Grand Summary for Project 997

The following table shows the summary of results in the various documents of Project 997.

Algorithm	Equation #	Upper Mean Penalty Factor	Document Files Part Number
frac(π+9997*r)	End of section 2	147.6264391	1
frac(0.8502+1237*r)	End of section 2	147.7205	1
frac(0.047907 + 15701*r)	End of section 2	147.7246371	1
frac(π +110011*r)	End of section 2	147.9719176	1
Double-random number generators [1]	5.1, 5.2	143.0952581	1
Whichmann-Hill Variant	2.5, 2.6	147.781193	2
ACORN Variant 1	2.2	147.769423	3
ACORN Variant 2	3.2	147.7426001	3
MRG32a Variant 1	2.2	147.773747	4
MRG32a Variant 2	3.1	147.7724717	4
Power Method version 1	2.2	147.776513	5
LCGM Squared version 1	2.2	147.6341104	6
LCGM Squared version 2	3.2	147.1263891	6
LCGM Cubed version 1	2.2	147.763261	7
LCGM Cubed version 2	3.2	144.480268	7

Table 1. Summary results listed by document part number.

Table 2 shows the results sorted by the upper mean penalty factors. The double-random number generators and the LCGM cubed version2 have significant advantages over the other algorithms.

Algorithm	Equation #	Upper Mean Penalty Factor	Document Files Part Number
Double-random number generators [1]	5.1, 5.2	143.0952581	1

LCGM Cubed version 2	3.2	144.480268	7
LCGM Squared version 2	3.2	147.1263891	6
frac(π+9997*r)	End of section 2	147.6264391	1
LCGM Squared version 1	2.2	147.6341104	6
frac(0.8502+1237*r)	End of section 2	147.7205	1
frac(0.047907 + 15701*r)	End of section 2	147.7246371	1
ACORN Variant 2	3.2	147.7426001	3
LCGM Cubed version 1	2.2	147.763261	7
ACORN Variant 1	2.2	147.769423	3
MRG32a Variant 2	3.1	147.7724717	4
MRG32a Variant 1	2.2	147.773747	4
Power Method version 1	2.2	147.776513	5
Whichmann-Hill Variant	2.5, 2.6	147.781193	2
frac(π +110011*r)	End of section 2	147.9719176	1

Table 2. Summary results sorted by the values of the upper mean penalty factors.

The best algorithm is:

$z = frac(991*r_n)$ (5.1 in Part 1)

$r_{n+1} = frac(11111*z)$ (5.2 in Part 1)

The second-best algorithm is:

```
M = 2^24-1
 a0 = 34876
 a1 = 9754
 a2 = 45847
 a3 = 29574
  ix(1) = round(rand*11, 0);
  ix(2) = a0+a1*ix(1) mod M
  for i=1 to maximum number of random numbers
    ix(3) = a0+a1*ix(1)^2 +a2*ix(2) mod M
    x(i) = ix(3)/M
    ix(1:2) = ix(2:3)
  end
```

(3.2 in Part 7)

The value x(i) is the uniform random number generated in the range of 0 to 1 (excluded) in each iteration.

[1] All the results of the various flavors of the double-random number generators in tables 5.1 and 5.2 of Part 1 show values for the upper mean penalty factors below 147! So basically, any one of the algorithms should be a good PRNG.