A Note on the Bays–Durham Shuffle

This version of the document is dated 2020-11-03.

The Bays–Durham shuffle extends a pseudorandom number generator's (PRNG) maximum cycle length by giving it a bigger state. Generally, for a size of `tablesize`, this maximum is at most the factorial of `tablesize`, which is about the number of ways to arrange a list of size `tablesize`.

The following describes the Bays–Durham shuffle with a size of `tablesize`. (C++'s `shuffle_order_engine` implements something similar to the shuffle described below.) For PRNGs that output 32- or 64-bit integers 0 or greater, a `tablesize` of 256, 512, or 1024 is suggested.

- To initialize, fill a list with as many numbers from the underlying PRNG as `tablesize`, then set `k` to another number from that PRNG.
- For each "random" number, take the entry at position (`k` % `tablesize`) in the list, where '%' is the remainder operator and positions start at 0, then set `k` to that entry, then replace the entry at that position with a new number from the underlying PRNG, then output `k`.

The following variant of the Bays–Durham shuffle was used in the Kybos PRNG by **J. Baagøe**:

- Initialize the table with random numbers (such as those from the underlying PRNG), then set `k` to the first entry of the table.
- For each "random" number, set `v` to the entry at position (`k % tablesize`) in the list, where '%' is the remainder operator and positions start at 0, then replace the entry at that position with `v` minus a new number from the underlying PRNG (using wraparound subtraction), then set `k` to the result, then output `v`.