Hash Performance Test

Two hashing functions are described in the Hashing section. In order two decide which of these I would use in the algorithm, a few tests were performed in order to determine the speed of both.

To perform these tests, I wrote a short java program. It randomized a given number of k-mer transformations[1], and then counted the number of nanoseconds it took the Carter Wegman- and multiply-shift hashing schemes to hash all the transformations. The multiply-shift algorithm was set to shift only if the input was longer than 32 bits. In this case, the shift was $2 \cdot k - 32$.

The results of the tests can be seen in Figure 1. As we can note, the Carter Wegman implementation is consequently smaller than multiply-shift, in addition to having larger margin of error. We may note that the difference in speed is most notable at $k = 10$, diminishing at higher k. It should be noted however that the differences in speed are less than 50 ms, even at 100 mio. input hash values. Also, as the time increases linearly with the number of input values, so will the difference in speed.

Conclusively, since the multiply-shift proved slightly quicker than the Carter Wegman scheme, I shall use **multiply-shift** for my algorithm.

k-mer size	10		20		30	
# of hasvalues	Mult.shift	Carter	Mult.shift	Carter	Mult.shift	Carter
1 mio.	10.3 ± 0.2	11.9 ± 0.5	10.3 ± 0.1	11.6 ± 0.1	10.5 ± 0.1	11.6 ± 0.1
10 mio.	40.9 ± 0.5	42.1 ± 0.2	41.3 ± 0.1	42.1 ± 0.4	41.8 ± 1.6	42.4 ± 0.5
50 mio.	175.5 ± 3.4	187.2 ± 10.6	177.5 ± 2.1	181.6 ± 3.8	176.2 ± 1.3	179.7 ± 1.7
100 mio.	343.0 ± 1.4	377.5 ± 14.0	341.6 ± 1.0	364.1 ± 12.6	346.7 ± 6.9	355.8 ± 3.5

Figure 1: ms for multiply-shift- and Carter Wegman hashing schemes to 4 different numbers of randomized k-mer transformations.

[1] which are 2·k bits long