well-founded recursion, part 6.

Johan G. F. Belinfante
revised 2004 September 15

summary

In this sixth notebook on well-founded recursion, a uniqueness theorem is derived, and the result is applied to establish a connection between \texttt{rec}[x,y] and the function \texttt{RANK}.

a fact about history

The concepts of \texttt{HISTORY} and \texttt{history} are in fact equal, but deliberately, fewer rewrite rules have been made available for \texttt{history}. It is easy, however, to transfer facts about \texttt{HISTORY} to obtain corresponding facts about \texttt{history} whenever the need arises. To illustrate this, consider the following fact which is known to hold for the function \texttt{HISTORY}[\texttt{funpart}[x],\texttt{thinpart}[y]].

\begin{verbatim}
In[2]:= implies[equal[cart[V, V], domain[x]],
                equal[V, domain[composite[x, HISTORY[funpart[w], thinpart[y]]]]]]

\end{verbatim}

The transfer from \texttt{HISTORY} to \texttt{history} can be done using equality substitution
In [3]:\[
\text{SubstTest}[\text{implies, and[equal[cart[V, V], domain[x]], equal[t, composite[x, HISTORY[funpart[w], thinpart[y]]]], equal[V, domain[t]], t \to composite[x, history[funpart[w], thinpart[y]]]]}
\]

Out[3]: or[equal[V, image[inv \text{history[funpart[w], thinpart[y]]}, domain[x]]], not[equal[cart[V, V], domain[x]]]] = True

In [4]: (% /. \{w \to \_, x \to x\_, y \to y\_\}) /. \text{Equal} \to \text{SetDelayed}

---

temporary abbreviations

The following two temporary predicates are introduced here to provide a convenient language to describe various results that will be derived in this notebook. Note that partial solutions are not required to be sets. In the definition of a total solution, no condition is placed on the domain.

In [5]:\[
\text{partialsolution}[\_, x\_, y\_] := \\
\text{and}[\text{invariant}[y, \text{domain}[w]], \text{subclass}[w, \text{composite}[x, \text{history}[w, y]]]]
\]

In [6]:\[
\text{totalsolution}[\_, x\_, y\_] := \text{equal}[w, \text{composite}[x, \text{history}[w, y]]]
\]

If \(x\) is a function, any partial or total solution is also a function:

In [7]:\[
\text{implies[} \\
\text{and[FUNCTION}[x], \text{subclass}[w, \text{composite}[x, \text{history}[w, y]]]], \text{FUNCTION}[w]]
\]

Out[7]: True

In [8]:\[
\text{implies[and[FUNCTION}[x], \text{equal}[w, \text{composite}[x, \text{history}[w, y]]]], \text{FUNCTION}[w]]
\]

Out[8]: True

Recall that the class \text{partrec}[x,y] consists of partial solutions that are sets, and \text{rec}[x,y] is defined to be the union of this class.

In [9]:\[
\text{equiv[member}[w, \text{partrec}[x, y]], \\
\text{and[member}[w, V], \text{partialsolution}[w, x, y]]] // \text{not} // \text{not}
\]

Out[9]: True

In [10]:\[
\text{U[partrec}[x, y]
\]

Out[10]: \text{rec}[x, y]
on the totality of total solutions

A total function is a function whose domain is $V$. A total binary function is one whose domain is $\text{cart}[V, V]$. Under suitable hypotheses, total solutions are also total functions.

```
In[11]:= SubstTest[implies, and[equal[r, funpart[w]],
  equal[s, thinpart[y]], equal[t, composite[x, HISTORY[r, s]]],
  equal[cart[V, V], domain[x][], equal[V, domain[t]],
  {r -> w, s -> composite[Id, y], t -> composite[x, history[w, y]]}]]
Out[11]= or[equal[V, image[inverse[history[w, y]], domain[x][], not[equal[V, domain[VERTSECT[y]]]],
  not[equal[cart[V, V], domain[x][], not[FUNCTION[w]]] = True

In[12]:= (% /. (w -> w_, x -> x_, y -> y_)) / Equal -> SetDelayed
```

The condition $w = t$ for $w$ to be a total solution requires special attention:

```
In[13]:= SubstTest[implies, and[FUNCTION[w], equal[cart[V, V], domain[x][], thin[y],
  equal[t, composite[x, history[w, y]]], equal[V, domain[t]], t -> w]]
Out[13]= or[equal[V, domain[w][], not[equal[V, domain[VERTSECT[y]]]],
  not[equal[w, composite[x, history[w, y]]],
  not[equal[cart[V, V], domain[x][], not[FUNCTION[w]]] = True

In[14]:= (% /. (w -> w_, x -> x_, y -> y_)) / Equal -> SetDelayed
```

In practice, it is convenient to replace the condition that $w$ be a function by the condition that $x$ be a function.

```
In[15]:= Map[not, SubstTest[and, implies[and[p1, p2], p3],
  implies[and[p1, p2, p3], p4], not[implies[and[p1, p2, p4]],
  {p1 -> and[FUNCTION[x], equal[cart[V, V], domain[x][], thin[y]],
  p2 -> equal[w, composite[x, history[w, y]]],
  p3 -> FUNCTION[w], p4 -> equal[V, domain[w]]}]]
Out[15]= or[equal[V, domain[w][], not[equal[V, domain[VERTSECT[y]]]],
  not[equal[w, composite[x, history[w, y]]]],
  not[equal[cart[V, V], domain[x][], not[FUNCTION[x]]] = True

In[16]:= or[equal[V, domain[w_]], not[equal[V, domain[VERTSECT[y_]]]],
  not[equal[w_, composite[x_, history[w_, y_]]]],
  not[equal[cart[V, V], domain[x_]], not[FUNCTION[x_]]] := True
Restatement: if $x$ is a total binary function and $y$ is thin, then any total solution $w$ is a total function.

**total solutions are partial solutions**

It is an easy corollary of the result derived in the preceding section that under the same hypotheses, the domain of a total solution $w$ is invariant under $y$.

It follows that if $x$ is a total binary function, and if $y$ is thin, then any total solution is also a partial solution.
a condition for \( \text{rec}[x,y] \) to be a total solution

In this section it is shown that if \( x \) is a total binary function, and if \( y \) is a thin relation whose inverse is well-founded, then \( \text{rec}[x,y] \) is a total solution. It is already known that under these conditions \( \text{rec}[x,y] \) is a partial solution, and that it is a total function. To complete the proof, the idea is to show that \( \text{composite}[x, \text{history}[\text{rec}[x,y],y]] \) is also a function, and therefore \( \text{rec}[x,y] \) is a restriction of it. Since \( \text{rec}[x,y] \) is a total function, this restriction reduces to an equality.

\[
\begin{align*}
\text{In}[22]:= & \quad \text{Map}[\text{not, SubstTest}[\text{and, implies}[p1, p2],}
\text{implies}[p1, p3], \text{implies}[\text{and}[p2, p3], p4], \text{implies}[p1, p5],
\text{implies}[\text{and}[p4, p5], p6], \text{not}[\text{implies}[p1, p6]], (p1 \rightarrow \text{and}[\text{FUNCTION}[x],
\text{equal}[\text{cart}[V, V], \text{domain}[x]], \text{thin}[y], \text{WELLFOUNDED}[\text{inverse}[y]]),
\text{p2} \rightarrow \text{subclass}[\text{rec}[x, y], \text{composite}[x, \text{history}[\text{rec}[x, y], y]]],
\text{p3} \rightarrow \text{FUNCTION}[\text{composite}[x, \text{history}[\text{rec}[x, y], y]]],
\text{p4} \rightarrow \text{equal}[\text{rec}[x, y], \text{composite}[x, \text{history}[\text{rec}[x, y], y]],
\text{id}[\text{domain}[\text{rec}[x, y]]],] , p5 \rightarrow \text{equal}[\text{domain}[\text{rec}[x, y]], V],
\text{p6} \rightarrow \text{equal}[\text{rec}[x, y], \text{composite}[x, \text{history}[\text{rec}[x, y], y]]])]
\text{Out}[22]= & \quad \text{or}[\text{equal}[\text{composite}[x, \text{history}[\text{rec}[x, y], y]], \text{rec}[x, y]],
\text{not}[\text{equal}[V, \text{domain}[\text{VERTSECT}[y]]]], \text{not}[\text{equal}[\text{cart}[V, V], \text{domain}[x]]],
\text{not}[\text{FUNCTION}[x]], \text{not}[\text{WELLFOUNDED}[\text{inverse}[y]]] ] = \text{True}
\text{In}[23]:= & \quad \text{or}[\text{equal}[\text{composite}[x, \text{history}[\text{rec}[x, y], y]], \text{rec}[x, y]],
\text{not}[\text{equal}[V, \text{domain}[\text{VERTSECT}[y]]]], \text{not}[\text{equal}[\text{cart}[V, V], \text{domain}[x]]],
\text{not}[\text{FUNCTION}[x]], \text{not}[\text{WELLFOUNDED}[\text{inverse}[y]]] ] = \text{True}
\text{Restatement:}
\text{In}[24]:= & \quad \text{implies}[\text{and}[\text{FUNCTION}[x], \text{equal}[\text{cart}[V, V], \text{domain}[x]],
\text{thin}[y], \text{WELLFOUNDED}[\text{inverse}[y]]], \text{totalsolution}[\text{rec}[x, y], x, y]]
\text{Out}[24]= \quad \text{True}
\]

compatibility of partial solutions

Functions are \textbf{compatible} if their union is a function. Recall the following basic theorem about compatibility of partial solutions:
In [25]: = implies[
    and[WELLFOUNDED[inverse[y]], partialsolution[u, funpart[x], thinpart[y]],
    partialsolution[v, funpart[x], thinpart[y]], FUNCTION[union[u, v]]]

Out[25]= True

The **funpart** and **thinpart** wrappers can be removed:

In [26]: = SubstTest[implies,
    and[equal[s, thinpart[y]], equal[t, funpart[x]], WELLFOUNDED[inverse[y]],
    partialsolution[u, t, s], partialsolution[v, t, s]],
    FUNCTION[union[u, v]], {s -> composite[Id, y], t -> x}]

Out[26]= or[FUNCTION[union[u, v]], not[equal[V, domain[VERTSECT[y]]]]],
    not[FUNCTION[x]], not[subclass[u, composite[x, history[u, y]]]],
    not[subclass[v, composite[x, history[v, y]]]],
    not[subclass[composite[y, domain[u]], domain[u]]],
    not[subclass[composite[y, domain[v]], domain[v]]],
    not[WELLFOUNDED[inverse[y]]] = True

In [27]: = (% /: {u -> u_, v -> v_, x -> x_, y -> y_}) /. Equal -> SetDelayed

Restatement of the compatibility theorem.

In [28]: = implies[and[FUNCTION[x], thin[y], WELLFOUNDED[inverse[y]],
    partialsolution[u, x, y], partialsolution[v, x, y]], FUNCTION[union[u, v]]]

Out[28]= True

a condition for partial solutions to be contained in rec[x,y]

In this section it is shown that under suitable hypotheses, any partial solution is con-
tained in **rec[x,y]**. When **w** is a set, this is obvious, because **rec[x,y]** is the union of
all small partial solutions:

In [29]: = implies[member[w, partrec[x, y]], subclass[w, rec[x, y]]]

Out[29]= True

The compatibility theorem for partial solutions can be applied to the case that one of the
partial solutions is **rec[x,y]**:
This statement can be simplified by using the fact that \( \text{rec}[x,y] \) is a partial solution under the assumed hypotheses:

One can now deduce that if \( x \) is a total binary function, and if \( y \) is a thin relation whose inverse is well-founded, then any partial solution \( w \) is contained in \( \text{rec}[x,y] \). Here it is not assumed that \( w \) is a set.
Restatement:

\[
\text{implies} \left[ \text{and}[\text{FUNCTION}[x]], \text{equal}[\text{cart}[V, V], \text{domain}[x]], \text{thin}[y], \text{WELLFOUNDED}[\text{inverse}[y]], \text{partialsolution}[w, x, y], \text{subclass}[w, \text{rec}[x, y]] \right]
\]

\text{Out}[36]= \text{True}

uniqueness theorem

It is an easy corollary of the result derived in the preceding section that, under the same hypotheses, any total solution is contained in \text{rec}[x,y].

\[
\text{Map}[\text{not}, \text{SubstTest}[\text{and}, \text{implies}[\text{and}[p1, p2], p3]], \text{implies}[\text{and}[p1, p3, p4], \text{not}[\text{implies}[\text{and}[p1, p2], p4]]],
\{p1 \rightarrow \text{and}[\text{FUNCTION}[x]], \text{equal}[\text{cart}[V, V], \text{domain}[x]], \text{thin}[y], \text{WELLFOUNDED}[\text{inverse}[y]]\}, p2 \rightarrow \text{totalsolution}[w, x, y],
p3 \rightarrow \text{partialsolution}[w, x, y], p4 \rightarrow \text{subclass}[w, \text{rec}[x, y]]]
\]

\text{Out}[37]= \text{or}[\text{not}[\text{equal}[V, \text{domain}[\text{VERTSECT}[y]]]],
\text{not}[\text{equal}[\text{cart}[V, V], \text{domain}[x]]], \text{not}[\text{FUNCTION}[x]],
\text{WELLFOUNDED}[\text{inverse}[y]], \text{subclass}[w, \text{rec}[x, y]]] = \text{True}

\text{In}[38]= (\% /. \{w \rightarrow \_., x \rightarrow \_., y \rightarrow \_\}) /. \text{Equal} \rightarrow \text{SetDelayed}

This result can be sharpened to obtain the following uniqueness theorem for \text{rec}[x,y]:

\[
\text{Map}[\text{not}, \text{SubstTest}[\text{and}, \text{implies}[\text{and}[p1, p2], p3]], \text{implies}[\text{and}[p1, p2], p4],
\text{implies}[\text{and}[p1, p4], p5], \text{implies}[\text{and}[p1, p2], p6], \text{implies}[\text{and}[p3, p5], p7],
\text{implies}[\text{and}[p6, p7], p8], \text{not}[\text{implies}[\text{and}[p1, p2], p8]]],
\{p1 \rightarrow \text{and}[\text{FUNCTION}[x]], \text{equal}[\text{cart}[V, V], \text{domain}[x]], \text{thin}[y], \text{WELLFOUNDED}[\text{inverse}[y]]\}, p2 \rightarrow \text{totalsolution}[w, x, y],
p3 \rightarrow \text{equal}[V, \text{domain}[w]], p4 \rightarrow \text{partialsolution}[w, x, y],
p5 \rightarrow \text{FUNCTION}[\text{union}[w, \text{rec}[x, y]]], p6 \rightarrow \text{subclass}[w, \text{rec}[x, y]],
p7 \rightarrow \text{subclass}[\text{rec}[x, y], w], p8 \rightarrow \text{equal}[w, \text{rec}[x, y]]]
\]

\text{Out}[39]= \text{or}[\text{equal}[w, \text{rec}[x, y]], \text{not}[\text{equal}[V, \text{domain}[\text{VERTSECT}[y]]]],
\text{not}[\text{equal}[w, \text{composite}[x, \text{history}[w, y]]]], \text{not}[\text{equal}[\text{cart}[V, V], \text{domain}[x]]], \text{not}[\text{FUNCTION}[x]],
\text{WELLFOUNDED}[\text{inverse}[y]]] = \text{True}
Restatement: if $x$ is a total binary function, and if $y$ is a thin relation whose inverse is well-founded, then $\text{rec}[x, y]$ is the only total solution.

The uniqueness theorem can also be restated using $\text{HISTORY}$ in place of $\text{history}$. This is admittedly less easy to read, but it is easier to use because more rewrite rules are available when one uses $\text{HISTORY}$.

Under the usual hypotheses, the conditions $\text{equal}[V, \text{domain}[w]]$ and $\text{subclass}[w, \text{composite}[x, \text{history}[w, y]]]$ imply that $w = \text{rec}[x, y]$. 
In practice this version of the uniqueness theorem appears to be harder to use. For the application considered below, the subclass version of the recursion condition on \( w \) led to complicated rewrites that are avoided when one uses the equal form.

---

**RANK**

The theory of rank and the Zermelo-von Neumann cumulative hierarchy were developed a few years ago without using recursive definitions. In this section, some of the facts about rank and the corresponding function RANK are reviewed. The rank of a regular set \( x \) is the lowest level of the cumulative hierarchy that holds it. The class of levels that hold a given set \( x \) is:

\[
\text{class}[w, \text{and}[\text{member}[w, \text{OMEGA}], \text{member}[x, \text{image}[z, \text{singleton}[w]]]]] / z \rightarrow \text{ZN}
\]

The intersection of this is class is

\[
\text{A}[	ext{intersection}\text{OMEGA}, \text{image}\text{inverse}\text{ZN}, \text{singleton}[x]]]
\]

Note that if \( x \) is regular, it does belong to the level of its rank.
The corresponding function \texttt{RANK} is \texttt{lambda[x, rank[x]]}, or equivalently:

\texttt{In[49]} := \texttt{VERTSECT[reify[x, rank[x]]]}
\texttt{Out[49]} = \texttt{RANK}

The connection between \texttt{rank[x]} and \texttt{RANK} can also be exhibited more simply as follows:

\texttt{In[50]} := \texttt{member[pair[x, y], RANK]}
\texttt{Out[50]} = \texttt{and[equal[y, rank[x]], member[y, V]]}

The condition that \texttt{y} be a set forces \texttt{x} to be regular:

\texttt{In[51]} := \texttt{member[pair[x, rank[x]], RANK]}
\texttt{Out[51]} = \texttt{member[x, REGULAR]}

In other words, the domain of \texttt{RANK} is the class of regular sets:

\texttt{In[52]} := \texttt{domain[RANK]}
\texttt{Out[52]} = \texttt{REGULAR}

---

**a recursion relation for RANK**

The rank of a regular set can be recursively computed using the following recursion relation:

\texttt{In[53]} := \texttt{implies[member[x, REGULAR], equal[rank[x], tc[image[RANK, x]]]]}
\texttt{Out[53]} = \texttt{True}

The corresponding recursion relation holds for the function \texttt{RANK}.

\texttt{In[54]} := \texttt{composite[TC, IMAGE[RANK]]}
\texttt{Out[54]} = \texttt{composite[RANK, IMAGE[id[REGULAR]]]}

The goal in this section is to provide a connection between these facts and the theory of well-founded recursion, using the following well-founded relation:
One needs of course to extend \textbf{RANK} to a total function. The following result summarizes the connection between \textbf{RANK} and \texttt{rec[x,y]}.

\[
\begin{align*}
\text{In}[56]:= & \quad \text{equal}[	ext{w}, \text{composite}[\text{x}, \text{HISTORY}[\text{w}, \text{y}]]] /.
\{\text{w} \to \text{union}[\text{RANK}, \text{cart}[\text{complement}[\text{REGULAR}], \text{singleton}[0]]], \\
& \quad \quad \text{x} \to \text{composite}[\text{TC}, \text{IMAGE}[\text{SECOND}], \text{SECOND}], \\
& \quad \quad \text{y} \to \text{composite}[\text{inverse}[\text{E}], \text{id}[\text{REGULAR}]]
\}
\end{align*}
\]

\texttt{Out}[56]= True

Using the \textbf{HISTORY} form of the uniqueness theorem, one finds:

\[
\begin{align*}
\text{In}[57]:= & \quad \text{SubstTest}[\text{implies}, \text{and}[	ext{FUNCTION}[\text{x}], \text{equal}[	ext{cart}[	ext{V}, \text{V}], \text{domain}[\text{x}]], \text{thin}[\text{y}], \\
& \quad \quad \text{WELLFOUND}[\text{inverse}[\text{y}]], \text{equal}[	ext{w}, \text{composite}[\text{x}, \text{HISTORY}[\text{w}, \text{y}]]], \\
& \quad \quad \text{equal}[	ext{w}, \text{rec}[\text{x}, \text{y}]], \\
& \quad \quad \{\text{w} \to \text{union}[\text{RANK}, \text{cart}[\text{complement}[\text{REGULAR}], \text{singleton}[0]]], \\
& \quad \quad \text{x} \to \text{composite}[\text{TC}, \text{IMAGE}[\text{SECOND}], \text{SECOND}], \\
& \quad \quad \text{y} \to \text{composite}[\text{inverse}[\text{E}], \text{id}[\text{REGULAR}]]
\}
\end{align*}
\]

\texttt{Out}[57]= \text{equal}[\text{rec}[\text{composite}[\text{TC}, \text{IMAGE}[\text{SECOND}], \text{SECOND}], \\
& \quad \text{composite}[\text{inverse}[\text{E}], \text{id}[\text{REGULAR}]]] := \\
& \quad \text{union}[\text{RANK}, \text{cart}[\text{complement}[\text{REGULAR}], \text{singleton}[0]]]
\]

\begin{Verbatim}
some corollaries
\end{Verbatim}

\[
\begin{align*}
\text{In}[59]:= & \quad \text{SubstTest}[\text{implies}, \\
& \quad \quad \text{and}[	ext{FUNCTION}[\text{x}], \text{thin}[\text{y}], \text{WELLFOUND}[\text{inverse}[\text{y}]], \text{FUNCTION}[\text{rec}[\text{x}, \text{y}]], \\
& \quad \quad \{\text{x} \to \text{composite}[\text{TC}, \text{IMAGE}[\text{SECOND}], \text{SECOND}], \\
& \quad \quad \text{y} \to \text{composite}[\text{inverse}[\text{E}], \text{id}[\text{REGULAR}]]
\}
\end{align*}
\]

\texttt{Out}[59]= \text{FUNCTION}[\text{union}[\text{RANK}, \text{cart}[\text{complement}[\text{REGULAR}], \text{singleton}[0]]]] = True

\[
\begin{align*}
\text{In}[60]:= & \quad \text{FUNCTION}[\text{union}[\text{RANK}, \text{cart}[\text{complement}[\text{REGULAR}], \text{singleton}[0]]]] := \text{True}
\end{align*}
\]

Lemma needed to derive a \textbf{history} form of the recursion relation:
The history form of the recursion relation holds:

\[\text{In[63]} := \text{composite[TC, IMAGE[SECOND], SECOND, history[union[RANK, cart[complement[REGULAR], singleton[0]]], composite[inverse[E], id[REGULAR]]]]}\]

\[\text{Out[63]} = \text{union[RANK, cart[complement[REGULAR], singleton[0]]]}\]