1 Lagged Fibonacci random number generator

The idea here is to generate i_n from some older random numbers. The choice

$$i_n = (i_{n-r} + i_{n-s}) \bmod m$$

with $r = 24$ and $s = 55$ is common and will give the random number generator a cycle of at least $2^{55} \approx 10^{16}$. Our code uses `unsigned int` and $m = 2^{32}$ which means that the modulus operation is made automatically through overflow in the addition. The generator is first initialized with random numbers produced with another method.

2 Usage

You can find both `ran.h` and `ran.c` at `http://www.tp.umu.se/mc/src` together with this documentation, `ran-doc.pdf`. In the beginning of the source file you should use

```
#include "ran.h"
```

Initialization

The random number generator has to be initialized before it can produce any random numbers. The initialization is done with a call

```
init_ran(seed)
```

where `seed` is a user-specified integer. If the function is called with seed equal to zero, a new seed is generated from the system clock. The uninitialized random number generator will always return zero.

Integer values

Since the generator works with integers the most direct routines return integers. A call to `iran()` will return a positive integer in the range $[0, 2^{31} - 1]$. Similarly, a call to `iran_sign()` returns a signed integer.

Floating point values

The standard random numbers, floating point values in the range $[0, 1)$ are given by `dran()`. There is also a signed version `dran_sign()` that gives values in the range $[-1, 1)$.