
Maple Tools for Use in Conjecture Testing and Iteration Mappings in Number Theory

Douglas B. Meade¹ and Charles A. Nicol²

The purpose of this paper is to i) introduce procedures which can be used in the investigation of a general class of problems in number theory and ii) illustrate some of the powerful new features of Maple V, Release 2.

Introduction

Let \mathbf{N}_0 be the set of non-negative integers and f a function from \mathbf{N}_0 to \mathbf{N}_0 . Given an initial value $n_0 \in \mathbf{N}_0$, consider the sequence $\{n_i\}$ defined by $n_{i+1} := f(n_i)$, for $i \in \mathbf{N}_0$. Our interest is in whether the sequence $\{n_i\}$ contains only a finite number of distinct elements. That is, after some initial iterations, does the sequence become periodic. (Note that this includes the cases when the iterates reach a fixed point of the function.)

We have implemented Maple procedures which assist in the investigation of problems of this type. Our presentation will cover three specific examples. The first two examples are well-known and relatively simple. The third example, which was the driving force behind the development of these tools, is currently being studied by Nicol.

Example 1

The Collatz, or $3n + 1$, problem [3], [5] is the most well-known problem of this type. For this problem the function f is

$$f(n) := \begin{cases} 3n + 1 & \text{if } n \text{ is odd} \\ n/2 & \text{if } n \text{ is even} \end{cases} \quad (1)$$

or, as the Maple procedure `Collatz`,

```
> Collatz := proc( n:nonnegint ) # 3n+1 Problem
>   option remember;
>
>   if ( type(n,odd) ) then
>     3*n+1
>   else
>     n/2
>   fi
> end;
```

¹Department of Mathematics, University of South Carolina, Columbia, SC 29208. meade@math.sc.edu

²Department of Mathematics, University of South Carolina, Columbia, SC 29208. nicol@math.sc.edu

It has been conjectured that the sequence of iterates of Collatz starting from any positive integer becomes periodic with “cycle” 4, 2, 1.

The `Iterate` procedure can be used to motivate this conjecture.

```

> _ShowRpt:=true: _FreqRpt:=1:
> Iterate( 5, Collatz );

          0, 5, 5
          1, 16, 16
          2, 8, 16
          3, 4, 16
          4, 2, 16
          5, 1, 16
          6, 4, 16

Number of Iterations = 6; Cycle length = 3
          [6, 3, [4, 2, 1]]

> _ShowRpt:=false:
> for init from 10 to 15 do init, Iterate( init, Collatz ) od;

          10, [7, 3, [4, 2, 1]]
          11, [15, 3, [4, 2, 1]]
          12, [10, 3, [4, 2, 1]]
          13, [10, 3, [4, 2, 1]]
          14, [18, 3, [4, 2, 1]]
          15, [18, 3, [4, 2, 1]]

```

Once the conjecture has been formulated, we provide a more efficient facility for directly testing the conjecture. In preparation, we define a Boolean function which evaluates to `true` when the conjecture holds, and `false` otherwise. The `CollatzCycle` procedure is one possibility.

```

> CollatzCycle := proc( n:nonnegint )
>     local ans, r, s;
>     r := _ShowRpt; s := _ShowPlt;
>     _ShowRpt := false; _ShowPlt := false;
>     ans := evalb( convert(op(3,Iterate(n,Collatz)),set) = {1,2,4} );
>     _ShowRpt := r; _ShowPlt := s;
>     RETURN(ans)
> end:

```

The conversion of the cycle to a set is needed to avoid treating the initial values 1 and 2 as special cases. (Check the output of `Iterate(2, Collatz)`);).

The arguments to the `Conjecture` procedure are the procedure to be used to test for consistency with the conjecture and the domain on which the conjecture is to be tested.

```
> Conjecture := proc( condition:procedure, domain:{nonnegint,list,set} )
>     local result;
>     result := map( condition, domain, args[3..nargs] );
>     RETURN( result );
> end;
```

It should be mentioned that, were it not for the optional arguments that are permitted in `\Conjecture`, this function is just an alias to the builtin function `map`. The following example takes about 2 minutes of CPU time on a Sun SPARCstation 2 with 32MB of memory

```
> st:=time(): Conjecture( CollatzCycle, {$1..1000} ); time()-st;
                                     {true}
                                     62.867
```

Example 2

Another interesting function to consider is $f(n) := \sigma(n) - n$, where, for all $n \in \mathbf{N}_0$, $\sigma(n)$ is the sum of all divisors of n . To begin, try

```
> Catalan := proc( n:nonnegint ) # Catalan problem
>     option remember;
>
>     sigma(n)-n
> end;
>
> _ShowRpt:=false:
> for init from 1 to 10 do init, Iterate( init, Catalan ) od;
                                     1, [2, 1, [0]]
                                     2, [3, 1, [0]]
                                     3, [3, 1, [0]]
                                     4, [4, 1, [0]]
                                     5, [3, 1, [0]]
                                     6, [1, 1, [6]]
                                     7, [3, 1, [0]]
                                     8, [4, 1, [0]]
                                     9, [5, 1, [0]]
                                    10, [5, 1, [0]]
```

These limited results could lead the reader to think that these sequences always reach a fixed point of the function, and that the fixed point is usually 0. We test this conjecture as follows:

```

> HasCycleLength := proc( n:nonnegint, fn, len: posint )
>   local ans, r, s;
>   r := _ShowRpt; s := _ShowPlt;
>   _ShowRpt := false; _ShowPlt := false;
>   ans := evalb( Iterate(n,fn)[2] = len );
>   _ShowRpt := r; _ShowPlt := s;
>   RETURN( ans )
> end:
>
> Conjecture( HasCycleLength, {$1..100}, Catalan, 1 );
>
>                                     {true}
>
> Conjecture( HasCycleLength, {$1..250}, Catalan, 1 );
>
>                                     {false, true}
>
>
> check := Conjecture( HasCycleLength, [$1..250], Catalan, 1):
> member( false, check, 'first' );
>
>                                     true
>
> Iterate( first, Catalan );
>
>                                     [2, 2, [220, 284]]

```

It is still possible that these sequences become periodic, but with cycles of varying lengths. This is the well-known conjecture of Catalan [1] (see also [4]). The `IsPeriodic` procedure can be used to test this conjecture

```

> IsPeriodic := proc( n:nonnegint, fn )
>   local ans, r, s;
>   r := _ShowRpt; s := _ShowPlt;
>   _ShowRpt := false; _ShowPlt := false;
>   ans := evalb( Iterate(n,fn)[2] <> -1 );
>   _ShowRpt := r; _ShowPlt := s;
>   RETURN( ans )
> end:
>
> Conjecture( IsPeriodic, {$1..250}, Catalan );
>
>                                     {true}

```

An interesting question is what length cycles will be found. The reader is challenged to implement a procedure for use with `Conjecture` which would be of use in studying this question.

Example 3

Consider the sequence formed by the iterates of $f := \phi \circ \sigma$, i.e. the composition of ϕ and σ , where ϕ is the totient or Euler “ ϕ ” function. The corresponding Maple procedure, `PhiSigma`, is

```

> PhiSigma := proc( n:nonnegint ) # Composition of phi and sigma
>   option remember;
>
>   phi( sigma(n) )

```

> end:

When, in 1932, Poulet [8] looked at this problem, he observed that the sequence of iterates eventually became periodic in all of the numerical examples he considered. Subsequent work by various mathematicians has provided additional information about these sequences, but no proof of the conjecture (see, e.g., [2], [7]).

Our discussion of this problem will be brief. The following example illustrates some of the different situations that have been encountered. These include sequences which reach a fixed point after a large number of iterations, sequences where the cycle consists of several hundred elements, and one sequence for which no cycle has been found. It is not possible to include the output in this article.³

```
> _ShowRpt:=false: _MaxIter:=2500:
> init_vals := [ 360^15, 6^35, 2^5*7^6*11^2, 2^59*3^11*5^3,
>               290^2, 2^47*3^23*5^11, 2^47*3^23*5^11*23 ];

init_vals := [221073919720733357899776000000000000000,
              1719070799748422591028658176, 455536928, 12764786611036820078592000, 84100,
              646946998725511426867200000000000, 14879780970686762817945600000000000]

> for init in init_vals do init, Iterate( init, PhiSigma )[1..2] od;

                221073919720733357899776000000000000000, 1455, 595
                    1719070799748422591028658176, 254, 1
-----
More iterations for 455536928? (Enter y; or n;) n;
More iterations for 455536928? (Enter y; or n;) n;
                    455536928, 2500, -1
                12764786611036820078592000, 663, 1
                    84100, 2, 1
                646946998725511426867200000000000, 2222, 595
                14879780970686762817945600000000000, 1490, 595
```

The iterates starting from $2^5 7^2 11^2 = 45536928$ did not cycle in the first 13,261 iterations (about 2 CPU days). In independent studies Sid Graham has observed that this appears to be the smallest number which does not cycle.

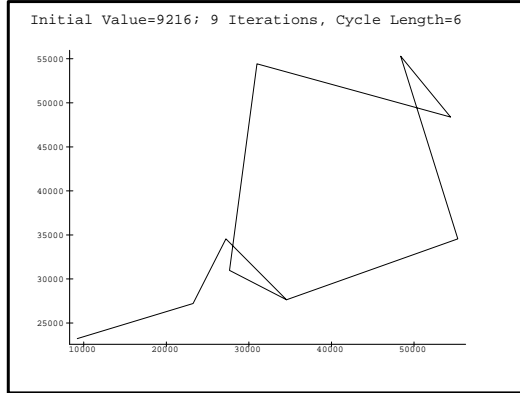
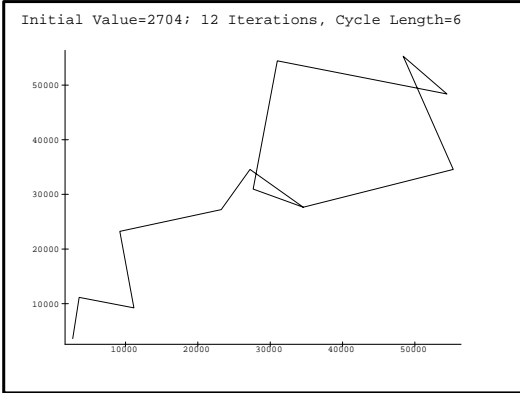
These sequences often involve numbers in excess of 10^{50} (this explains why the above example takes so long to run!). We have found a graphical representation of the sequence to be useful. To have these plots generated it is necessary to assign the global variable `_ShowPlt` the value `true`. This example shows two initial values which terminate with the same cycle.

```
> _ShowRpt:=false: _ShowPlt:=true:
> init_vals := [ 2704, 9216 ];

                init_vals := [2704, 9216]
```

³The Maple programs, including a script which generates all the examples in this paper, are available from the authors.

```
> for init in init_vals do init, Iterate( init, PhiSigma ) od;
      2704, [12, 6, [34560, 27648, 30976, 54432, 48384, 55296]]
      9216, [9, 6, [34560, 27648, 30976, 54432, 48384, 55296]]
> _ShowPlt:=false:
```



There are many other interesting examples that can be considered. We mention only the sequences formed with $f := \phi \circ \sigma \circ \sigma$. It is conjectured [2] that for sufficiently large initial values, this sequence tends to ∞ . In [6] it is shown that this conjecture, if true, implies the finiteness of the Mersenne primes.

Pre-Images

Another tool that is useful in the study of these sequences is the *pre-images* for the function f . The pre-image of $M \in \mathbf{N}_0$ for the function f is the set of all $n \in \mathbf{N}_0$ for which $f(n) = M$. Oftentimes these sets can be very difficult to determine. `PreImage` is a general function which returns the pre-image (as a list) for a given function. To ensure that all elements of the set have been obtained, it is also necessary to provide a monotone increasing lower bound for the function.

For the function in Example 3, an appropriate lower bound is the composition of monotone lower bounds for ϕ and σ . For ϕ we find an appropriate bound in [9];

$$\phi(n) > \frac{n}{e^\gamma \log \log n + \frac{2.50637}{\log \log n}}, \quad \text{for } n \geq 3;$$

a trivial bound for σ is $\sigma(n) > n$.

```
> PreImage( 8, PhiSigma, phi_lowbound );
      [8, [8, 14, 15, 19, 23, 29]]
> PreImage( [1..8], PhiSigma, phi_lowbound );
[[1, [1]], [2, [2, 3, 5]], [4, [6, 7, 11]], [6, [4, 10, 13, 17]],
 [8, [8, 14, 15, 19, 23, 29]]]
```

It should be noted that the algorithm used in `PreImage` is not very sophisticated. The amount of work is determined by the largest element in the first argument to `PreImage`. In particular, the previous two examples require the same amount of work, but the second provides much more information.

With the delivery of Maple V, Release 2 the `numtheory` library contains `invphi`, which computes pre-images for the `phi` function (without specifying a monotone lower bound). In spite of the generality of `PreImage` and the brute force algorithm which it uses, we found that it out performed `invphi` in many cases. This difference is significantly more noticeable as the number of factors in the prime factorization of the first argument increases.

```
> st:=time(): PreImage( 960, phi, phi_lowbound ): time()-st;
                               159.050
> st:=time(): invphi( 960 ): time()-st;
                               396.983
>
> st:=time(): PreImage( 1000, phi, phi_lowbound ): time()-st;
                               74.000
> st:=time(): invphi( 1000 ): time()-st;
                               4.667
>
> st:=time(): PreImage( 4032, phi, phi_lowbound ): time()-st;
                               984.950
```

```
#st:=time(): invphi( 4032 ): time()-st; # used 104539.8 Maple CPU seconds
```

That's right, this last command took a little more than 29 Maple CPU hours! Our unsophisticated function required less than 1 percent of the CPU time used by `invphi`.

Conclusion

We have presented several tools for use in the investigation of sequences formed by the iteration of an arithmetic function and the testing of conjectures. These procedures illustrate the effective use `seq`, `map`, and other powerful features of Maple V, Release 2, including graphics.

Included in this work is a general pre-image facility, `PreImage`. It should be noted that we have found many non-trivial cases in which `PreImage` is much faster than the `invphi` function in the `numtheory` library.

One of the constraints in our investigation is the prime factorization routine used by Maple. The consideration of larger examples in an efficient manner, it seems to be necessary to replace the current Maple factorization algorithm with a more efficient algorithm, such as the elliptic curve method.

Acknowledgments

The ideas for this paper developed from Charles Nicol's collaboration with various colleagues, including Michael Filaseta, Sid Graham and John Selfridge.

References

- [1] E. Catalan, *Bull. Soc. Math. France* 16 (1887–88), pp. 128–129.
- [2] Leon Alaoglu and Paul Erdős, A conjecture in elementary number theory, *Bull. Amer. Math. Soc.* 50 (1944), pp. 881–882.
- [3] Gaston H. Gonnet, Computations on the $3n + 1$ Conjecture, *Maple Technical Newsletter*, Number 6, Fall 1991, pages 18–22.
- [4] Richard K. Guy, *Unsolved Problems in Number Theory*, vol. 1, Springer-Verlag, 1981.
- [5] Jeffrey Lagarias, The $3x + 1$ problem and its generalizations, *American Mathematical Monthly*, vol 92, no. 1, January 1985, pages 3–23.
- [6] A. Makowski and A. Schinzel, On the functions $\phi(n)$ and $\sigma(n)$, *Colloquium Mathematicum*, vol. XIII, 1964, Fasc. 1, pp. 95–99.
- [7] Carl Pomerance, On the composition of the arithmetic functions σ and ϕ , *Colloquium Mathematicum*, vol. LVII, 1989, Fasc. 1, pp. 11–15.
- [8] P. Poulet, Nouvelles suites arithmétiques, *Sphinx*, vol. 2 (1932), pp. 53–54.
- [9] J. Barkley Rosser and Lowell Schoenfeld, Approximate formulas for some functions of prime numbers, *Illinois Journal of Mathematics*, vol. 6, no. 1, March 1962, pp. 64–94.