```
> ### Example 7.8. p = 331 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**

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

In this Maple-to-pdf conversion we exhibit all solutions of our paper's congruence (2.6) for the prime $p = 331$, **a specially chosen (because of the very large number of support primes) standard Jacobi prime.**

It should be emphasised that for $p = 331$ **the '**$s = 9$**' 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** $s = 10$ **would require knowing the complete factorisation of the 1291-digit** $331^{29} + 1$ **(that '9' being** $10 - 1$**).**

Note. At $p = 331$ **we have** $ord_p\left(\frac{1}{3}\ \{p-1\}\right)! = (3)$ **and** $ord_p\left(\frac{1}{6}\ \{p-1\}\right)! = 3.2^1$.

No square support.

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 $p = 331$

```
>              p := 331:

    level||2[p] := [5, 11, 29, 83, 1889]:              ### These divide (331 - 1)*(331 + 1)*
(331^2 + 1)

    level||3[p] := [17, 41]:                           ### These divide 331^(2^2) + 1

    level||4[p] := [14927201, 66411377]:               ### These divide 331^(2^3) + 1

    level||5[p] := [36833, 1776833, 6271510529, 185812754068491:

                                                       ### These divide 331^(2^4) + 1

    level||6[p] := [30977, 26705372033, 22651529502630467180252834145433 7,

                    1150085914749541327603538276348993]:

                                                       ### These divide 331^(2^5) + 1
                                                       ### All Maple obtained

    level||7[p] := [641, 3329, 4481, 51713, 1024640129, 20973135548033]:

                                                       ### These divide the 162-digit 331^(2^6) +
1
                                                       ### Our COMPLETE factorisation

    level||8[p] := [257, 345807320321]:                ### These divide the 323-digit 331^(2^7) +
1
                                                       ### factordb.com COMPLETE factorisation

    level||9[p] := []:                                 ### No 'q' for the 646-digit 331^(2^8) + 1
                                                       ### Maple provides a complete
factorisation
                                                       ### It is: 2*q, prime q = 1 (mod 6),
unfortunately

                                  ### COMPLETE TO HERE
```

```
              LEV := 9:
     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, 11, 29, 83, 1889, 17, 41, 14927201, 66411377, 36833, 1776833, 6271510529, 18581275406849, 30977, 26705372033, 22651529502630467180252834145433 7, 1150085914749541327603538276348993, 641, 3329, 4481, 51713, 1024640129, 20973135548033, 257, 345807320321]

(2.1)

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

true, true

(2.2)

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

25

(2.3)

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

5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5

(2.4)

```
> print(``);
  [[2], seq((p - 1)*(p + 1)*(p^2 + 1) mod q, q = level||2[p])];
  for LEV from 3 to 9 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, 0]
[[3], 0, 0]
[[4], 0, 0]
[[5], 0, 0, 0, 0]
[[6], 0, 0, 0, 0]
[[7], 0, 0, 0, 0, 0, 0]
[[8], 0, 0]

(2.5)

```
> ### 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 9 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;
```

(2.6)

Thus, in the case of $p = 331$, there is no square support.

```
>            p := 331;
          fac := PRFAC(1, (p-1)/6, p, 1); ### Note the '6'
          ord := ifactor(order(fac, p));
```

$p := 331$
$fac := 32$
$ord := (2)\ (3)$

(1)

s = 2 has # of supports = 3. 3 primitive solutions

```
>              p := 331:
              fac := 32:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
              LEV := 2:
   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(``); lprint(`generated from`, nops(S_potential||p), `support primes.`);

  print(_____________________); print(``);
```

$$S\_potential331 := [5, 11, 29, 83, 1889]$$

$$[29, 83]$$

`Here is a solution:`

796717

`which has`, 6, `digits.`

$$[29, 1889]$$

`Here is a solution:`

18132511

`which has`, 8, `digits.`

$$[83, 1889]$$

`Here is a solution:`

51896497

`which has`, 8, `digits.`

`There were`, 3, `solutions altogether,`

`generated from`, 5, `support primes.`

(3.1)

s = 3 has # of supports = 7. 5 primitive solutions

```
>              p := 331:
              fac := 32:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
              LEV := 3:
   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(``); lprint(`generated from`, nops(S_potential||p), `support primes.`);

  print(_____________________); print(``);
```

S_potential331 := [5, 11, 29, 83, 1889, 17, 41]

[5, 11, 17]

`Here is a solution:`

309485

```
`which has`, 6, `digits.`
```

[5, 11, 29]

`Here is a solution:`

527945

```
`which has`, 6, `digits.`
```

[5, 11, 41]

`Here is a solution:`

746405

```
`which has`, 6, `digits.`
```

[5, 11, 83]

`Here is a solution:`

1511015

```
`which has`, 7, `digits.`
```

[5, 11, 1889]

`Here is a solution:`

34389245

```
`which has`, 8, `digits.`
```

```
`There were`, 5, `solutions altogether,`
`generated from`, 7, `support primes.`
```

(4.1)

s = 4 has # of supports = 9. 41 primitive solutions

```
> p := 331:
  fac := 32:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
  LEV := 4:
 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(``); lprint(`generated from`, nops(S_potential||p), `support primes.`);

print(_____________________); print(``);
```

S_potential331 := [5, 11, 29, 83, 1889, 17, 41, 14927201, 66411377]

[5, 11, 14927201, 66411377]

`Here is a solution:`

18047271391482970285

`which has`, 20, `digits.`

[5, 17, 29, 41]

`Here is a solution:`

33452515

`which has`, 8, `digits.`

[5, 17, 29, 83]

`Here is a solution:`

67720945

`which has`, 8, `digits.`

[5, 17, 29, 1889]

`Here is a solution:`

1541263435

`which has`, 10, `digits.`

[5, 17, 41, 83]

`Here is a solution:`

95743405

`which has`, 8, `digits.`

[5, 17, 41, 1889]

`Here is a solution:`

2179027615

`which has`, 10, `digits.`

[5, 17, 83, 1889]

`Here is a solution:`

4411202245

`which has`, 10, `digits.`

[5, 29, 41, 83]

`Here is a solution:`

163326985

`which has`, 9, `digits.`

[5, 29, 41, 1889]

`Here is a solution:`

3717164755

`which has`, 10, `digits.`

[5, 29, 83, 1889]

`Here is a solution:`

7524992065

`which has`, 10, `digits.`

[5, 41, 83, 1889]

`Here is a solution:`

10638781885

`which has`, 11, `digits.`

[11, 17, 29, 41]

`Here is a solution:`

73595533

`which has`, 8, `digits.`

[11, 17, 29, 83]

`Here is a solution:`

148986079

`which has`, 9, `digits.`

[11, 17, 29, 1889]

`Here is a solution:`

3390779557

`which has`, 10, `digits.`

[11, 17, 41, 83]

`Here is a solution:`

210635491

`which has`, 9, `digits.`

[11, 17, 41, 1889]

`Here is a solution:`

4793860753

`which has`, 10, `digits.`

[11, 17, 83, 1889]

`Here is a solution:`

9704644939

`which has`, 10, `digits.`

[11, 29, 41, 83]

`Here is a solution:`

359319367

`which has`, 9, `digits.`

[11, 29, 41, 1889]

`Here is a solution:`

8177762461

`which has`, 10, `digits.`

[11, 29, 83, 1889]

`Here is a solution:`

16554982543

`which has`, 11, `digits.`

[11, 41, 83, 1889]

`Here is a solution:`

23405320147

`which has`, 11, `digits.`

[17, 29, 41, 14927201]

`Here is a solution:`

99870483072103

`which has`, 14, `digits.`

[17, 29, 41, 66411377]

`Here is a solution:`

444325517052631

`which has`, 15, `digits.`

[17, 29, 83, 14927201]

`Here is a solution:`

202176831584989

`which has`, 15, `digits.`

[17, 29, 83, 66411377]

`Here is a solution:`

899488241838253

`which has`, 15, `digits.`

[17, 29, 1889, 14927201]

`Here is a solution:`

4601349817639087

`which has`, 16, `digits.`

[17, 29, 1889, 66411377]

`Here is a solution:`

20471485407619999

`which has`, 17, `digits.`

[17, 41, 83, 14927201]

`Here is a solution:`

285836210171881

`which has`, 15, `digits.`

[17, 41, 83, 66411377]

`Here is a solution:`

1271690272943737

`which has`, 16, `digits.`

[17, 41, 1889, 14927201]

`Here is a solution:`

6505356638731123

`which has`, 16, `digits.`

[17, 41, 1889, 66411377]

`Here is a solution:`

28942444886635171

`which has`, 17, `digits.`

[17, 83, 1889, 14927201]

`Here is a solution:`

13169380512553249

`which has`, 17, `digits.`

[17, 83, 1889, 66411377]

`Here is a solution:`

58590803063188273

`which has`, 17, `digits.`

[29, 41, 83, 14927201]

`Here is a solution:`

487602946763797

`which has`, 15, `digits.`

[29, 41, 83, 66411377]

`Here is a solution:`

2169353995021669

`which has`, 16, `digits.`

[29, 41, 1889, 14927201]

`Here is a solution:`

11097373089600151

`which has`, 17, `digits.`

[29, 41, 1889, 66411377]

`Here is a solution:`

49372405983083527

`which has`, 17, `digits.`

[29, 83, 1889, 14927201]

`Here is a solution:`

22465413815532013

`which has`, 17, `digits.`

[29, 83, 1889, 66411377]

`Here is a solution:`

99949016990144701

`which has`, 17, `digits.`

[41, 83, 1889, 14927201]

`Here is a solution:`

31761447118510777

`which has`, 17, `digits.`

[41, 83, 1889, 66411377]

`Here is a solution:`

141307230917101129

`which has`, 18, `digits.`

`There were`, 41, `solutions altogether,`

```
`generated from`, 9, `support primes.`
```

(5.1)

s = 5 has # of supports = 13. 411 primitive solutions, having between 13 and 47 digits

The probability came to 0.3193

```
> number_of_supports := 13:
                 LEV := 5:
  print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
  print(``);      number_of_solutions := 411;
  print(``);              probability := evalf(number_of_solutions/number_of_constructed_ns, 4);
```

$$number\_of\_constructed\_ns := 1287$$

$$number\_of\_solutions := 411$$

$$probability := 0.3193$$

(6.1)

```
>             p := 331:
            fac := 32:  ### 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:
            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(``);
```

$$S\_potential331 := [5, 11, 29, 83, 1889, 17, 41, 14927201, 66411377, 36833, 1776833, 6271510529, 185812754068 49]$$

`There were`, 411, `solutions altogether,`

`generated from`, 13, `support primes.`

`The smallest solution is:`

1048983893861

`The largest solution is:`

679427541075069626518962066727858106233141192 91

(6.2)

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

13

47 (6.3)

```
> PI6(MIN); PI6(MAX);
```

1

1 (6.4)

s = 6 has # of supports = 17. 4362 primitive solutions, having between 11 and 110 digits

The probability came to 0.3525

```
> number_of_supports := 17:
                LEV := 6:
print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
print(``);      number_of_solutions := 4362;
print(``);              probability := evalf(number_of_solutions/number_of_constructed_ns, 4);
```

$$number\_of\_constructed\_ns := 12376$$

$$number\_of\_solutions := 4362$$

$$probability := 0.3525$$ (7.1)

```
>             p := 331:
            fac := 32:  ### 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_potential331 := [5, 11, 29, 83, 1889, 17, 41, 14927201, 66411377, 36833, 1776833, 6271510529, 18581275406849, 30977, 26705372033, 22651529502630467180252834145433 7, 1150085914749541327603538276348993]

```
`There were`, 4362, `solutions altogether,`
`generated from`, 17, `support primes.`
`The smallest solution is:`
```

30542146195

```
`The largest solution is:`
```

1782151434242837800264702055169616699048454591405226282998935062379771467883416429879564355672648571753624073 1

(7.2)

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

11

110 (7.3)

```
> PI6(MIN);
```

1 (7.4)

s = 7 has # of supports = 23. 81 690 primitive solutions, having between 15 and 123 digits

The probability came to 0.33322

```
> number_of_supports := 23:
                 LEV := 7:
  print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
  print(``);      number_of_solutions := 81690;
  print(``);              probability := evalf(number_of_solutions/number_of_constructed_ns, 5);
```

$$number\_of\_constructed\_ns := 245157$$

$$number\_of\_solutions := 81690$$

$$probability := 0.33322 \qquad \textbf{(8.1)}$$

```
>             p := 331:
            fac := 32:  ### 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(``);
```

$$S\_potential331 := [5, 11, 29, 83, 1889, 17, 41, 14927201, 66411377, 36833, 1776833, 6271510529, 18581275406849, 30977, 26705372033, 22651529502630467180252834145433 7, 11500859147495413276035382763489 93, 641, 3329, 4481, 51713, 1024640129, 20973135548033]$$

`There were`, 81690, `solutions altogether,`

`generated from`, 23, `support primes.`

`The smallest solution is:`

136859357099795

`The largest solution is:`

```
3737730359749645693045976013031528032816815811653336563046524547027974730624035390363449328889776683529455914823\
00301532123
```

(8.2)

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

15

123 (8.3)

```
> PI6(MIN);
```

1 (8.4)

s = 8 has # of supports = 25. 360 381 primitive solutions (out of the 1 081 575 candidates), having between 17 and 135 digits

The probability came to 0.33320

That largest solution is, in fact:

331 * 66411377 * 6271510529 * 26705372033 * 345807320321 * 18581275406849 * 20973135548033 *

2265152950263046718025283414543337 * 1150085914749541327603538276348 99

The time taken to effect that (remarkable) computation was only 19.32 SECONDS

```
> number_of_supports := 25:
                LEV := 8:
  print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
  print(``);      number_of_solutions := 360381;
  print(``);              probability := evalf(number_of_solutions/number_of_constructed_ns, 5);
```

number_of_constructed_ns := 1081575

number_of_solutions := 360381

probability := 0.33320 (9.1)

```
>             st:= time[real]():
              p := 331:
            fac := 32:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
            LEV := 8:
   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(``);
lprint(`The time taken to effect this computation was`, time[real]() - st, `SECONDS.`);
```

S_potential331 := [5, 11, 29, 83, 1889, 17, 41, 14927201, 66411377, 36833, 1776833, 6271510529, 18581275406849, 30977, 26705372033, 22651529502630467180252834145433 7, 11500859147495413276035382763489 93, 641, 3329, 4481, 51713, 1024640129, 20973135548033, 257, 345807320321]

`There were`, 360381, `solutions altogether,`

`generated from`, 25, `support primes.`

`The smallest solution is:`

35172854774647315

`The largest solution is:`

1292534519787472293541407869318476084716375625295662961465850725659892286220799494457053598685839484903427606376\
0462957383374533217148 3

`The time taken to effect this computation was`, 19.313, `SECONDS.`

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

17

135 **(9.2)**

```
> PI6(35172854774647315);
```

1 **(9.3)**

```
> length
  (1292534519787472293541407869318476084716375625295662961465850725659892286220799494457053598685839484903427606376046295738337 45332171483);
```

135 **(9.4)**

```
> ifactor(35172854774647315);
```

(5) (11) (17) (29) (41) (83) (257) (331) (4481) **(9.5)**

```
> n :=
  1292534519787472293541407869318476084716375625295662961465850725659892286220799494457053598685839484903427606376046295738337 45332171483;
```

n := **(9.6)**

```
1292534519787472293541407869318476084716375625295662961465850725659892286220799494457053598685839484903427 6\
0637604629573833745332171483
> is(n = 331 * 66411377 * 6271510529 * 26705372033 * 345807320321 * 18581275406849 * 20973135548033
  *
  2265152950263046718025283414543337 * 1150085914749541327603538276348993);
                                                    true                                          (9.7)
> ifactor(66411377);
                                                (66411377)                                        (9.8)
```

s = 9 has # of supports ALSO = 25. 680 973 primitive solutions (out of the 2 042 975 candidates), having between 20

and 142 digits. The probability came to 0.33332

That largest solution is:

331 * 1776833 * 6271510529 * 18581275406849 * 26705372033 * 2265152950263046718025283414543337

*** 1150085914749541327603538276348993 * 1024640129 * 20973135548033 * 345807320321**

The time taken to effect that (remarkable) computation was only 49.25 SECONDS

1776833 divides $331^{24} + 1$

6271510529 divides $331^{24} + 1$

18581275406849 (divides $331^{24} + 1$

26705372033 (divides $331^{25} + 1$

2265152950263046718025283414543337 divides $331^{25} + 1$

1150085914749541327603538276348993 divides $331^{25} + 1$

1024640129 divides $331^{26} + 1$

20973135548033 divides $331^{26} + 1$

345807320321 divides $331^{28} + 1$

```
> number_of_supports := 25:
                 LEV :=  9:
  print(``); number_of_constructed_ns := binomial(number_of_supports, LEV);
  print(``);      number_of_solutions := 680973;
  print(``);              probability := evalf(number_of_solutions/number_of_constructed_ns, 5);
```

$$number\_of\_constructed\_ns := 2042975$$

$$number\_of\_solutions := 680973$$

$$probability := 0.33332 \qquad \textbf{(10.1)}$$

```
>              st:= time[real]():
               p := 331:
             fac := 32:  ### THE ABOVE PRE-COMPUTED VALUE OF fac
             LEV := 9:
   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(``);
  lprint(`The time taken to effect this computation was`, time[real]() - st, `SECONDS.`);
```

$$S\_potential331 := [5, 11, 29, 83, 1889, 17, 41, 14927201, 66411377, 36833, 1776833, 6271510529, 18581275406849, 30977, 26705372033, 22651529502630467180252834145433 7, 1150085914749541327603538276348993, 641, 3329, 4481, 51713, 1024640129, 20973135548033, 257, 345807320321]$$

`There were`, 680973, `solutions altogether,`

`generated from`, 25, `support primes.`

`The smallest solution is:`

16749602299088905235

`The largest solution is:`

3543379249455058721072560513097692520545223747635695325291959732758336766241432306561453370604242199116483156572\
45213716647254911362438626 5803

_____

```
`The time taken to effect this computation was`, 49.252, `SECONDS.`
```

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

20

142 **(10.2)**

```
> PI6(MIN);
```

1 **(10.3)**

```
> ifactor(MIN);
```

(5) (11) (17) (29) (41) (83) (257) (331) (641) (3329) **(10.4)**

```
> MAX;
```

354337924945505872107256051309769252054522374763569532529195973275833676624143230656145337060424219911648315657\
2 **(10.5)**

452137166472549113624386265803

```
> print(``);
```

```
is(MAX = 331 * 1776833 *  6271510529 *  18581275406849 * 26705372033 *
226515295026304671802528341454337 *

1150085914749541327603538276348993 *  1024640129 * 20973135548033 * 345807320321);
```

true **(10.6)**

• COMPLETE to here

Rechecked Friday 8th May 2015.