Root filling

Pseudo-random screen filling using Primitive Roots: a related challenge regarding Abundant Numbers, presented by Mike Mudge.

Some of the ideas featured this month have appeared previously in Numbers Count; references are given, however the article is completely self-contained.

The area of application to PSEUDO-RANDOM SCREEN FILLING together with the final challenge regarding maximal abundance (within a given interval) are due to George Sassoon of the Isle of Mull, Argyllshire. George programs in UBasic v8.6 with Toshiba T3200SX hardware but is hoping to use a friend's 75MHz 486DX2 to continue his investigations.

Mathematical background

(A) Abundant, PERFECT & Deficient Numbers (PCW October 1983)

Let Sigma(n) denote the sum of all the positive integer factors of a given positive integer,n; frequently S(n) denotes Sigma (n) - n.

A number is called PERFECT if and only if (iff) Sigma(n)=2n i.e. S(n)=n: it is Abundant iff Sigma(n) is greater than 2n and, finally, deficient iff Sigma(n) is less than 2n.

Note: If n is an even PERFECT number then there exists m such that 2^m-1 is a prime number (called a Mersenne Prime) and $n = 2^{m-1}(2^m-1)$. e.g. m=2,3,5,...generating 6,28,496,...

A number, n, is said to be WEIRD iff it is abundant and there is no set of DISTINCT DIVISORS of it which sum to 2n. The sequence of WEIRD NUMBERS includes 70, 836, 4030,...

(B) Primitive Roots, Euler's TOTIENT Function (*PCW* February 1986, September 1990; May 1985)

(i) (a,n) denotes the greatest common divisor of a & n, thus (a,n)=1 means that a & n are co-prime.

(ii) If (a,n)=1 then the smallest positive k such that $a^k \equiv 1 \pmod n$ (i.e. a^k leaves the remainder 1 when divided by n) is called the order of a modulo n and written $ord_n a$.

(iii) Phi(n), Euler's Totient Function, is the number of integers,x, satisfying 1 less than or equal to x less than or equal to n and such that (x,n)=1.

Definition: A number,a, with (a,n)=1 and $ord_n a$ = Phi(n) is called a PRIMITIVE ROOT of a modulus n.

G Sassoon's Pseudo-Random Screen Filling Algorithm

(As used 'for doing fractals and other things which take a long time to execute, so you can get a rough idea of the shape of the thing before it has been running too long.')

Say the screen is 640*480 (VGA):

1. Find the first prime greater than 640*480.
2. Find a PRIMITIVE ROOT of this prime.
3. Raise the primitive root to successive powers modulo the prime, and use these to generate X and Y coordinates, discarding any which are off the screen.
4. When the power comes back to 1, all points on the screen have been filled except for (0,0) which must be done separately.

QUESTION ONE How does this algorithm compare with others?

Now, if any given positive integer,n, has the prime power decomposition $n = p_1^{n1}p_2^{n2}p_3^{n3}...p_k^{nk}$ then $Phi(n) = n(1-1/p_1)(1-1/p_2)(1-1/p_3)(...(1-1/p_k)$.

G Sassoon's Primitive Root Finding Algorithm

1. Find the prime factors of P-1 and store them in one array and their powers (the n_i above) in another, using NPF for the k: 'Number of (distinct) prime factors'.
2. Find every combination in succession of 1,2,3..NPF of the prime factors.
3. For each of these combinations, find every combination of the powers of the primes in it, and generate a factor from them.
4. The procedure also generates N itself but not the factor 1, so adjust the result accordingly.

QUESTION TWO 'Finding primitive roots...is not as trivial as it appears at first sight.' Is there a better way of doing it?

Concentrating now on Abundant Numbers, defining the Abundance Ratio as (Factor Sum)/N = S(N)/N = F: It is easy to generate N's which have large F-values: e.g. 1010824870255200 = $2^5x3^3x5^2x7^2x11x13x17x19x23x29x31$

Factor Sum = S(N) = 5194025993872800

F-value = 5.138, number of factors 27647.

QUESTION THREE G Sassoon's Abundant Numbers Challenge!

Determine the integers less than 10^M having the greatest F-value for M= 1,2,3,...16 say, why stop there? (Because it is the limit of integer precision in some common versions of Basic.) What is the most efficient algorithm for doing this?

Answers to some, or all, of the above questions may be sent to Mike Mudge, 22 Gors Fach, Pwll-Trap, St Clears, Carmarthen, Dyfed SA33 4AQ, tel (0994) 231121, to arrive by the closing date. It would be appreciated if such submissions contained a brief description of the hardware used, details of coding, run times and a summary of results obtained, all in a form suitable for publication in *PCW*.

Review of Numbers Count -123-, *PCW* July 1993: Exercises in symmetry

Paul Rayner's invitation to the Chess Board generated a number of detailed responses, among them:

John Taylor used QBasic on a 286 machine, some symmetry concepts, and counted the number of unrotated (unduplicated), unreflected solutions to the Multi-Queens problem upto N=9, giving 46. The result for N=8 i.e. 12 taking 27 secs.

Stephen Poley of Holland used a 25MHz 80486SX programmed in Pascal, confirming the quoted results for F(N) and extending to all board sizes upto 16, with similar execution times. Stephen then had limited success with a mathematical model, generating 'ballpark' (well, nearly!) figures for boards, both square and rectangular, upto 20x20.

However, the very worthy winner this month is Chris Higley of 31 Uppercliff Close, Penarth, South Glamorgan CF64 1BE. Extension to 20x20 in 177 hours on a DEC Alpha in Turbo C, empirical formula for estimating the number of solutions:

$0.3886^{n-1}x\ n!$

and for the computation time on the given system:

$0.42^n x\ n!\ x\ 10^{-5}$ secs.

Congratulations on the attached graphical output, Chris.