```
> ### Example 7.11. p = 55681 for (2.6).mws
```

A role for generalised Fermat numbers

by

John B. Cosgrave and Karl Dilcher

A reminder of the meaning of our paper's congruence (2.6): for $n = p^{\alpha}\, q_1^{\beta_1}\, q_2^{\beta_2} \ldots q_s^{\beta_s}$, distinct primes $p, q_1, q_2, \ldots, q_s = 1, -1, -1, \ldots, -1 \pmod 6$ we seek solutions of the Gauss factorial congruence

$$\mathrm{floor}\left(\left(\frac{1}{6}\ \{n-1\} \right)_n ! \right) = 1 \pmod n$$

In this Maple-to-pdf conversion we exhibit all solutions of our paper's congruence (2.5) for the prime $p = 55681$, a specially chosen non-standard Jacobi prime.

It should be emphasised that for $p = 55681$ the '$s = 7$' is the current limit for which we can make a definitive statement concerning the exact number of solutions of (2.6). A similar statement for $s = 8$ would require knowing the complete factorisation of the 608-digit $55681^{27} + 1$ (that '7' being $8 - 1$).

This $p = 55681$ satisfies $ord_p\left(\frac{1}{3}\ \{p-1\} \right) = 3.2^6$, and $ord_p\left(\frac{1}{6}\ \{p-1\} \right) = 3.2^5$.

Procedures

```
> with(numtheory): ### needed for 'order'

  with(combinat): ### needed for 'choose'

  ### 01:

  PI := proc(n, M, i) local k, r; r := 1:
        for k from floor(((i-1)*(n-1)/M + 1)) to i*(n-1)/M do
          if igcd(n, k) = 1 then r := mods(r*k, n); fi; od; r; end:

  ### 02:

  PRFAC := proc(le, la, p, alpha) local r, k, MOD; r := le; MOD := p^alpha;

           for k from (le+1) to la do r := mods(r*k, MOD) od; r; end:

  ### 03:

  the_ones := proc(n, M) local L, p; L := []: for p in factorset(n) do if p mod M = 1 then

              L := [op(L), p] fi od; L; end:

  ### 04:

  the_minus_ones := proc(n, M) local L, p; L := []: for p in factorset(n) do if mods(p, M) = -1
                    then L := [op(L), p] fi od; L; end:
```

```
### 05:

Pow := proc(n, p) local t, a; t := n: a := 0: while t mod p = 0 do t := t/p: a := a+1: od: a;
end:

### 06:

### "more" simply means that 's' (in application) > 1
### (i.e., there is 'more' than ONE 'q')

### Bear in mind that in using this procedure we are making the understanding that p
### occurs only to the 1st power, and 'w' will be square-free ('primitive' solutions)

### 'L' REPLACES the INITIAL 'w'
### 'w' becomes a local variable, DEFINED as the product of the members of L (the 'q')

PI6_more_modified := proc(L, s, p, fac) local w, EF, FAC, R;

        if s < 2 then lprint(`Remember that s should be greater than 1.`); RETURN(); fi;

            w := mul(q, q in L):

           EF := mods(w&^((p - 1)/3), p):   ### Euler-Fermat element for s > 1.

          FAC := mods( fac&^(2^s), p):      ### This is the Gauss 6 closed form mod p

     if mods(w, 6) = 1 and s mod 2 = 0 then

               R := mods(EF * FAC, p):

   elif mods(w, 6) = 1 and s mod 2 = 1 then

              R := mods(FAC / EF, p):

   elif mods(w, 6) = -1 and s mod 2 = 0 then

              R := mods(1 / ( EF * FAC ), p):

   elif mods(w, 6) = -1 and s mod 2 = 1 then

              R := mods(EF / FAC, p):

      fi;

    R; end:

#########################################################################

### The following PI6 - of which we make only limited use - tests some individual n-values
### It allows for having 'alpha' > 1

### 07:

### The following PHI6 uses the D.H. Lehmer formulae for the
### PHI-values of w = 1/-1 (mod 6) for w having NO prime factor = 1 (mod 6)

PHI6 := proc(w) local s; s := nops(factorset(w)):

   if w mod 6 = 1 then (phi(w) + 2^(s+1))/6 elif mods(w, 6) = -1 then (phi(w) - 2^(s+1))/6 fi;
end:

### 08:

PI6 := proc(n) local p, a, Gf, w, s, signs, PHIw6,

                     Sw, Q, PARI, EF1, q1, sign1, EF, Rpa, Rw, R;

           p := op(the_ones(n, 6)):                  ### This gives 'p'
           a := Pow(n, p):                           ### This gives 'a', i.e. 'alpha'

  if a = 1 then Gf := PRFAC(1, (p-1)/6, p, 1) elif a > 1 then Gf := PI(p^a, 6, 1) fi:

     ### This is ((p^a - 1)/6)_p! mod p^a, speeded up in the case a(lpha) = 1

           w := n/(p^a):                             ### This is 'w'
        PARI := proc(w) if mods(w, 6) = 1 then 1     ### This is '1' if w = 1 (mod 6)
                elif mods(w, 6) = -1 then -1 fi end: ### but is '-1' if w = -1 (mod 6)

           s := nops(factorset(w)):   ### This is 's'
       signs := (-1)^s:               ### This is '-1' at ODD 's', and '1' at EVEN 's'
       PHIw6 := PHI6(w):

          Sw := factorset(w):         ### The set of all ('s' of) the 'q'
           Q := mul(q, q = Sw):       ### The product of all ('s' of) the 'q'
```

```
    ### We need TWO EULER-FERMATS (EF1 and EF) to distinguish between s = 1 and s > 1:

    if s = 1 then

            q1 := op(1, factorset(w)):           ### This is 'q[1]'
         sign1 := (-1)^((p+q1)/6):               ### the sign element in the closed form
           EF1 := mods(q1&^(phi(p^a)/6), p^a):   ### the Euler-Fermat element for q[1].
                                                 ### NOTE the 6th-root
           Rpa := mods(sign1 * EF1 * Gf^2, p^a):
            Rw := mods((-1)^((p-1)/6)/p&^PHIw6, w): ### Note the EXTRA SIGN element here at Gauss
6

             R := mods(chrem([Rpa^PARI(w), Rw], [p^a, w]), n):

                                                 ### Note the 1/Rpa, as with Gauss 4 closed forms
  elif s > 1 then

            EF := mods(Q&^(phi(p^a)/3), p^a):     ### Euler-Fermat element for s > 1. Note the
3rd-root
          Rpa := mods(EF^signs*Gf^(2^s), p^a):   ### Note the SIGN element in the EF term
           Rw := mods(1/p&^PHIw6, w):            ### There is NO SIGN element here at s > 1

             R := mods(chrem([Rpa^PARI(w), Rw], [p^a, w]), n):

    fi; R; end:
```

The Gauss 6 support primes for Jacobi $p = 55681$

```
>              p := 55681:

     level||2[p] := [5, 29, 11, 2531]:              ### These divide (55681 - 1)*(55681 + 1)*
(55681^2 + 1)

     level||3[p] := [41, 121477457]:                ### These divide (55681^(2^2) + 1)

     level||4[p] := [17, 12075324422351249]:        ### These divide 55681^(2^3) + 1

     level||5[p] := []:                             ### No 'q' dividing 556811^(2^4) + 1
                                                    ### Karl complete factorisation

     level||6[p] := [257, 610817, 476600704619911891073, 494039575542372154409346497]:

                                                    ### These divide 55681^(2^5) + 1
                                                    ### Our complete factorisation

     level||7[p] := [652980135409104837678582610371183252063988 49,

5430065006389584982184906089683575088937603970131954895946359874894517106136643531669045869571635
9316934391090865493136004628257534364175802467156932863636609772154081500828008188169170737952634
9679898975924600406498220496441124102521925202319337217]:

                                                    ### These divide 55681^(2^6) + 1
                                                    ### Our complete factorisation
                                                    ### Tues 27th Jan 2015
                                                    ### the latter has 249 digits

                                  ### COMPLETE TO HERE


              LEV := 7:

    LEVEL||LEV[p] := []:

   for lev from 2 to LEV do

 for q||lev in level||lev[p] do LEVEL||LEV[p] := [ op(LEVEL||LEV[p]), q||lev ]; od od:

 print(``); LEVEL||LEV[p]; print(``);
```

[5, 29, 11, 2531, 41, 121477457, 17, 12075324422351249, 257, 610817, 476600704619911891073, 494039575542372154409346497, 652980135409104837678582610371183252063988 49,

```
5430065006389584982184906089683575088937603970131954895946359874894517106136643531669045869571635931693439109\
0908654931360046282575343641758024671569328636366097721540815008280081881691707379526349679898975924600406498\
98220496441124102521925202319337217]
```
(2.1)

```
> length
  (5430065006389584982184906089683575088937603970131954895946359874894517106136643531669045869571635931693439109\
  0908654931360046282575343641758024671569328636366097721540815008280081881691707379526349679898975924600406498\
  98220496441124102521925202319337217);
                                  249
```
(2.2)

```
> seq(isprime(q), q = LEVEL||LEV[p]);
        true, true, true, true, true, true, true, true, true, true, true, true, true, true
```
(2.3)

```
> seq(q mod 6, q = LEVEL||LEV[p]);
                          5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5
```
(2.4)

```
> nops(LEVEL||LEV[p]);
                                  14
```
(2.5)

```
> print(``);
  [[2], seq((p - 1)*(p + 1)*( p^2 + 1) mod q, q = level||2[p])];
  for LEV from 3 to 7 do if level||LEV[p] <> [] then
  print([[LEV], seq(p&^(2^(LEV - 1)) + 1 mod q, q = level||LEV[p])]); fi od;

                              [[2], 0, 0, 0, 0]
                                [[3], 0, 0]
                                [[4], 0, 0]
                              [[6], 0, 0, 0, 0]
                                [[7], 0, 0]
```
(2.6)

```
> ### The following determines any q-support for which q^2 is a factor:
```

```
> print(``);
  for q in level||2[p] do if (p - 1)*(p + 1)*(p^2 + 1) mod q^2 = 0 then print([2], q) fi od:
  for LEV from 3 to 7 do if level||LEV[p] <> [] then
  for q in level||LEV[p] do if p&^(2^(LEV - 1)) + 1 mod q^2 = 0 then print([LEV], q) fi od: fi od;
```
(2.7)

Thus, in the case of $p = 55681$, there is no square support.

```
>   p := 55681;
  fac := PRFAC(1, (p-1)/6, p, 1); ### Note the '6'
  ord := ifactor(order(fac, p));
                              p := 55681
                              fac := 9445
                              ord := (2)^5 (3)
```
(1)

At $p = 55681$, the first solutions (for $2 \leq s$) are at (min) $s = 5$.

All solutions are necessarily 'primitive' as this prime has no square support up to the highest factored level.

s = 5 (there are 8 known support primes). 18 primitive solutions, the least having 11, the largest 36 digits

```
> p := 55681:
  fac := 9445:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
  LEV := 5:
 LEVEL||LEV[p] := []:
  for lev from 2 to LEV do
for q||lev in level||lev[p] do LEVEL||LEV[p] := [ op(LEVEL||LEV[p]), q||lev ]; od od:
print(``); S_potential||p := LEVEL||LEV[p];
   count := 0:
  SOLN_L := []:
 print(``); print(_____________________); print(``);
for L_ in choose(S_potential||p, LEV) do if PI6_more_modified(L_, LEV, p, fac) = 1 then
   count := count+1:
print(``); print(L_); print(``); lprint(`Here is a solution:`); print(``);
print(p*mul(j, j = L_));
  SOLN_L := [op(SOLN_L), p*mul(j, j = L_)]:
print(``); lprint(`which has`, length(p*mul(j, j = L_)), `digits.`); print(_____________________);
fi; od:
print(``); lprint(`There were`, count, `solutions altogether.`); print(_____________________);
print(``);
```

S_potential55681 := [5, 29, 11, 2531, 41, 121477457, 17, 12075324422351249]

[5, 11, 17, 29, 41]

`Here is a solution:`

61901402915

`which has`, 11, `digits.`

[5, 11, 17, 41, 2531]

`Here is a solution:`

5402498302685

`which has`, 13, `digits.`

[5, 11, 17, 41, 12075324422351249]

`Here is a solution:`

257751559447346308966376l5

`which has`, 26, `digits.`

[5, 11, 29, 2531, 121477457]

`Here is a solution:`

27305840606101452065

`which has`, 20, `digits.`

[5, 11, 29, 121477457, 12075324422351249]

`Here is a solution:`

1302753393692956521284904624126 35

`which has`, 33, `digits.`

[5, 11, 2531, 121477457, 12075324422351249]

`Here is a solution:`

113698925497823205357658400126 33765

`which has`, 35, `digits.`

[5, 17, 29, 41, 2531]

`Here is a solution:`

14242950070715

`which has`, 14, `digits.`

[5, 17, 29, 41, 12075324422351249]

`Here is a solution:`

67952683854300390545680985

`which has`, 26, `digits.`

[5, 17, 41, 2531, 12075324422351249]

`Here is a solution:`

59306290632839409817627094 15

`which has`, 28, `digits.`

[5, 29, 2531, 121477457, 12075324422351249]

`Here is a solution:`

29975171267607935957928123669670835

`which has`, 35, `digits.`

[11, 17, 29, 2531, 121477457]

`Here is a solution:`

92839858060744937021

`which has`, 20, `digits.`

[11, 17, 29, 121477457, 12075324422351249]

`Here is a solution:`

44293615385560521723686757220295 9

`which has`, 33, `digits.`

[11, 17, 2531, 121477457, 12075324422351249]

`Here is a solution:`

38657634669259889821603856042954801

`which has`, 35, `digits.`

[11, 29, 41, 2531, 121477457]

`Here is a solution:`

223907892970031906933

`which has`, 21, `digits.`

[11, 29, 41, 121477457, 12075324422351249]

`Here is a solution:`

10682577828282243474536217917836 07

`which has`, 34, `digits.`

[11, 41, 2531, 121477457, 12075324422351249]

`Here is a solution:`

93233118908215028393279888103596873

`which has`, 35, `digits.`

[17, 29, 2531, 121477457, 12075324422351249]

`Here is a solution:`

10191558230986698225695562047688 0839

`which has`, 36, `digits.`

[29, 41, 2531, 121477457, 12075324422351249]

`Here is a solution:`

```
                                  24579640439438507485501061409130084 7

                                        `which has`, 36, `digits.`
                                        ____________________

                                    `There were`, 18, `solutions altogether.`
                                    ____________________
```

(3.1)

```
> print(``); sort([seq(length(n), n = SOLN_L)]);
```

[11, 13, 14, 20, 20, 21, 26, 26, 28, 33, 33, 34, 35, 35, 35, 35, 36, 36] (3.2)

```
> seq(PI6(n), n = SOLN_L);
```

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 (3.3)

s = 6 (there are exactly 12 known support primes). 306 primitive solutions, having between 14 and 84 digits

The probability came to 0.3312

```
> number_of_supports := 12:
                 LEV := 6:
  print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
  print(``);      number_of_solutions := 306;
  print(``);              probability := evalf(number_of_solutions/number_of_constructed_ns, 4);
```

number_of_constructed_ns := 924

number_of_solutions := 306

probability := 0.3312 (4.1)

```
>             p := 55681:
            fac := 9445:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
            LEV := 6:
   LEVEL||LEV[p] := []:
  for lev from 2 to LEV do
  for q||lev in level||lev[p] do LEVEL||LEV[p] := [op(LEVEL||LEV[p]), q||lev ]; od od:
  print(``);
  S_potential||p := LEVEL||LEV[p];
          count := 0:
            MIN := 10^100: ### Clearly greater than ANY small solution
            MAX := 0:
          print(``);
   for L_ in choose(S_potential||p, LEV) do if PI6_more_modified(L_, LEV, p, fac) = 1 then
              w := mul(q, q = L_):
            MIN := min(MIN, p*w):
```

```
            MAX := max(MAX, p*w):
          count := count+1:
  fi; od:
        lprint(`There were`, count, `solutions altogether,`);
  print(``); lprint(`generated from`, nops(S_potential||p), `support primes.`);
  print(``); lprint(`The smallest solution is:`); print(``); MIN;
  print(``); lprint(`The largest solution is:`); print(``); MAX; print(``);
  print(_____________________); print(``);
```

S_potential55681 := [5, 29, 11, 2531, 41, 121477457, 17, 12075324422351249, 257, 610817, 476600704619911891073, 4940395755423721544093464 97]

`There were`, 306, `solutions altogether,`

`generated from`, 12, `support primes.`

`The smallest solution is:`

15908660549155

`The largest solution is:`

48162932077560866286003622514811550791342466649841574869196418798519214754545288468 1

(4.2)

```
> length(MIN); length(MAX);
```

14

84 **(4.3)**

s = 7 (there are exactly 14 known support primes). 1140 primitive solutions, having between 19 and 371 digits

The probability came to 0.3322

```
> number_of_supports := 14:
                 LEV := 7:
  print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
  print(``);       number_of_solutions := 1140;
  print(``);               probability := evalf(number_of_solutions/number_of_constructed_ns, 4);
```

number_of_constructed_ns := 3432

number_of_solutions := 1140

probability := 0.3322 **(5.1)**

```
> p := 55681:
  fac := 9445:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
  LEV := 7:
 LEVEL||LEV[p] := []:
for lev from 2 to LEV do
for q||lev in level||lev[p] do LEVEL||LEV[p] := [op(LEVEL||LEV[p]), q||lev ]; od od:
print(``);
S_potential||p := LEVEL||LEV[p];
   count := 0:
    MIN := 10^100: ### Clearly greater than ANY small solution
    MAX := 0:
   print(``);
 for L_ in choose(S_potential||p, LEV) do if PI6_more_modified(L_, LEV, p, fac) = 1 then
      w := mul(q, q = L_):
    MIN := min(MIN, p*w):
    MAX := max(MAX, p*w):
   count := count+1:
fi; od:
  lprint(`There were`, count, `solutions altogether,`);
print(``); lprint(`generated from`, nops(S_potential||p), `support primes.`);
print(``); lprint(`The smallest solution is:`); print(``); MIN;
print(``); lprint(`The largest solution is:`); print(``); MAX; print(``);
print(_____________________); print(``);
```

S_potential55681 := [5, 29, 11, 2531, 41, 121477457, 17, 12075324422351249, 257, 610817, 476600704619911891073,
494039575542372154409346497, 652980135409104837678582610371183252063988 49,
5430065006389584982184906089683575088937603970131954895946359874894517106136643531669045869571635931693439 1\
0908654931360046282575343641758024671569328636366097721540815008280081881691707379526349679898975924600406 4\
9822049644112410252192520231933721 7]

`There were`, 1140, `solutions altogether,`

`generated from`, 14, `support primes.`

`The smallest solution is:`

9717280310653209635

`The largest solution is:`

279580450878270080935533165154211434159400842337452645729949051190322348928283627353466937877063734100523796576 8\
45211386867295171610742452674120310937098254669469794561590072856758115310080887018767909672544506876920973\
71800699762787601314966864947223040435048066568389613735504859168457435140559705120898997428436756306647610\
02438449925588198389768371163819905298705696 9

(5.2)

```
> length(MIN); length(MAX);
```

19

371 **(5.3)**

```
> PI6(9717280310653209635);
```

1 **(5.4)**

- **COMPLETE to here**