Commutativity Theorems in Groups with Power-like Maps

R. Padmanabhan and Yang Zhang
Department of Mathematics
University of Manitoba
Winnipeg, MB, R3T 2N2, Canada
Emails: {Ranganathan.Padmanabhan; Yang.Zhang}@umanitoba.ca

There are several commutativity theorems in groups and rings which involve power maps $f(x) = x^n$. The most famous example of this kind is Jacobson's theorem which asserts that any ring satisfying the identity $x^n = x$ is commutative. Such statements belong to first order logic with equality and hence provable, in principle, by any first-order theorem-prover. However, because of the presence of an arbitrary integer parameter n in the exponent, they are outside the scope of any first-order theorem-prover. In particular, one cannot use such an automated reasoning system to prove theorems involving power maps. Here we focus just on the needed properties of power maps $f(x) = x^n$ and show how one can avoid having to reason explicitly with integer exponents. Implementing these new equational properties of power maps, we show how a theorem-prover can be a handy tool for quickly proving or confirming the truth of such theorems.

1. INTRODUCTION

In response to a friendly challenge by the first author, Professor B. H. Neumann proved several semigroup implications in group theory (see, [10]). Way back in 1979, Nicholson and Yaqui[11] proved a class of commutativity theorems in groups. All these implications involve power maps, i.e., $x \to x^n$, where n stands for an arbitrary integer variable. But for the presence of n, all the implications belong to the first order logic with equality. However, these statements are not provable in the first-order theorem provers because of the presence of n in the exponent. In [2], Michael Beeson has modified Otter, a precursor of Prover9, to perform Peano-like arithmetic and mathematical induction thus demonstrating the feasibility and usefulness of a first-order prover. In this paper, we give new first order proofs of these implications by extracting equational properties of power-maps. Consequently these proofs remain valid for all natural numbers n but the length and complexity of the proofs remain the same independent of the actual values of n. This is because the integer parameter n never physically appears in the input file - only its formal first order properties! We demonstrate this by using the software Prover9, a popular first-order theorem prover developed by the late Dr. William McCune[6].

2. POWER-LIKE MAPS

Let us start with the semigroup implication of B.H. Neumann (Theorem 17, [10]). In a group $(G, *)$, we have the following commutativity theorem: for a fixed positive integer n,

$$x^{1+n} * y^2 * x^n = y * x^{1+2n} * y \quad \Longrightarrow \quad x * y = y * x, \text{ for all } \ x, y \in G.$$

But for the presence of the integer variable n in the hypothesis, this statement belongs to the first order equational theory of semigroups. However, statements involving an arbitrary integer parameter lie outside the scope of first order logic. In fact, it is impossible to even input the above statement in a first-order theorem-prover without specifying an actual value for n. In this paper, we define the concept of "power-like maps" by extracting some crucial equational properties valid for $f(x) : x \to x^n$ in groups and semigroups. Then we reformulate the above theorem of B.H. Neumann in the language of power-like maps and demonstrate that an automated first-order theorem prover can derive the commutativity as a consequence.

Definition 1. Let $(G; *,', e)$ be a group, where $'$ is the inverse of group operator $*$ and e is a group identity. A unary map $f : G \to G$ is called power-like map if the map satisfies the following: for any $x, y \in G$,

(i) $x * f(y * x) = f(x * y) * x$.
(ii) $f(x * x) = f(x) * f(x)$.
(iii) $f(e) = e$.
(iv) $x * y = y * x \Longrightarrow f(x) * y = y * f(x)$
(v) $x' * f(y) * x = f(x' * y * x)$

If g is another power-like function, then we demand

(vi) $f(x) * g(x) = g(x) * f(x)$.
(vii) $f(g(x)) = g(f(x))$.

It is easy to check the validity of these conditions for any power map f in a group or a semigroup.

Theorem 1. (Nicholson and Yaqub [11]) Let m and n be relatively prime positive integers. If a group $(G, *)$ satisfies the two identities $x^m * y^m = y^m * x^m$ and $x^n * y^n = y^n * x^n$, for all $x, y \in G$ then G must be commutative.

Historical Remark. Several proofs of this classical theorem have been churned out during the past 50+ years starting from a problem of J.R. Isbell [3] raised in the American Math Monthly. Using a deep embedding theorem of Krempa and Macedonska [5], it is possible to give an easy proof of this theorem for cancellation semigroups. However, as early as 1983, Chen-Te Yen[13] gave a direct elementary proof of this for cancellation semigroups without using any group theory. In spite of such plethora of proofs, the interest in this theorem has not diminished. Thus G. Venkataraman in [9] proved this for the special case of finite groups with $m = 2$ and $n = 3$. In a recent publication [1], Araujo and Kinyon proved this for completely regular semigroups.

Here we give a new first order proof using only power-like functions thereby completely avoiding the use of integer parameters. This enables one to create an input file for a first order theorem prover to formally prove such theorems involving power maps. To prove this theorem by using power-like functions, we first establish the following lemma.

Lemma 1. Let $(G, *)$ be a cancellation semigroup with $g(x)$ a power-like map of G. Let $f(x) = x * g(x)$. If the semigroup G satisfies the two conditions

$$g(x) * g(y) = g(y) * g(x) \quad \text{and} \quad f(x) * f(y) = f(y) * f(x)$$

then the semigroup G is commutative.

Proof. We consider the properties of $g(x)$ first.

$$\begin{aligned} g(x * g(y)) * g(y) &= g(y) * g(x * g(y)) && \text{since g's commute} \\ &= g(g(y) * x) * g(y) && \text{since g is power-like} \end{aligned}$$

Hence $g(x * g(y)) = g(g(y) * x)$ by right cancellation. Also,

$$\begin{aligned} f(x * g(y)) * f(x) &= f(x) * f(x * g(y)) && \text{since f's commute} \\ &= f(x) * g(x * g(y)) * x * g(y) && \text{by definition of } f(x) \\ &= f(x) * g(g(y) * x) * x * g(y) && \text{by the above step} \\ &= g(x) * x * g(g(y) * x) * x * g(y) && \text{by definition of } f \\ &= g(x) * g(x * g(y)) * x * x * g(y) && \text{since g is power-like} \\ &= g(x * g(y)) * g(x) * x * x * g(y) && \text{since g's commute} \\ &= g(x * g(y)) * x * x * g(x) * g(y) && \text{since g is power-like} \\ &= g(x * g(y)) * x * x * g(y) * g(x) && \text{since g's commute} \end{aligned}$$

Rewriting $f(x)$ in terms of $g(x)$, we have

$$f(x * g(y)) * x * g(x) = g(x * g(y)) * x * x * g(y) * g(x)$$

Right canceling $g(x)$, we get $f(x * g(y)) * x = g(x * g(y)) * x * x * g(y)$,

Again, expressing f in terms of g, the above equation becomes

$$g(x * g(y)) * x * g(y) * x = g(x * g(y)) * x * x * g(y).$$

Left-canceling the term $g(x * g(y)) * x$, we obtain $g(y) * x = x * g(y)$.

Also, by assumption, we have $f(x) * f(y) = f(y) * f(x)$. Rewriting this in terms of g, we get $x * g(x) * y * g(y) = y * g(y) * x * g(x)$. Since $g(x)$ and $g(y)$ are now central elements, we get the desired conclusion $x * y = y * x$ after canceling the common element $g(x) * g(y)$. This completes the proof of Lemma 1. □

Prover9 input/out proofs of Lemma 1 are given in the Appendix.

Proof of Theorem 1 (for cancellation semigoups).

Let now $(G; *)$ be a cancellation semigroup satisfying the two identities

$$x^m * y^m = y^m * x^m, \quad x^n * y^n = y^n * x^n$$

for some positive integers m and n with $\gcd(m, n) = 1$, and all $x, y \in G$.

Since m and n are relatively prime, there exists natural numbers a and b such that $am = 1 + bn$. Now define $g(x) = x^{bn}$ and $f(x) = x^{am}$. It is clear that $f(x) = x * g(x)$. Also, $f(x)$ and $g(x)$ are power-like, indeed they are the usual power maps! Moreover,

$$f(x) * f(y) = x^{am} * y^{am} = (x^a)^m * (y^a)^m = (y^a)^m * (x^a)^m = f(y) * f(x)$$

and similarly $g(x) * g(y) = g(y) * g(x)$. Hence by the Lemma 1, the cancellation semigroup G is commutative. $\square$

Thanks to the familiar inverse law $(xy)^{-1} = y^{-1}x^{-1}$, whenever an identity $f = g$ is valid in a group, then its mirror image $f^{\sim} = g^{\sim}$ is also valid where the word $f^{\sim}$ is obtained from f by simply writing it in the reverse order. For example, the mirror image of the identity $xy^2x^2 = yx^3y$ is $x^2y^2x = yx^3y$. Notice that the term yx^3y is self-dual i.e. it is equal to its own mirror image. Since all terms x^n are clearly self-dual and since a power-like function $f(x)$ is just an abstraction of a power function x^n, we naturally postulate that all power-like functions are self-dual.

Hence it is immediately clear that, in groups, the following chain of semigroup implications are valid:

	$xy^2x^2 = yx^3y,$	by the given hypothesis
$\Longrightarrow$	$xy^2x^2 = x^2y^2x,$	since, by the mirror principle, two sides both are equal to yx^3y.
$\Longrightarrow$	$y^2x = xy^2,$	after two cancellations of$x$
$\Longrightarrow$	$xy^2x^2 = yxyx^2,$	since the term $x^2$is now a central element
$\Longrightarrow$	$xy^2 = yxy,$	after right cancellations ofx^2
$\Longrightarrow$	$xy = yx,$	after a right cancellation of$y$.

This was the context in which the friendly challenge was made that was alluded to by Professor B.H. Neumann in [10]. In the process he gave a sequence of new semigroup implications which are valid in groups. Being a veteran group theorist, Professor Neumann uses, naturally, a good amount of commutator calculus and conjugates in his proofs. However, our proof uses only the methods of equational logic of semigroups and the mirror principle. No deep group theory like commutator calculus or conjugates are used here, thus paving the way for eventually proving their validity in the much larger theory of cancellation semigroups. Needless to say that we employ the power-like maps instead of the actual power maps and thus provable in, any first-order theorem-prover like Prover9 because the hypothesis is now free from integer exponents. Also, unlike the usual power maps, commutators and conjugates are not that well related in the context of power-like maps (see, however, [7]). Here, we have appended the full Prover9 proof output done on an iMac apart from providing a new human proof.

Theorem 2. (B.H. Neumann [10], Theorem 17). In a group $(G; *)$, we have the following implication

$$x^{1+t} * y^2 * x^t = y * x^{2t+1} * y \quad \text{implies} \quad x * y = y * x \text{ for all } x, y \in G.$$

Human Proof. Consistent with the theme of this paper, we reformulate and prove the above in terms of power-like functions. Accordingly, we claim the implication

$$x * g(x) * y^2 * g(x) = y * x * g(x) * g(x) * y \quad \text{implies} \quad x * y = y * x$$

where $g(x)$ is an arbitrary power-like function in the group G.

Since x and $g(x)$ commute, the term $y * x * g(x) * g(x) * y$ is its own mirror image. Hence we get the equation

$$g(x) * x * y^2 * g(x) = g(x) * y^2 * x * g(x)$$

from which, we derive, after on two cancellations of $g(x)$, $x * y^2 = y^2 * x$. In other words, squares are central elements in our group. Now rewriting the given equation, we get

$$x * y^2 * g(x) * g(x) = y * x * y * g(x) * g(x).$$

Finally, three right cancellations yield the desired commutativity $x * y = y * x$. □

Taking the special case of $t(x) = x^t$, we get the above mentioned theorem of B.H. Neumann for groups as a corollary. A Prover9 INPUT file is given in the Appendix.

3. APPENDIX

```
======================== INPUT file for Lemma 1 ===================
%% Proof of Lemma 1, PadZhang JFR_2018
%% Cancellation semigroups defined
(x * y) * z = x * (y * z).
x * y != x * z | y = z.
x * y != z * y | x = z.

%% power-like functions defined
f(x) = x * g(x). %% i.e. f = 1+g
g(x) * x = x * g(x).  %% power-like properties
g(x * y) * x = x * g(y * x). %% power-like properties
f(x) *  g(x) = g(x) * f(x).
x * f(y * x) = f(x * y) * x.
f(x * x) = f(x) * f(x).
g(x) * g(x) = g(x * x).
f(x) * x = x * f(x).

%% Hypothesis of Lemma 1
f(x) * f(y) = f(y) * f(x).
g(x) * g(y) = g(y) * g(x).

%% goal to derive commutativity
x * y = y * x.

======================= End INPUT file ===========================
```

The output file of Lemma 1.

```
================================ prooftrans ==========================
Prover9 (32) version Dec-2007, Dec 2007.
Process 898 was started  on yangzhangsimac2.ad.umanitoba.ca,
Wed Nov 21 14:47:21 2018
The command was "/Users/yangzhang/Desktop/Prover9-Mace4-v05B.app
/Contents/Resources/bin-mac-intel/prover9".
```

```
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 105.74 (+ 2.52) seconds.
% Length of proof is 38.
% Level of proof is 8.
% Maximum clause weight is 23.
% Given clauses 1492.

1 x * y = y * x # label(non_clause) # label(goal).  [goal].
2 (x * y) * z = x * (y * z).  [assumption].
3 x * y != x * z | y = z.  [assumption].
4 x * y != z * y | x = z.  [assumption].
5 f(x) = x * g(x).  [assumption].
6 g(x) * x = x * g(x).  [assumption].
7 g(x * y) * x = x * g(y * x).  [assumption].
15 f(x) * f(y) = f(y) * f(x).  [assumption].
16 x * (g(x) * (y * g(y))) = y * (g(y) * (x * g(x))).
    [copy(15),rewrite([5(1),5(3),2(5),5(6),5(8),2(10)])].
17 g(x) * g(y) = g(y) * g(x).  [assumption].
18 c2 * c1 != c1 * c2.  [deny(1)].
21 x * (y * z) != u * z | x * y = u.  [para(2(a,1),4(a,1))].
22 g(x) * (x * y) = x * (g(x) * y).  [para(6(a,1),2(a,1,1)),
     rewrite([2(3)]),flip(a)].
26 g(x * y) * (x * z) = x * (g(y * x) * z).  [para(7(a,1),2(a,1,1)),
     rewrite([2(4)]),flip(a)].
35 x * (g(x) * (y * g(y))) != y * z | g(y) * (x * g(x)) = z.
     [para(16(a,1),3(a,1))].
38 g(x) * (g(y) * z) = g(y) * (g(x) * z).  [para(17(a,1),2(a,1,1)),
     rewrite([2(4)])].
40 g(x) * g(y) != g(y) * z | g(x) = z.  [para(17(a,1),3(a,1))].
44 x * (c2 * c1) != x * (c1 * c2).  [ur(3,b,18,a)].
61 g(c2 * c1) * (c1 * c2) != c2 * (c1 * g(c2 * c1)).
     [para(6(a,1),44(a,1)),rewrite([2(8)]),flip(a)].
75 g(x) * y != x * (g(x) * z) | x * z = y.
     [para(22(a,1),3(a,1)),flip(a)].
91 x * (y * z) != u * (w * z) | x * y = u * w.
     [para(2(a,1),21(a,2))].
105 g(c2 * c1) * (c1 * (c2 * x)) != c2 * (c1 * (g(c2 * c1) * x)).
       [ur(21,b,61,a),rewrite([2(8),2(18),2(17)])].
124 g(g(x) * y) = g(y * g(x)).  [hyper(40,a,7,a)].
131 x * (g(y * x) * z) != u * (x * z) | g(x * y) = u.
       [para(26(a,1),4(a,1))].
```

```
170 g(g(x) * y) * (y * z) = y * (g(g(x) * y) * z).
       [para(124(a,2),26(a,1,1))].
184 g(x) * (g(y) * x) = g(y) * (x * g(x)).
       [para(6(a,1),38(a,1,2)),flip(a)].
186 g(x * y) * (g(z) * x) = g(z) * (x * g(y * x)).
       [para(7(a,1),38(a,1,2)),flip(a)].
369 x * (g(y) * (g(x) * y)) = y * (g(y) * (x * g(x))).
       [para(184(a,2),16(a,1,2))].
562 x * (g(x) * (y * g(x * y))) = y * (x * (g(x) * g(y * x))).
       [hyper(35,a,2,a),rewrite([2(5),22(6),26(11),17(10)])].
567 g(c2 * c1) * (c1 * (c2 * g(x))) != c2 *(c1*(g(x)*g(c2 * c1))).
       [para(17(a,1),105(a,2,2,2))].
1604 g(x) * y != z * (g(x) * (g(z) * x)) | x * (z * g(z)) = y.
         [para(369(a,2),75(a,2))].
2631 g(c2 * c1) * (c1 * (c2 * g(c1))) != c1* g(c1)*(c2 *g(c1* 2))).
         [para(562(a,2),567(a,2))].
2640 g(c2 * c1) * (c1 * (c2 * g(c1))) != c1*(g(c2*c1)*(g(c1)*c2)).
         [para(186(a,2),2631(a,2,2))].
2901 g(x) * (y * (y * g(y))) = y * (g(y) * (g(x) * y)).
         [hyper(1604,a,170,a),rewrite([2(5)])].
2921 g(x) * (y * (y * g(y))) = y * (g(x) * (y * g(y))).
          [para(38(a,1),2901(a,2,2)),rewrite([6(8)])].
2978 g(x) * y = y * g(x).  [hyper(91,a,2921,a)].
2979 g(x * y) = g(y * x).  [hyper(131,a,2921,a(flip))].
3030 $F.  [back_rewrite(2640),rewrite([2979(4),26(11),2979(5),
         2978(10),2(10),2979(16),2978(20),2978(21),2(21)]),xx(a)].

============================== end of proof ======================
```

Next we present the INPUT and OUTPUT files for Theorem 2.

```
======================== INPUT file for Theorem 2 =================
%% Group theory defined
(x * y) * z = x * (y * z).
x * e = x.
x * x' = e.

%% power-like functions defined
f(x) = x * t(x). %% "degree" of f = 1+t
t(x) * x = x * t(x).  %% power-like properties
t(x * y) * x = x * t(y * x). %% power-like properties
t(x) * t(x) = t(x * x). %% power-like properties
t(x)' = t(x'). %% power-like properties

%% BHN 2001 CMS/also Macedonska&Krempa 1992
%% the powermap x^t is replaced by a power-like t(x)
```

```
f(x) * ((y * y) * t(x)) = y * ((f(x) * t(x)) * y).

%% goal to derive commutativity
x * y - y * x.
======================= END of INPUT file =====================
```

The output of the proof.

```
=========================== PROOF =============================


=========================== prooftrans ========================
Prover9 (32) version Dec-2007, Dec 2007.
Process 776 was started by admin on Admins-iMac.hitronhub.home,
Sat Nov 10 21:29:53 2018
=========================== end of head =======================

=========================== PROOF =============================

% -------- Comments from original proof --------
% Proof 1 at 3.32 (+ 0.17) seconds.
% Length of proof is 53.
% Level of proof is 18.
% Maximum clause weight is 24.
% Given clauses 480.

1 x * y = y * x # label(non_clause) # label(goal).  [goal].
2 (x * y) * z = x * (y * z).  [assumption].
3 x * e = x.  [assumption].
4 x * x' = e.  [assumption].
5 f(x) = x * t(x).  [assumption].
6 t(x) * x = x * t(x).  [assumption].
7 t(x * y) * x = x * t(y * x).  [assumption].
8 t(x) * t(x) = t(x * x).  [assumption].
9 t(x)' = t(x').  [assumption].
10 f(x) * ((y * y) * t(x)) = y * ((f(x) * t(x)) * y). [assumption].
11 x * (t(x) * (y * (y * t(x)))) = y * (x * (t(x * x) * y)).
    [copy(10),rewrite([5(1),2(5),2(6),5(7),2(10),8(9),2(10)])].
12 c2 * c1 != c1 * c2.  [deny(1)].
13 x * (e * y) = x * y.  [para(3(a,1),2(a,1,1)),flip(a)].
14 x * (x' * y) = e * y.  [para(4(a,1),2(a,1,1)),flip(a)].
15 x * (y * (x * y)') = e.  [para(4(a,1),2(a,1)),flip(a)].
16 t(x) * (x * y) = x * (t(x) * y).  [para(6(a,1),2(a,1,1)),
     rewrite([2(3)]),flip(a)].
19 t(e*x) = e*t(x). [para(7(a,1),3(a,1)),rewrite([3(3)]),flip(a)].
22 t(x) * (t(x) * y) = t(x * x)*y. [para(8(a,1),2(a,1,1)),flip(a)].
24 t(x) * t(x') = e.  [para(9(a,1),4(a,1,2))].
45 e * x'' = x.  [para(4(a,1),14(a,1,2)),rewrite([3(2)]),flip(a)].
```

```
50 x * y'' = x * y.  [para(45(a,1),2(a,2,2)),rewrite([3(2)])].
53 e * x = x.  [para(45(a,1),13(a,2)),rewrite([50(5),13(4)])].
54 t(x'') = t(x). [para(45(a,1),19(a,1,1)),rewrite([53(6)]),flip(a)].
61 x * (x' * y) = y.  [back_rewrite(14),rewrite([53(5)])].
70 t(x') * t(x) = e.  [para(54(a,1),24(a,1,2))].
72 x'' = x.  [para(4(a,1),61(a,1,2)),rewrite([3(2)]),flip(a)].
75 x' * (x * y) = y.  [para(72(a,1),61(a,1,2,1))].
85 t(x') * (t(x) * y) = y.
    [para(70(a,1),2(a,1,1)),rewrite([53(2)]),flip(a)].
94 x * (y * x)' = y'.
    [para(15(a,1),75(a,1,2)),rewrite([3(3)]),flip(a)].
97 x * ((y * x)' * z) = y' * z.  [para(94(a,1),2(a,1,1)),flip(a)].
99 x * (y * (z * x))' = (y * z)'.  [para(2(a,1),94(a,1,2,1))].
107 x' * y' = (y * x)'.  [para(94(a,1),75(a,1,2))].
108 (x * y)' * x = y'.  [para(94(a,1),94(a,1,2,1)),rewrite([72(4)])].
114 (x * (y * (t(y * y) * x)))' * y = (t(y) * (x * (x * t(y))))'.
       [para(11(a,1),108(a,1,1,1))].
115 (x' * y)' = y' * x.  [para(61(a,1),108(a,1,1,1)),flip(a)].
117 (x * (t(x) * y))' *t(x) = (x*y)'. [para(16(a,1),108(a,1,1,1))].
158 t(x') * y' = (y * t(x))'.  [para(94(a,1),85(a,1,2))].
159 x' * t(y') = (t(y) * x)'.  [para(85(a,1),108(a,1,1,1))].
357 (t(x) * (y * (y * (x * t(x)))))' = (y * (x * (t(x * x) * y)))'.
     [para(159(a,1),11(a,1,2,2,2)),rewrite([107(8),2(6),158(8),2(6),
       2(5),107(8),2(6),2(5),2(4),6(3),107(12),158(14),107(14),2(12),
       7(11),107(14),2(12),2(11)])].
368 x * ((y * (z * x))' * u) = (y * z)' * u.
       [para(2(a,1),97(a,1,2,1,1))].
381 x * (y * (z * (u * x)))' = (y * (z * u))'.
     [para(2(a,1),99(a,1,2,1,2))].
677 (x * (y * (t(y * y) * x)))' * t(y) = (y * (x * (x * t(y))))'.
       [para(11(a,1),117(a,1,1,1))].
1668 t(x * x) * (y * (t(x) * (y * (y * t(x))))')  = (y * x)' * x.
        [para(114(a,1),368(a,1,2)),rewrite([2(10)])].
2407 (x * (y * (y * t(x))))' = (y * (y * (x * t(x))))'.
          [para(357(a,1),108(a,1,1)),rewrite([677(8)])].
2784 x * (x * (x * y))' = (y * x)'.
         [para(2407(a,1),99(a,1,2)),rewrite([2(8),381(7)])].
2811 (x * y)' * y = y * (y * x)'.  [para(61(a,1),2784(a,1,2,1,2)),
          rewrite([2(6),115(7)]),flip(a)].
2819 x * (y * y) = y * (y * x).  [para(2784(a,1),108(a,1,1,1)),
          rewrite([72(3),2(2),72(6)])].
2829 t(x * x) * (t(x * x) * y)' = y'.  [para(2784(a,1),22(a,1,2)),
           rewrite([94(5),22(7)]),flip(a)].
3003 t(x * x) * (y * (t(x) * (y * (y * t(x))))') = x * (x * y)'.
          [back_rewrite(1668),rewrite([2811(13)])].
3066 t(x) * (y * (y * t(x))) = t(x * x) * (y * y).
          [para(2819(a,1),22(a,1,2))].
```

```
3487 x * (x * y)' = y'.  [back_rewrite(3003),
           rewrite([3066(7),99(8),2829(7)]),flip(a)].
3777 x * y = y * x.
     [para(94(a,1),3487(a,1,2,1)),rewrite([72(2),72(4)])].
3778 $F.  [resolve(3777,a,12,a)].

============================== end of proof ========================
```

Acknowledgment:

This research was supported by Canada NSERC and UofM URGP.

References

[1] F. Araujo and M. Kinyon, *Commutativity theorems for groups and semigroups.* Port. Math. 74 (2017), no. 3, 243 - 255.

[2] M. Beeson, Mathematical induction in Otter-lambda. J. Automat. Reason. 36 (2006), no. 4, 311 - 344.

[3] J. R. Isbell, *Commuting Powers in a Group.* Amer. Math. Monthly, 77 (1970), 909.

[4] J. R. Isbell and B. M. Green, *E2259.* American Math Monthly, Vol 78 (1971), 909-910. (DOI: 10.2307/2316502)

[5] J. Krempa and O. Macedonska, *On identities of cancellative semigroups.* Contemp. Math., 131, Part 3, Amer. Math. Soc., Providence, R.I. 1992.

[6] W. McCune, *Prover9, version 2009-02A.* http://www.cs.unm.edu/ mccune/prover9/.

[7] W. McCune and R. Padmanabhan, *Automated deduction in equational logic and cubic curves.* Lecture Notes in Artificial Intelligence 1095. Springer-Verlag, Berlin, 1996.

[8] G. I. Moghaddam and R. Padmanabhan, *Commutativity theorems for cancellative semigroups.* Semigroup Forum 95(2017), no. 3, 448-454.

[9] G. I. Moghaddam, R. Padmanabhan and Yang Zhang, *Automated reasoning with power-maps.* submitted.

[10] B. H. Neumann, *Some semigroup laws in groups.* Canad. Math. Bull. 44 (2001), no. 1, 93-96.

[11] W. K. Nicholson and Yaqub Adil, *A commutativity theorem for rings and groups.* Canad. Math. Bull. 22 (1979), no. 4, 419-423.

[12] G. Venkataraman, *Groups in which squares and cubes commute.* arXiv: 1605.05463v1.

[13] Chen-Te Yen, *On the Commutativity of Rings and Cancellative Semigroups.* Chinese J. of Math, Vol 11 (1983), 99-113.