In this notebook it is shown that a (small) function is a canonical projection of an equivalence relation if and only if it can be written in the form \( \text{composite}[\text{id}[x], E] \) for some set \( x \).

**canonical projections of equivalences**

If \( q \) is an equivalence relation, the function \( \text{VERTSECT}[q] \) maps any set \( x \) to its vertical section \( \text{image}[q, \text{set}[x]] \). This vertical section is the equivalence class of \( x \), provided \( x \) belongs to the domain of \( q \).

\[
\text{In}[2] = \text{class}[\text{pair}[x, y], \text{equal}[y, \text{image}[q, \text{set}[x]]]]
\]

\[
\text{Out}[2] = \text{VERTSECT}[q]
\]

When \( q \) is a set, the restriction of the function \( \text{VERTSECT}[q] \) to the domain of \( q \) is also a set. The function \( \text{VS} \) takes \( q \) to this restriction. Comment: The combination \( \text{VERTSECT}[\text{reify}[q, \ldots]] \) is equivalent to \( \text{lambda}[q, \ldots] \), but works more efficiently.

\[
\text{In}[3] = \text{VERTSECT}[\text{reify}[q, \text{composite}[\text{VERTSECT}[q], \text{id}[\text{domain}[q]]]]]
\]

\[
\text{Out}[3] = \text{VS}
\]

Thus the canonical projection for a (small) equivalence relation \( \text{eqv}[\text{setpart}[x]] \) can be obtained by applying the function \( \text{VS} \).

\[
\text{In}[4] = \text{APPLY}[\text{VS}, \text{eqv}[\text{setpart}[x]]]
\]

\[
\text{Out}[4] = \text{composite}[\text{VERTSECT}[\text{eqv}[\text{setpart}[x]]], \text{id}[\text{fix}[\text{eqv}[\text{setpart}[x]]]]]
\]
The canonical projection for a (small) equivalence relation $q$ is the function $f = \text{APPLY}[\text{VS}, q]$. The class $\text{image}[\text{VS, EQV}]$ is therefore the class of all canonical projections of (small) equivalence relations. One can recover an equivalence relation $q$ from its canonical projection $f$ by the formula $q = \text{composite}[\text{inverse}[f], f]$. A variable-free version of this fact is:

\begin{align*}
\text{In}[5] &= \text{Map}[\text{range}, \text{SubstTest}[\text{reify}, x, \\
& \quad \text{image}[\text{composite}[\text{COMPOSE}, \text{intersection}[\text{composite}[\text{inverse}[\text{FIRST}], \text{INVERSE}], \\
& \quad \text{composite}[\text{inverse}[\text{SECOND}], \text{IMAGE}[\text{id}[\text{cart}[V, V]]]]], \\
& \quad \text{VS}, w], \text{set}[\text{setpart}[x]], w \rightarrow \text{EQUIV}]]] \text{// Reverse} \\
\text{Out}[5] &= \text{image}[\text{COMPOSE}, \text{composite}[\text{id}[\text{image}[\text{VS, EQV}]], \text{INVERSE}]] = \text{EQV} \\
\text{In}[6] &= \text{image}[\text{COMPOSE}, \text{composite}[\text{id}[\text{image}[\text{VS, EQV}]], \text{INVERSE}]] := \text{EQV}
\end{align*}

**co-restrictions**

The class of restrictions of any class $x$ is:

\begin{align*}
\text{In}[7] &= \text{class}[z, \text{exists}[y, \text{equal}[z, \text{composite}[x, \text{id}[y]]]]] \\
\text{Out}[7] &= \text{RS}[x]
\end{align*}

The class of co-restrictions of $x$ is the class of inverse of restrictions of the inverse of $x$.

\begin{align*}
\text{In}[8] &= \text{class}[z, \text{assert}[\text{exists}[y, \text{equal}[z, \text{composite}[\text{id}[y], x]]]]] \\
\text{Out}[8] &= \text{intersection}[\text{invar}[\text{composite}[\text{id}[x]], \text{inverse}[\text{SECOND}], \text{SECOND}], \\
& \quad \text{P}[\text{composite}[\text{id}[\text{domain}[\text{VERTSECT}[\text{inverse}[x]]]], x]]]
\end{align*}

Another formula for this class will be derived in this section. It is analogous to the following result for restrictions:

\begin{align*}
\text{In}[9] &= \text{range}[\text{IMAGE}[\text{composite}[\text{id}[x], \text{inverse}[\text{FIRST}]]]] \\
\text{Out}[9] &= \text{RS}[x]
\end{align*}

**Lemma.** A temporary rewrite rule.

\begin{align*}
\text{In}[10] &= (\text{image}[\text{INVERSE}, \text{RS}[\text{inverse}[x]]]) \text{// Normality // Reverse} \\
\text{Out}[10] &= \text{intersection}[\text{invar}[\text{composite}[\text{id}[x], \text{inverse}[\text{SECOND}], \text{SECOND}], \\
& \quad \text{P}[\text{composite}[\text{id}[\text{domain}[\text{VERTSECT}[\text{inverse}[x]]]], x]]] = \text{image}[\text{INVERSE}, \text{RS}[\text{inverse}[x]]]
\end{align*}

\begin{align*}
\text{In}[11] &= \text{intersection}[\text{invar}[\text{composite}[\text{id}[x_\text{__}], \text{inverse}[\text{SECOND}], \text{SECOND}], \\
& \quad \text{P}[\text{composite}[\text{id}[\text{domain}[\text{VERTSECT}[\text{inverse}[x_\text{__}]]]], x_\text{__}]] := \\
& \quad \text{image}[\text{INVERSE}, \text{RS}[\text{inverse}[x]]]
\end{align*}

**Theorem.** A formula for the class of co-restrictions of $x$.

\begin{align*}
\text{In}[12] &= (\text{range}[\text{IMAGE}[\text{composite}[\text{id}[x], \text{inverse}[\text{SECOND}]]]]) \text{// Renormality} \\
\text{Out}[12] &= \text{range}[\text{IMAGE}[\text{composite}[\text{id}[x], \text{inverse}[\text{SECOND}]]]] = \text{image}[\text{INVERSE}, \text{RS}[\text{inverse}[x]]]
\end{align*}
canonical projections of equivalences

Canonical projections are functions, but not every function is the canonical projection of an equivalence relation. Canonical projections of thin equivalence relations are of the form $\text{composite}[\text{id}[x], E]$, where $x$ is the set of equivalence classes. To derive a variable-free formulation of this result, the thinpart wrapper is first replaced with a setpart wrapper.

Next the $\text{eqv}$ and setpart wrappers are removed:

The variable $x$ can now be eliminated as follows:

Lemma.

Lemma.
One can now reformulate this property of canonical projections as follows:

\[ \text{SubstTest[implies, subclass[u, v], subclass[image[w, u], image[w, v]],}
\{u \rightarrow \text{image}\left[\text{IMAGE[SWAP], image[VS, EQV]}\right], v \rightarrow \text{RS[inverse[E]], w \rightarrow \text{INVERSE}]\}} \]

\[ \text{subclass[image[VS, EQV], image[\text{INVERSE, RS[inverse[E]]}]] = True} \]

\[ \text{Subclass[image[VS, EQV], image[\text{INVERSE, RS[inverse[E]]}]] := True} \]

It will be shown in this notebook that the reverse inclusion also holds. In other words, canonical projections of (small) equivalences are characterized by the property of being co-restrictions of \( E \).

---

**an observation**

The class of all functions of the form \( \text{APPLY[VS, x]} \) is the class of functions whose range does not hold the empty set.

\[ \text{range[VS]} \]

\[ \text{intersection[FUNS, P[cart[V, complement[set[0]]]]]} \]

This observation can be made more explicit: If the range of a function \( y \) does not hold the empty set, then \( y \) can be written as \( \text{composite[VERTSECT[z], id[domain[z]]]} \) for the relation \( z = \text{composite[inverse[E], y]} \).

\[ \text{implies[and[equal[y, funpart[w]],}
\text{not[member[0, range[y]]], equal[z, composite[inverse[E], y]],}
\text{equal[y, composite[VERTSECT[z], id[domain[z]]]]]} \]

\[ \text{Subclass[implies, and[equal[y, funpart[w]],}
\text{not[member[0, range[y]]], equal[z, composite[inverse[E], y]],}
\text{equal[y, composite[VERTSECT[z], id[domain[z]]]]], w \rightarrow y]} \]

\[ \text{or[equal[y, composite[VERTSECT[z], id[domain[z]]]], member[0, range[y]],}
\text{not[equal[z, composite[inverse[E], y]]], not[FUNCTION[y]] ] = True} \]

\[ \text{or[equal[y_, composite[VERTSECT[z_], id[domain[z_]]]], member[0, range[y_]],}
\text{not[equal[z_, composite[inverse[E], y_]]], not[FUNCTION[y_]]] := True} \]

For functions of the form \( \text{composite[id[x], E]} \), the class \( x \) must be pairwise disjoint.
Specializing the characterization of \texttt{VERTSECT} functions to those of the form \texttt{composite[id[x], E]} yields this corollary:

\begin{verbatim}
In[31]:= SubstTest[or, equal[y, composite[VERTSECT[z], id[domain[z]]]], member[0, range[y]],
not[equal[z, composite[inverse[E], y]]], not[function[y], y \rightarrow composite[id[x], E]]
Out[31]= or[equal[composite[id[x], E], composite[VERTSECT[z], id[domain[z]]]],
not[equal[z, composite[inverse[E], id[x], E]]],
not[subclass[cart[x, x], union[DISJOINT, Id]]] = True
\end{verbatim}

A variable-free formulation of the following statement will now be derived:

\begin{verbatim}
In[33]:= implies[and[function[z], equal[z, composite[id[x], E]],
equal[y, composite[inverse[z], z]], equal[z, composite[VERTSECT[y], id[domain[y]]]]]
Out[33]= True
\end{verbatim}

There are three variables to be eliminated. One can eliminate the variable \texttt{x} by simply rewriting the statement as follows:

\begin{verbatim}
In[34]:= implies[and[function[z], member[z, image[INVERSE, RS[inverse[E]]]],
equal[y, composite[inverse[z], z]], equal[z, composite[VERTSECT[y], id[domain[y]]]]]
Out[34]= True
\end{verbatim}

Lemma.

\begin{verbatim}
In[35]:= (member[pair[x, y], composite[w, id[INVERSE], inverse[SECOND]]] // AssertTest) /. w -> COMPOSE
Out[35]= member[pair[x, y], composite[COMPOSE, id[INVERSE], inverse[SECOND]]] =
and[equal[y, composite[inverse[x], x]],
member[x, V], member[y, V], subclass[x, cart[V, V]]]
\end{verbatim}

The remaining variables can now be eliminated as follows.

\begin{verbatim}
In[37]:= Map[empty[composite[Id, complement[#]]] & , SubstTest[class, pair[x, y], implies[and[member[x, y], member[pair[x, setpart[y]], v]], member[pair[setpart[y], x], w]],
{u \rightarrow intersection[FUNS, image[INVERSE, RS[inverse[E]]]],
v \rightarrow composite[COMPOSE, id[INVERSE], inverse[SECOND]], w \rightarrow VS]} // Reverse
Out[37]= subclass[intersection[FUNS, image[INVERSE, RS[inverse[E]]]],
fix[composite[VS, COMPOSE, id[INVERSE], inverse[SECOND]]]] = True
\end{verbatim}

\texttt{Inverse} \rightarrow \texttt{SetDelayed}

\texttt{Simplify}
Lemma.

\[ \text{Lemma.} \]

\[ \text{In[39]} := \text{SubstTest[fix, composite[id[u], v], \{u \rightarrow \text{FUNS, v} \rightarrow \text{composite[VS, x]}}] \} \] // Reverse

\[ \text{Out[39]} = \text{intersection[FUNS, fix[composite[VS, x]]]} = \text{fix[composite[VS, x]]} \]

\[ \text{In[40]} := \text{intersection[FUNS, fix[composite[VS, x_]]]} := \text{fix[composite[VS, x]]} \]

Lemma.

\[ \text{In[41]} := \text{SubstTest[subclass, fix[z], range[z],}
\]
\[ \text{z} \rightarrow \text{composite[VS, COMPOSE, id[INVERSE], inverse[s], id[FUNS]]}] / s \rightarrow \text{SECOND} \]

\[ \text{Out[41]} = \text{subclass[fix[composite[VS, COMPOSE, id[INVERSE], inverse[SECOND]]]}, \text{image[VS, EQV]}] = \text{True} \]

\[ \text{In[42]} := \% /. \text{Equal} \rightarrow \text{SetDelayed} \]

Theorem. Every (small) function which is a co-restriction of \(E\) is the canonical projection of some equivalence relation.

\[ \text{In[43]} := \text{SubstTest[implies, and[subclass[u, v], subclass[v, w], subclass[u, w],}
\]
\[ \{u \rightarrow \text{intersection[FUNS, image[INVERSE, RS[inverse[E]]]],}
\]
\[ v \rightarrow \text{fix[composite[VS, COMPOSE, id[INVERSE], inverse[SECOND]]]}, w \rightarrow \text{image[VS, EQV]}}] \]

\[ \text{Out[43]} = \text{subclass[intersection[FUNS, image[INVERSE, RS[inverse[E]]]], image[VS, EQV]]} \] = \text{True} \]

\[ \text{In[44]} := \% /. \text{Equal} \rightarrow \text{SetDelayed} \]

Combining this result with the reverse inclusion yields an equation.

\[ \text{In[45]} := \text{SubstTest[and, subclass[u, v], subclass[v, u],}
\]
\[ \{u \rightarrow \text{intersection[FUNS, image[INVERSE, RS[inverse[E]]]], v \rightarrow \text{image[VS, EQV]}}] \]

\[ \text{Out[45]} = \text{True} = \text{equal[image[VS, EQV], intersection[FUNS, image[INVERSE, RS[inverse[E]]]]]} \]

\[ \text{In[46]} := \text{intersection[FUNS, image[INVERSE, RS[inverse[E]]]]} := \text{image[VS, EQV]} \]

\[ \text{comment} \]

The class of co-restrictions of the membership relation \(E\) that appears in the formula for \(\text{image[VS,EQV]}\) can also be written as a certain subclass of the power class \(P[E]\) as follows:

\[ \text{In[47]} := \text{image[INVERSE, RS[inverse[E]]]} // \text{Normality} // \text{Reverse} \]

\[ \text{Out[47]} = \text{intersection[invar[composite[id[E], inverse[SECOND], SECOND]], P[E]}] = \text{image[INVERSE, RS[inverse[E]]]]} \]

\[ \text{In[48]} := \text{intersection[invar[composite[id[E], inverse[SECOND], SECOND]], P[E]}] := \text{image[INVERSE, RS[inverse[E]]]]} \]