Hausdorff Paradox in ACL2(r)
Definitions and Theorems

Jagadish Bapanapally
University of Wyoming
Laramie, Wyoming
jbapanap@uwyo.edu

1 Introduction

This is a summary of the Hausdorff paradox with all the definitions and theorems.
Statement: There is a countable set $D \subseteq S^2$ such that $S^2 - D$ can be divided into 5 pieces which can be rotated to form 2 copies of $S^2 - D$.

1. **Definition of a group of reduced words in ACL2(r):**

   Representing in Math:
   $F(a,b) = \{1\} \uplus w(a) \uplus w(a^{-1}) \uplus w(b) \uplus w(b^{-1})$

```
(defun reducedwordp (x)
  (or (a-wordp x)
      (a-inv-wordp x)
      (b-wordp x)
      (b-inv-wordp x)
      (equal x '())))
```

   I have proved that the sets a-wordp, a-inv-wordp, b-wordp, b-inv-wordp and '() are disjoint in ACL2(r).

   corollaries:
   $F(a,b) = a^{-1}w(a) \uplus w(a^{-1})$
   $F(a,b) = b^{-1}w(b) \uplus w(b^{-1})$

   Definitions in ACL2(r):
   $a^{-1}w(a)$:

```
(defun-sk a-inv*w-a-p (w)
  (exists word-a
          (and (a-wordp word-a)
               (equal (compose (list (wa-inv)) word-a) w))))
```

   $b^{-1}w(b)$:

```
(defun-sk b-inv*w-b-p (w)
  (exists word-b
          (and (b-wordp word-b)
               (equal (compose (list (wb-inv)) word-b) w))))
```

Submitted to:

corollaries in ACL2(r):

```
(defthmd reducedword-equiv-4
  (iff (reducedwordp w)
       (or (a-inv*w-a-p w)
           (a-inv-wordp w))))

(defthmd reducedword-equiv-2
  (iff (reducedwordp w)
       (or (b-inv*w-b-p w)
           (b-inv-wordp w))))
```

2. **Definition of a point in** $\mathbb{R}^3$ in ACL2(r)

```
(defun point-in-r3 (x)
  (and (array2p :fake-name x)
       (equal (car (dimensions :fake-name x)) 3)
       (equal (cadr (dimensions :fake-name x)) 1)
       (realp (aref2 :fake-name x 0 0))
       (realp (aref2 :fake-name x 1 0))
       (realp (aref2 :fake-name x 2 0))))
```

3. **Set of points that belong to the surface of a unit sphere centered at the origin**

```
(defun s2-def-p (point)
  (and (point-in-r3 point)
       (equal (+ (* (aref2 :fake-name point 0 0) (aref2 :fake-name point 0 0))
                 (* (aref2 :fake-name point 1 0) (aref2 :fake-name point 1 0))
                 (* (aref2 :fake-name point 2 0) (aref2 :fake-name point 2 0)))
              1)))
```

Let's say this set is S^2.

There is a one to one relation between the set of rotations and the set of reduced words. Lets name the set of rotations as $R(a,b)$. Then,
$R(a,b) = 1 \uplus R(a) \uplus R(a^{-1}) \uplus R(b) \uplus R(b^{-1})$

4. **Set of all points of** S^2 **fixed by any non-identity rotation** $\rho \in R(a,b)$
In math: $D = \{p \in S^2 \mid \text{There is a } \rho \in R(a,b), \rho \neq 1, \text{with } \rho(p) = p\}$

Definition in ACL2(r):

```
(defun-sk word-exists (point)
  (exists w
          (and (reducedwordp w)
               w
               (m-= (m-* (rotation w (acl2-sqrt 2)) point)
                    point))))

(defun d-p (point)
  (and (s2-def-p point)
       (word-exists point)))
```

5. $S^2 - D$ in ACL2(r):

Definition in ACL2(r):

```
(defun s2-d-p (point)
  (and (s2-def-p point)
       (not (d-p point))))
```

6. Orbit of a point $p \in S^2$

In math: $\{\rho(p) \mid \rho \in R(a,b)\}$

Definition in ACL2(r):

Returns true if o-point belongs to the orbit of point.

```
(defun-sk orbit-point-p-q (o-point point)
  (exists w
          (and (reducedwordp w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) point) o-point))))
```

7. Choice set for the orbits of points in the set $S^2 - D$:

Lets name this set M.

Definition in ACL2(r):

```
(defchoose choice-set-s2-d-p (c-point) (p)
  (and (point-in-r3 c-point)
       (orbit-point-p-q c-point p))
  :strengthen t)
```

8. Partitions of $S^2 - D$

(a) First partition:

In math:

$S^2 - D = R(a,b)M = M \uplus R(a)M \uplus R(a^{-1})M \uplus R(b)M \uplus R(b^{-1})M$

Definitions in ACL2(r):

Set $R(a,b)M$:

```
(defun-sk diff-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (reducedwordp w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))

(defun-sk diff-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-s2-d-p-q p))
```

Set M:

```
(defun-sk diff-n-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (reducedwordp w)
               (equal w nil)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))

(defun-sk diff-n-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-n-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-n-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-n-s2-d-p-q p)))
```

Set $R(a)M$:

```
(defun-sk diff-a-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (a-wordp w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))

(defun-sk diff-a-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-a-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-a-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-a-s2-d-p-q p)))
```

Set $R(b)M$:

```
(defun-sk diff-b-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (b-wordp w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))

(defun-sk diff-b-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-b-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-b-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-b-s2-d-p-q p)))
```

Set $R(a^{-1})M$:

```
(defun-sk diff-a-inv-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (a-inv-wordp w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))
```

```
(defun-sk diff-a-inv-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-a-inv-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-a-inv-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-a-inv-s2-d-p-q p)))
```

Set $R(b^{-1})M$:

```
(defun-sk diff-b-inv-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (b-inv-wordp w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))

(defun-sk diff-b-inv-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-b-inv-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-b-inv-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-b-inv-s2-d-p-q p)))
```

Theorems in ACL2(r):

```
(defthmd s2-d-p-equiv
  (iff (s2-d-p p)
       (diff-s2-d-p p)))

(defthmd diff-s2-d-p-equivalence-1
  (iff (diff-s2-d-p p)
       (or (diff-n-s2-d-p p)
           (diff-a-s2-d-p p)
           (diff-a-inv-s2-d-p p)
           (diff-b-s2-d-p p)
           (diff-b-inv-s2-d-p p))))

(defthmd s2-d-p-equivalence-1
  (iff (s2-d-p p)
       (or (diff-n-s2-d-p p)
           (diff-a-s2-d-p p)
           (diff-a-inv-s2-d-p p)
           (diff-b-s2-d-p p)
           (diff-b-inv-s2-d-p p))))
```

(b) Second partition
In math:
$S^2 - D \ = \ a^{-1}(R(a)M) \ \uplus R(a^{-1})M$

Definitions in ACL2(r):
Set $a^{-1}(R(a)M)$:

```
(defun-sk diff-a-inv-wa-s2-d-p-q-1 (cp1 p)
  (exists w
          (and (a-inv*w-a-p w)
               (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))


(defun-sk diff-a-inv-wa-s2-d-p-q (p)
  (exists p1
          (and (s2-d-p p1)
               (diff-a-inv-wa-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-a-inv-wa-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-a-inv-wa-s2-d-p-q p)))
```

Set $a^{-1}(R(a)M)$:

```
(defun-sk a-inv-diff-a-s2-d-p-1 (p)
  (exists p1
          (and (diff-a-s2-d-p p1)
               (m-= (m-* (rotation (list (wa-inv)) (acl2-sqrt 2)) p1) p))))

(defun a-inv-diff-a-s2-d-p (p)
  (and (point-in-r3 p)
       (a-inv-diff-a-s2-d-p-1 p)))
```

Theorems in ACL2(r):

```
(defthmd diff-a-inv-wa-s2-d-p-equiv
  (iff (diff-a-inv-wa-s2-d-p p)
       (a-inv-diff-a-s2-d-p p)))

(defthmd diff-s2-d-p-equivalence-2
  (iff (diff-s2-d-p p)
       (or (diff-a-inv-wa-s2-d-p p)
           (diff-a-inv-s2-d-p p))))

(defthmd s2-d-p-equivalence-2
  (iff (s2-d-p p)
       (or (a-inv-diff-a-s2-d-p p)
           (diff-a-inv-s2-d-p p))))
```

(c) Third partition
In math:
$S^2 - D \ = \ b^{-1}(R(b)M) \ \uplus R(b^{-1})M$

Definitions in ACL2(r):
Set $b^{-1}(R(b)M)$:

```
(defun-sk diff-b-inv-wb-s2-d-p-q-1 (cp1 p)
  (exists w
```

```
        (and (b-inv*w-b-p w)
             (m-= (m-* (rotation w (acl2-sqrt 2)) cp1) p))))

(defun-sk diff-b-inv-wb-s2-d-p-q (p)
  (exists p1
        (and (s2-d-p p1)
             (diff-b-inv-wb-s2-d-p-q-1 (choice-set-s2-d-p p1) p))))

(defun diff-b-inv-wb-s2-d-p (p)
  (and (point-in-r3 p)
       (diff-b-inv-wb-s2-d-p-q p)))
```

Set $b^{-1}(R(b)M)$:

```
(defun-sk b-inv-diff-b-s2-d-p-1 (p)
  (exists p1
        (and (diff-b-s2-d-p p1)
             (m-= (m-* (rotation (list (wb-inv)) (acl2-sqrt 2)) p1) p))))

(defun b-inv-diff-b-s2-d-p (p)
  (and (point-in-r3 p)
       (b-inv-diff-b-s2-d-p-1 p)))
```

Theorems in ACL2(r):

```
(defthmd diff-b-inv-wb-s2-d-p-equiv
  (iff (diff-b-inv-wb-s2-d-p p)
       (b-inv-diff-b-s2-d-p p)))

(defthmd diff-s2-d-p-equivalence-3
  (iff (diff-s2-d-p p)
       (or (diff-b-inv-wb-s2-d-p p)
           (diff-b-inv-s2-d-p p))))

(defthmd s2-d-p-equivalence-3
  (iff (s2-d-p p)
       (or (b-inv-diff-b-s2-d-p p)
           (diff-b-inv-s2-d-p p))))
```

I have proved that the sets (diff-n-s2-d-p p), (diff-a-s2-d-p p), (diff-a-inv-s2-d-p p), (diff-b-s2-d-p p), (diff-b-inv-s2-d-p p) are disjoint and the sets (a-inv-diff-a-s2-d-p p), (diff-a-inv-s2-d-p p) are disjoint and the sets (b-inv-diff-b-s2-d-p p), (diff-b-inv-s2-d-p p) are disjoint.

2 To Do

To completely prove the Hausdorff paradox, I have to prove that the set D is countable.