(finchar[x] & 0 \in x) \Rightarrow \text{Uchains}[x] = x

Johan G. F. Belinfante
2008 January 4

**summary**

Any class of finite character which holds the empty set is closed under unions of chains. A corollary of this is Tukey's lemma: if the axiom of choice holds, then any class of finite character which holds the empty set has a maximal element.

**normalization of the predicate finchar**

In the course of the derivation several variables are introduced and later eliminated using class rules. Each time this happens, the negation of the predicate finchar needs to be restored. This normalization is easily accomplished using a double negation, but to save time it helps to have a rewrite rule that automatically takes care of this once and for all.
This reduction of two literals to one is not a minor matter. An important contributor to execution time is the proliferation of possible equality substitutions as the number of literals increases; combinatorial explosion due is always lurking just around the corner. For the same reason, the equality flag will be repeatedly cleared and reset as needed to control this problem.

**strategy of the derivation**

We want to prove that if \texttt{finchar[y]} and \texttt{member[0,y]}, then \texttt{member[x, intersection[chains[S], P[y]]]} implies \texttt{member[-U[x], y]}. Because \texttt{y} is of finite character, it suffices to show that every finite subset of \texttt{U[x]} belongs to \texttt{y}. We consider some finite subset \texttt{t} of \texttt{U[x]} and attempt to show that \texttt{t} belongs to \texttt{y}. Since \texttt{t} is a finite set covered by \texttt{x}, for each member of \texttt{t} one can select a member of \texttt{x} that holds that member of \texttt{t}. The set of these selected members is a finite subset of \texttt{x} whose union contains \texttt{t}. For the formal proof one needs a theorem of finite choice; for this purpose, the following formulation of finite choice will be used:

\begin{verbatim}
In[6]:= implies[and[member[t, FINITE], subclass[t, U[x]]], not[disjoint[P[E], map[t, x]]]]
\end{verbatim}

Another variable \texttt{z} will be used for this selection function; \texttt{z} is a member of the intersection of \texttt{P[E]} and \texttt{map[t,x]}. After reasoning with all these variables, all of them will be eliminated, one by one.

**temporary simplification rules**

To avoid introducing yet another variable for \texttt{range[z]}, some temporary rewrite rules are derived to undo the effects of a rewrite rule affects the range, namely this one:

\begin{verbatim}
In[7]:= equal[0, range[z]]
Out[7]= equal[0, domain[z]]
\end{verbatim}

Temporary lemma. (A finite nonempty chain has a greatest element.)

\begin{verbatim}
In[8]:= SubstTest[implies, and[member[t, FINITE], member[t, chains[S]]],
  or[empty[t], member[U[t], t]], t \rightarrow range[z]] // Reverse
Out[8]= or[equal[0, domain[z]], member[U[range[z]], range[z]], not[member[range[z], FINITE]],
  not[subclass|cart[range[z], range[z]], union[S, inverse[S]]]] = True
\end{verbatim}

\begin{verbatim}
In[9]:= (% /. z \rightarrow z_+) /. Equal \rightarrow SetDelayed
\end{verbatim}

Temporary lemma.

\begin{verbatim}
In[10]:= SubstTest[implies, and[subclass[t, U[s]], empty[s]], empty[t], s \rightarrow range[z]] // Reverse
Out[10]= or[equal[0, t], not[equal[0, domain[z]]], not[subclass[t, U[range[z]]]]] = True
\end{verbatim}

\begin{verbatim}
In[11]:= (% /. {t \rightarrow t_-, z \rightarrow z_-}) /. Equal \rightarrow SetDelayed
\end{verbatim}
a long series of lemmas

The original layout of the proof of the main theorem involved about twenty statements, far too many to be handled all at once by the GOEDEL program. For this reason the derivation was broken down into a series of lemmas, each of which eliminates one of these statements. The original labels for these statements are retained here, and for each lemma, the statement being eliminated is identified.

Lemma. (eliminating p14)

```
In[12]:= Map[not, SubstTest[and, implies[and[p4a, p4b], p7], implies[and[p13, p10], p14],
           implies[and[p7, p14], p15], not[implies[and[p4a, p4b, p10, p13], p15]],
           {p4a -> member[z, map[t, x]], p4b -> subclass[z, E], p7 -> subclass[t, U[range[z]]],
           p10 -> member[U[range[z]], range[z]], p13 -> subclass[range[z], y],
           p14 -> member[U[range[z]], y], p15 -> member[t, image[inverse[S], y]]}]] // Reverse
Out[12]= or[member[t, image[inverse[S], y]],
       not[member[z, map[t, x]]], not[member[U[range[z]], range[z]]],
       notsubclass[z, E]], notsubclass[range[z], y]] = True
```

In[13]:= (% /. {t -> t_, x -> x_, y -> y_, z -> z_}) /. Equal -> SetDelayed

Lemma. (eliminating p7 and p12) This lemma just requires a double negation.

```
In[14]:= or[member[t, y], not[equal[0, domain[z]]], not[member[0, y]],
       not[member[z, map[t, x]]], notsubclass[z, E]] // NotNotTest
Out[14]= or[member[t, y], not[equal[0, domain[z]]],
       not[member[0, y]], not[member[z, map[t, x]]], notsubclass[z, E]] = True
```

In[15]:= (% /. {t -> t_, x -> x_, y -> y_, z -> z_}) /. Equal -> SetDelayed

Lemma. (eliminating one of two uses of p8.)

```
In[16]:= Map[not, SubstTest[and, implies[p4b, p8],
           implies[and[p1c, p8], p9], not[implies[and[p1c, p4b], p9]],
           {p4b -> subclass[cart[x, x], union[S, inverse[S]]], p4b -> member[z, map[t, x]],
           p8 -> subclass[range[z], x], p9 -> subclass[range[z], chains[S]]}]] // Reverse
Out[16]= or[notmember[z, map[t, x]], notsubclass[cart[x, x], union[S, inverse[S]]],
         subclass[cart[range[z], range[z]], union[S, inverse[S]]] = True
```

In[17]:= (% /. {t -> t_, x -> x_, z -> z_}) /. Equal -> SetDelayed

Lemma. (eliminating a second use of p8.)

```
In[18]:= Map[not, SubstTest[and, implies[p4b, p8], implies[and[p1b, p8], p13],
           not[implies[and[p1b, p4b, p13]], {p1b -> subclass[x, y], p4b -> member[z, map[t, x]],
           p8 -> subclass[range[z], x], p13 -> subclass[range[z], y]]}]] // Reverse
Out[18]= or[notmember[z, map[t, x]], subclass[x, y], subclass[range[z], y]] = True
```
Lemma. (eliminating \texttt{p6} and \texttt{p9}.) This lemma only takes a fraction of a minute, but it seems like a long wait.

\begin{verbatim}
In[20]:= Map[not, SubstTest[and, implies[and[p3a, p4b], p6], implies[and[p1b, p4b], p9], implies[and[p6, p9], or[p10, p11]], not[implies[and[p1b, p3a, p4b], or[p10, p11]]], {p1b \rightarrow \text{subclass}[\text{cart}[x, x], \text{union}[S, \text{inverse}[S]]], p3a \rightarrow \text{member}[t, \text{FINITE}], p4b \rightarrow \text{member}[z, \text{map}[t, x]], p6 \rightarrow \text{member}[\text{range}[x], \text{FINITE}], p9 \rightarrow \text{subclass}[P[\text{range}[z]], \text{chains}[S]], p10 \rightarrow \text{member}[U[\text{range}[z]], \text{range}[z]], p11 \rightarrow \text{empty}[\text{range}[z]]}] // Reverse

Out[20]= or[\text{equal}[0, \text{domain}[z]], \text{member}[U[\text{range}[z]], \text{range}[z]], \text{not}[\text{member}[t, \text{FINITE}]], \text{not}[\text{member}[z, \text{map}[t, x]]], \text{not}[\text{subclass}[\text{cart}[x, x], \text{union}[S, \text{inverse}[S]]]]] = \text{True}
\end{verbatim}

\begin{verbatim}
In[21]:= (\% \rightarrow \text{SetDelayed})
\end{verbatim}

Lemma. (eliminating \texttt{p13})

\begin{verbatim}
In[22]:= Map[not, SubstTest[and, implies[and[pla, p4b], p13], implies[and[p4a, p4b, p10, p13], p15], not[implies[and[pla, p4a, p4b, p10, p15]]], {pla \rightarrow \text{subclass}[x, y], p4a \rightarrow \text{subclass}[z, E], p4b \rightarrow \text{member}[z, \text{map}[t, x]], p10 \rightarrow \text{member}[U[\text{range}[z]], \text{range}[z]], p11 \rightarrow \text{empty}[\text{range}[z]], p13 \rightarrow \text{subclass}[\text{range}[z], y], p15 \rightarrow \text{member}[t, \text{image}[\text{inverse}[S], y]]}] // Reverse

Out[22]= or[\text{member}[t, \text{image}[\text{inverse}[S], y]], \text{not}[\text{member}[z, \text{map}[t, x]]], \text{not}[\text{member}[U[\text{range}[z]], \text{range}[z]]], \text{not}[\text{subclass}[x, y]], \text{not}[\text{subclass}[z, E]]] = \text{True}
\end{verbatim}

\begin{verbatim}
In[23]:= (\% \rightarrow \text{SetDelayed})
\end{verbatim}

Lemma. (eliminating one use of \texttt{p15})

\begin{verbatim}
In[24]:= Map[not, SubstTest[and, implies[and[pla, p4a, p4b, p10], p15], implies[and[p15, p16, p17], not[implies[and[pla, p4a, p4b, p10, p16, p17]]], {pla \rightarrow \text{subclass}[x, y], p4a \rightarrow \text{subclass}[z, E], p4b \rightarrow \text{member}[z, \text{map}[t, x]], p10 \rightarrow \text{member}[U[\text{range}[z]], \text{range}[z]], p15 \rightarrow \text{member}[t, \text{image}[\text{inverse}[S], y]], p16 \rightarrow \text{equal}[\text{image}[\text{inverse}[S], y], y], p17 \rightarrow \text{member}[t, y]]}] // Reverse

Out[24]= or[\text{member}[t, y], \text{not}[\text{equal}[y, \text{image}[\text{inverse}[S], y]]], \text{not}[\text{member}[z, \text{map}[t, x]]], \text{not}[\text{member}[U[\text{range}[z]], \text{range}[z]]], \text{not}[\text{subclass}[x, y]], \text{not}[\text{subclass}[z, E]]] = \text{True}
\end{verbatim}

\begin{verbatim}
In[25]:= (\% \rightarrow \text{SetDelayed})
\end{verbatim}

Lemma. (Eliminating a second use of \texttt{p15})
Reverse

Lemma. (classes of finite character are hereditary.)

using wrappers to reduce the number of literals

The next step, to eliminate the variable z, was delayed for some time because there were too many literals. This problem was eventually resolved by using wrappers to reduce the number of literals. Three wrappers are introduced at this point: fin, setpart and spine.
Before eliminating \( z \), some flags are cleared.

\[
\text{In[32]} := \quad \text{simplify} = \text{False}; \ \text{cond} = \text{False}; \ \text{equality} = \text{False};
\]

The elimination of \( z \) does not take long now.

\[
\text{In[33]} := \quad \text{Map}[^\text{equal}[V, \#] \&, \text{SubstTest}[^\text{class}, z, \{\text{implies}[\text{and}[^\text{equal}[z, s], \text{member}[0, y], \text{subclass}[w, y], \text{member}[z, u]], \text{member}[v, y]], \{z \rightarrow \text{complement}[y], s \rightarrow \text{image}[S, \text{intersection}[\text{FINITE}, \text{complement}[y]]], u \rightarrow \text{intersection}[\text{P}[E], \text{map}[\text{fin}[t], \text{spine}[S, \text{setpart}[x]]]], v \rightarrow \text{fin}[t], w \rightarrow \text{spine}[S, \text{setpart}[x]]\}]])
\]

\[
\text{Out[33]} := \quad \text{or}[\text{equal}[0, \text{intersection}[\text{map}[\text{fin}[t], \text{spine}[S, \text{setpart}[x]]], \text{P}[E]]], \text{member}[\text{fin}[t], y], \text{not}[\text{equal}][\text{complement}[y], \text{image}[S, \text{intersection}[\text{FINITE}, \text{complement}[y]]]], \text{not}[\text{member}[0, y]], \text{not}[\text{subclass}[\text{spine}[S, \text{setpart}[x]], y]] = \text{True}
\]

\[
\text{In[34]} := \quad \% . \{x \rightarrow x_-, y \rightarrow y_-, t \rightarrow t_-\} / . \text{Equal} \rightarrow \text{SetDelayed}
\]

One needs to restore equality flag because it is needed for wrapper removal. The simplify and cond flags remain cleared at this point.

\[
\text{In[35]} := \quad \text{equality} = \text{True};
\]

Remove the compound spine[S, setpart[x]] wrapper.

\[
\text{In[36]} := \quad \text{SubstTest}[^\text{implies}, \text{equal}[x, \text{spine}[S, \text{setpart}[u]]], \{\text{or}[\text{equal}[0, \text{intersection}[\text{map}[\text{fin}[t], x], \text{P}[E]]], \text{member}[\text{fin}[t], y], \text{not}[\text{finchar}[y]], \text{not}[\text{member}[0, y]], \text{not}[\text{subclass}[x, y]]], u \rightarrow x\} // \text{Reverse}
\]

\[
\text{Out[36]} := \quad \text{or}[\text{equal}[0, \text{intersection}[\text{map}[\text{fin}[t], x], \text{P}[E]]], \text{member}[\text{fin}[t], y], \text{not}[\text{equal}][\text{complement}[y], \text{image}[S, \text{intersection}[\text{FINITE}, \text{complement}[y]]]], \text{not}[\text{member}[0, y]], \text{not}[\text{member}[x, V]], \text{not}[\text{subclass}[x, y]], \text{not}[\text{subclass}[\text{cart}[x, x], \text{union}[S, \text{inverse}[S]]]] = \text{True}
\]

\[
\text{In[37]} := \quad \% . \{x \rightarrow x_-, t \rightarrow t_-, y \rightarrow y_-\} / . \text{Equal} \rightarrow \text{SetDelayed}
\]

Remove the fin wrappers.

\[
\text{In[38]} := \quad \text{SubstTest}[^\text{implies}, \text{equal}[t, \text{fin}[v]], \{\text{or}[\text{equal}[0, \text{intersection}[\text{map}[t, x], \text{P}[E]]], \text{member}[t, y], \text{not}[\text{finchar}[y]], \text{not}[\text{member}[0, y]], \text{not}[\text{member}[x, \text{chains}[S]]], \text{not}[\text{subclass}[x, y]]], v \rightarrow t\} // \text{Reverse}
\]

\[
\text{Out[38]} := \quad \text{or}[\text{equal}[0, \text{intersection}[\text{map}[t, x], \text{P}[E]]], \text{member}[t, y], \text{not}[\text{equal}][\text{complement}[y], \text{image}[S, \text{intersection}[\text{FINITE}, \text{complement}[y]]]], \text{not}[\text{member}[0, y]], \text{not}[\text{member}[t, \text{FINITE}]], \text{not}[\text{member}[x, V]], \text{not}[\text{subclass}[x, y]], \text{not}[\text{subclass}[\text{cart}[x, x], \text{union}[S, \text{inverse}[S]]]] = \text{True}
\]

\[
\text{In[39]} := \quad \% . \{x \rightarrow x_-, t \rightarrow t_-, y \rightarrow y_-\} / . \text{Equal} \rightarrow \text{SetDelayed}
\]

At this point one can eliminate all mention of \( \text{P}[E] \):
Lemma. (A set belongs to a class of finite character iff all its finite subsets are members.)

Before eliminating the variable $t$ the equality flag is cleared.

Lemma. (A set belongs to a class of finite character iff all its finite subsets are members.)

Theorem.
Now the variable \( x \) needs to be removed.

\[
\text{In [49]} := \text{Map}[	ext{equal}[V, #] \&, \text{SubstTest}[\text{class, } x, \\
\quad \text{implies}[\text{and}[	ext{finchar}[y], \text{member}[0, y], \text{member}[x, t]], \text{member}[U[x], y]], \\
\quad t \rightarrow \text{intersection}[\text{chains}[S], P[y]]]]
\]

\[
\text{Out [49]} := \text{or}[\text{not}[\text{equal}[\text{complement}[y], \text{image}[S, \text{intersection}[\text{FINITE, complement}[y]]]]], \\
\quad \text{not}[\text{member}[0, y]], \text{subclass}[\text{Uchains}[y], y]] = \text{True}
\]

\[
\text{In [50]} := \text{(}/. y \rightarrow y_/)/. \text{Equal} \rightarrow \text{SetDelayed}
\]

This can be cleaned up using this lemma:

\[
\text{In [51]} := \text{or}[\text{equal}[y, \text{Uchains}[y]], \text{not}[\text{subclass}[\text{Uchains}[y], y]]] // \text{AssertTest}
\]

\[
\text{Out [51]} := \text{or}[\text{equal}[y, \text{Uchains}[y]], \text{not}[\text{subclass}[\text{Uchains}[y], y]]] = \text{True}
\]

\[
\text{In [52]} := \text{(}/. y \rightarrow y_/)/. \text{Equal} \rightarrow \text{SetDelayed}
\]

Main Theorem. A class of finite character which holds the empty set is closed under unions of chains.

\[
\text{In [53]} := \text{Map}[\text{not}, \text{SubstTest}[\text{and}, \text{implies}[\text{and}[p0, p1], p4], \text{implies}[p4, p5], \\
\quad \text{not}[\text{implies}[\text{and}[p0, p1], p5]], \{p0 \rightarrow \text{member}[0, x], p1 \rightarrow \text{finchar}[x], \\
\quad p4 \rightarrow \text{subclass}[\text{Uchains}[x], x], p5 \rightarrow \text{equal}[\text{Uchains}[x], x]]]] // \text{Reverse}
\]

\[
\text{Out [53]} := \text{or}[\text{equal}[x, \text{Uchains}[x]], \\
\quad \text{not}[\text{equal}[\text{complement}[x]], \text{image}[S, \text{intersection}[\text{FINITE, complement}[x]]]]], \\
\quad \text{not}[\text{member}[0, x]]] = \text{True}
\]

\[
\text{In [54]} := \text{or}[\text{equal}[x_, \text{Uchains}[x_]], \\
\quad \text{not}[\text{equal}[\text{complement}[x_], \text{image}[S, \text{intersection}[\text{FINITE, complement}[x_]]]]], \\
\quad \text{not}[\text{member}[0, x_]]] := \text{True}
\]

Corollary. A completely variable-free statement is possible. This weaker statement only captures the case of sets of finite character, not proper classes.

\[
\text{In [55]} := \text{Map}[\text{equal}[V, #] \&, \\
\quad \text{dif}[\text{intersection}[\text{FINCHAR, image[E, set[0]]}], \text{fix}[\text{UCHAINS}]] // \text{complement} // \\
\quad \text{Renormality}
\]

\[
\text{Out [55]} := \text{subclass}[\text{FINCHAR, union[fix[UCHAINS], P[complement[set[0]]]]}] = \text{True}
\]

\[
\text{In [56]} := \text{subclass}[\text{FINCHAR, union[fix[UCHAINS], P[complement[set[0]]]]}] := \text{True}
\]

---

**Tukey's lemma**

As a corollary of Zorn's lemma, one can derive a famous result due to Tukey.

\[
\text{In [57]} := "\text{J. W. Tukey, Convergence and Uniformity} \\
\quad \text{in Topology, Ann. Math. Studies, No. 2, Princeton, 1940."};
\]
Tukey's lemma.

The converse is also true:

These results can be combined into a single rewrite rule:

Comment. Jean Rubin explicitly also adds an analog of the hypothesis \( \text{member}[0, x] \) to her version of Tukey's lemma. Her version of Tukey's lemma is presented as a theorem-schema with a 'variable predicate' in place of a free class variable.

One cannot omit the hypothesis \( \text{member}[0, x] \) in the main theorem. In general, a class of finite character need not be closed under unions of chains. This is simply because the union of the empty chain need not belong to a class of finite character.

Comment. Jean Rubin explicitly also adds an analog of the hypothesis \( \text{member}[0, x] \) to her version of Tukey's lemma. Her version of Tukey's lemma is presented as a theorem-schema with a 'variable predicate' in place of a free class variable.