V]}
\]

**other properties of the class TO**

Note that:

\[
\text{In[21]:= equal[intersection[TO, P[cart[V, V]]], TO]} \\
\text{Out[21]= True}
\]

This justifies the following rewrite rule:

\[
\text{In[22]:= intersection[TO, P[cart[V, V]]] := TO}
\]

Corollary.

\[
\text{In[23]:= ImageComp[IMAGE[\text{id[cart[V, V]]}], \text{id[P[cart[V, V]]]}, TO] // Reverse} \\
\text{Out[23]= image[IMAGE[\text{id[cart[V, V]]}], TO] = TO}
\]

\[
\text{In[24]:= image[IMAGE[\text{id[cart[V, V]]}], TO] := TO}
\]

Lemma.

\[
\text{In[25]:= implies[member[x, TO], member[inverse[x], TO]] // NotNotTest} \\
\text{Out[25]= or[and[member[domain[x], V], member[range[x], V], TOTALORDER[inverse[x]]], not[member[x, V]], not[TOTALORDER[x]]] = True}
\]

\[
\text{In[26]:= (% /. x \rightarrow x_) / . Equal \rightarrow SetDelayed}
\]

\[
\text{In[27]:= Map[equal[V, #] \&, SubstTest[class, x, implies[member[x, y], member[inverse[x], y]], y \rightarrow TO]] // Reverse} \\
\text{Out[27]= subclass[image[IMAGE[SWAP], TO], TO] = True}
\]

\[
\text{In[28]:= % / . Equal \rightarrow SetDelayed}
\]

The opposite inclusion also holds:

\[
\text{In[29]:= SubstTest[implies, subclass[u, v], subclass[image[w, u], image[w, v]], \{u \rightarrow image[IMAGE[SWAP], TO], v \rightarrow TO, w \rightarrow IMAGE[SWAP]\}]} \\
\text{Out[29]= subclass[TO, image[IMAGE[SWAP], TO]] = True}
\]

\[
\text{In[30]:= % / . Equal \rightarrow SetDelayed}
\]
These can be combined into an equation:

In[31]:= SubstTest[and, subclass[u, v], subclass[v, u], {u -> image[IMAGE[SWAP], TO], v -> TO}]

Out[31]= True == equal[TO, image[IMAGE[SWAP], TO]]

In[32]:= image[IMAGE[SWAP], TO] := TO

Corollary.

In[33]:= ImageComp[IMAGE[SWAP], id[P[cart[V, V]], TO]

Out[33]= image[INVERSE, TO] := TO

In[34]:= image[INVERSE, TO] := TO

---

two formulas for the class of total orderings

The fact that any two members of \( \text{fix}[x] \) are comparable for a total ordering \( x \) can be viewed as the statement that \( \text{fix}[x] \) is a clique in the symmetric hull of \( x \). This leads to a simple formula for \( \text{TO} \), but the derivation of it seems quite tricky.

In[35]:= composite[FIRST, id[INVERSE], inverse[CUP]] // DoubleInverse

Out[35]= composite[FIRST, id[INVERSE], inverse[CUP]] == inverse[HULL[SYM]]

In[36]:= composite[FIRST, id[INVERSE], inverse[CUP]] := inverse[HULL[SYM]]

A formula for the class of reflexive relations can be derived by using a sequestering trick:

In[37]:= (fix[composite[FIRST, id[x]], inverse[CUP], inverse[S], CART, DUP, IMAGE[inverse[DUP]]]) // Renormality) /. x -> INVERSE

Out[37]= fix[composite[inverse[HULL[SYM]], inverse[S], CART, DUP, IMAGE[inverse[DUP]]]] == RFX

In[38]:= fix[composite[inverse[HULL[SYM]], inverse[S], CART, DUP, IMAGE[inverse[DUP]]]] := RFX

A temporary normalization rule is added:

In[39]:= (TO // Normality // Reverse) /. Equal -> SetDelayed

This temporary rule yields the first of two formulas for the class of total orderings:

In[40]:= (Map[intersection[PO, #] &,
  (fix[composite[FIRST, id[funpart[x]], inverse[CUP], S, CART, DUP,
  IMAGE[inverse[DUP]]]) // Renormality) /. x -> INVERSE] // InvertFix

Out[40]= intersection[PO,
  fix[composite[inverse[IMAGE[inverse[DUP]]]], inverse[E], CLIQUES, HULL[SYM]]] == TO

In[41]:= intersection[PO,
  fix[composite[inverse[IMAGE[inverse[DUP]]]], inverse[E], CLIQUES, HULL[SYM]]] := TO

Lemma. A temporary rewrite rule.
In[42]:= fix[composite[reverse[x], S, CART, DUP, y]] // InvertFixTest

Out[42]= fix[composite[reverse[x], S, CART, DUP, y]]

In[43]:= fix[composite[reverse[x_, S, CART, DUP, y_]] :=
    fix[composite[reverse[y], S, CART, DUP, y]]]

Lemma.

In[44]:= SubstTest[intersection, fix[composite[reverse[funpart[x]], S, funpart[y]]],
    
    fix[composite[reverse[funpart[x]], S, funpart[y]]],
    
    {x -> composite[CUP, id[Reversal], reverse[First]],
    
    y -> composite[CART, DUP, IMAGE[reverse[DUP]]]}]

Out[44]= intersection[RFX,
    
    fix[composite[reverse[IMAGE[reverse[DUP]]], S, CART, DUP, IMAGE[reverse[DUP]]]]
]

In[45]:= intersection[RFX,
    
    fix[composite[reverse[IMAGE[reverse[DUP]]], S, CART, DUP, IMAGE[reverse[DUP]]]] :=
    
    fix[composite[reverse[IMAGE[reverse[DUP]]], S, CART, DUP, IMAGE[reverse[DUP]]]]]

Lemma.

In[46]:= equal[intersection[RFX, TO], TO]

Out[46]= True

In[47]:= intersection[RFX, TO] := TO

A second formula for the class of total orderings:

In[48]:= AssInt[PO, RFX,
    
    fix[composite[reverse[IMAGE[reverse[DUP]]], S, CART, DUP, IMAGE[reverse[DUP]]]]
]

Out[48]= intersection[PO,
    
    fix[composite[reverse[IMAGE[reverse[DUP]]], S, CART, DUP, IMAGE[reverse[DUP]]]]] := TO

In[49]:= intersection[PO,
    
    fix[composite[reverse[IMAGE[reverse[DUP]]], S, CART, DUP, IMAGE[reverse[DUP]]]]] := TO