Aclosure for unital submonoids

Johan G. F. Belinfante
2013 September 11

```
In[1]:= SetDirectory["l:"]; << goedel.13sep10a

        :Package Title: goedel.13sep10a              2013 September 10 at 11:40 a.m.

        Loading takes about seventeen minutes, half that time due to builtin pauses.

        It is now:  2013 Sep 11 at 5:57

        Loading Simplification Rules

        TOOLS.M is now incorporated in the GOEDEL program as of 2010 September 3

        weightlimit = 40

        Loading completed.

        It is now:  2013 Sep 11 at 6:14
```

summary

The set of unital submonoids of a monoid **x** is closed under arbitrary (non-empty) intersections. The key to deriving this result is a characterization of unital submonoids as submonoids that hold the point **pair[pair[e[x], e[x]], e[x]]**.

a characterization of unital submonoids

Theorem.

```
In[2]:= Map[not, SubstTest[and, implies[p1, p2],
           implies[p2, p3], not[implies[p1, p3]], {p1 → member[w, MONOIDS],
            p2 → member[w, SEMIGPS], p3 -> or[equal[e[w], e[x]], not[member[x, MONOIDS]],
              not[member[pair[pair[e[x], e[x]], e[x]], w]], not[subclass[w, x]]]}]] // Reverse

Out[2]= or[equal[e[w], e[x]], not[member[w, MONOIDS]], not[member[x, MONOIDS]],
          not[member[pair[pair[e[x], e[x]], e[x]], w]], not[subclass[w, x]]] == True

In[3]:= or[equal[e[w_], e[x_]], not[member[w_, MONOIDS]], not[member[x_, MONOIDS]],
          not[member[pair[pair[e[x_], e[x_]], e[x_]], w_]], not[subclass[w_, x_]]] := True
```

Observation.

```
In[4]:= implies[and[member[w, MONOIDS], subclass[w, x], member[x, MONOIDS]],
          equiv[equal[e[w], e[x]], member[pair[pair[e[x], e[x]], e[x]], w]]] // not // not

Out[4]= True
```

the main theorem

For convenience, the wrapper **monoid[x]** will be used. In general **monoid[x]** could be either empty or a monoid. In the present situation however the hypothesis **pair[pair[monoid[x], monoid[x]], monoid[x]] ∈ w** rules out the possibility that **monoid[x]** be empty.

Lemma.

```
In[5]:= SubstTest[Aclosure, intersection[SEMIGPS, P[semigp[t]]], t → monoid[x]] // Reverse

Out[5]= Aclosure[intersection[SEMIGPS, P[monoid[x]]]] == intersection[SEMIGPS, P[monoid[x]]]

In[6]:= Aclosure[intersection[SEMIGPS, P[monoid[x_]]]] := intersection[SEMIGPS, P[monoid[x]]]
```

Theorem.

```
In[7]:= SubstTest[implies, subclass[u, v],
          subclass[Aclosure[u], Aclosure[v]], {u -> intersection[MONOIDS, complement[
              P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
            P[monoid[x]]], v -> intersection[SEMIGPS, P[monoid[x]]]}] // Reverse

Out[7]= subclass[Aclosure[intersection[MONOIDS, complement[
              P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
            P[monoid[x]]], SEMIGPS] == True

In[8]:= (% /. x → x_) /. Equal → SetDelayed
```

Lemma.

```
In[9]:= SubstTest[implies, and[subclass[u, v], subclass[v, w]],
          subclass[u, w], {u -> Aclosure[intersection[MONOIDS, complement[
               P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
             P[monoid[x]]], v -> intersection[SEMIGPS, complement[
             P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
            P[monoid[x]]], w → MONOIDS}] // Reverse

Out[9]= subclass[Aclosure[intersection[MONOIDS, complement[
              P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
            P[monoid[x]]], MONOIDS] == True

In[10]:= (% /. x → x_) /. Equal → SetDelayed
```

Main Theorem. The class of unital submonoids of a monoid is closed under arbitrary intersections.

```
In[11]:= SubstTest[implies, subclass[Aclosure[t], t],
           equal[Aclosure[t], t], t -> intersection[MONOIDS, complement[
             P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]], P[
             monoid[x]]]] // Reverse

Out[11]= equal[Aclosure[intersection[MONOIDS, complement[
             P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
           P[monoid[x]]]], intersection[MONOIDS, complement[
           P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]], P[
           monoid[x]]]] == True

In[12]:= Aclosure[intersection[MONOIDS, complement[
           P[complement[set[pair[pair[e[monoid[x_]], e[monoid[x_]]], e[monoid[x_]]]]]],
         P[monoid[x_]]]] := intersection[MONOIDS, complement[P[complement[
           set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]], P[monoid[x]]]
```

Corollary. (Eliminating the **monoid** wrapper.)

```
In[13]:= SubstTest[implies, equal[x, monoid[t]],
           equal[Aclosure[intersection[MONOIDS, complement[
               P[complement[set[pair[pair[e[x], e[x]], e[x]]]]], P[x]]], intersection[
             MONOIDS, complement[P[complement[set[pair[pair[e[x], e[x]], e[x]]]]],
             P[x]]], t → x] // Reverse // MapNotNot

Out[13]= or[equal[Aclosure[intersection[MONOIDS,
             complement[P[complement[set[pair[pair[e[x], e[x]], e[x]]]]], P[x]]],
           intersection[MONOIDS, complement[P[complement[set[pair[pair[e[x], e[x]], e[x]]]]],
            P[x]]], not[member[x, MONOIDS]]] == True

In[14]:= or[equal[Aclosure[intersection[MONOIDS, complement[
              P[complement[set[pair[pair[e[x_], e[x_]], e[x_]]]]], P[x_]]], intersection[
            MONOIDS, complement[P[complement[set[pair[pair[e[x_], e[x_]], e[x_]]]]],
            P[x_]]], not[member[x_, MONOIDS]]] := True
```

Observation. One can replace **Aclosure[–]** with **fix[HULL[–]]** because the class of unital submonoids of a monoid is a set.

```
In[15]:= fix[HULL[intersection[MONOIDS, complement[P[complement[
             set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]], P[monoid[x]]]]]

Out[15]= intersection[MONOIDS,
          complement[P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]],
          P[monoid[x]]]
```

Theorem. The intersection of any non-empty collection of unital submonoids of **monoid[x]** is a monoid.

```
In[16]:= SubstTest[implies, and[equal[fix[HULL[t]], t], subclass[w, t], not[empty[w]]],
           member[A[w], t], t -> intersection[MONOIDS, complement[
             P[complement[set[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]]]]]],
            P[monoid[x]]]] // Reverse // MapNotNot

Out[16]= or[equal[0, w], member[A[w], MONOIDS],
           not[member[pair[pair[e[monoid[x]], e[monoid[x]]], e[monoid[x]]], A[w]]],
           not[subclass[w, MONOIDS]], not[subclass[U[w], monoid[x]]]] == True

In[17]:= or[equal[0, w_], member[A[w_], MONOIDS],
           not[member[pair[pair[e[monoid[x_]], e[monoid[x_]]], e[monoid[x_]]], A[w_]]],
           not[subclass[w_, MONOIDS]], not[subclass[U[w_], monoid[x_]]]] := True
```

Corollary. (Eliminating the **monoid** wrapper.)

```
In[18]:= SubstTest[implies, equal[x, monoid[t]], or[equal[0, w],
            member[A[w], MONOIDS], not[member[pair[pair[e[x], e[x]], e[x]], A[w]]],
            not[subclass[w, MONOIDS]], not[subclass[U[w], x]]], t → x] // Reverse // MapNotNot

Out[18]= or[equal[0, w], member[A[w], MONOIDS],
           not[member[x, MONOIDS]], not[member[pair[pair[e[x], e[x]], e[x]], A[w]]],
           not[subclass[w, MONOIDS]], not[subclass[U[w], x]]] == True

In[19]:= or[equal[0, w_], member[A[w_], MONOIDS],
           not[member[pair[pair[e[x_], e[x_]], e[x_]], A[w_]]], not[member[x_, MONOIDS]],
           not[subclass[w_, MONOIDS]], not[subclass[U[w_], x_]]] := True
```

Theorem. The submonoid **A[w]** is unital.

```
In[20]:= Map[not, SubstTest[and, implies[p1, p2],
            implies[p1, p3], implies[and[p1, p2, p3], p4], not[implies[p1, p4]],
            {p1 → and[subclass[w, MONOIDS], not[empty[w]], subclass[U[w], x], member[x, MONOIDS],
               member[pair[pair[e[x], e[x]], e[x]], A[w]]], p2 -> member[A[w], MONOIDS],
             p3 → subclass[A[w], x], p4 → equal[e[A[w]], e[x]]}]] // Reverse

Out[20]= or[equal[0, w], equal[e[x], e[A[w]]],
           not[member[x, MONOIDS]], not[member[pair[pair[e[x], e[x]], e[x]], A[w]]],
           not[subclass[w, MONOIDS]], not[subclass[U[w], x]]] == True

In[21]:= or[equal[0, w_], equal[e[A[w_]], e[x_]],
           not[member[pair[pair[e[x_], e[x_]], e[x_]], A[w_]]], not[member[x_, MONOIDS]],
           not[subclass[w_, MONOIDS]], not[subclass[U[w_], x_]]] := True
```